National University of Ireland Galway

Doctoral Thesis

# Suggestion Mining from Text

*Author*
Sapna Negi

*Supervisor*
Dr. Paul Buitelaar

*A thesis submitted in fulfillment of the requirements*
*for the degree of Doctor of Philosophy*

*at the*

Insight Centre for Data Analytics
National University of Ireland Galway

February 2019

# Declaration of Authorship

I, Sapna Negi, declare that this thesis titled, *Suggestion Mining from Text*, and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

*"The important thing is not to stop questioning.*
*Curiosity has its own reason for existing."*

Albert Einstein

# Abstract

With the ever growing availability of opinions on the web, opinion mining has become a popular area of research in the fields of natural language processing and applied machine learning. We argue that opinion summaries based solely on the sentiment polarity of text towards an entity of interest, excludes explicit recognition and extraction of information appearing in the form of recommendations, tips, and advice. Suggestions and advice are often sought by different stakeholders through surveys and suggestion forms, where readers are explicitly asked to provide suggestions. On the other hand, customers tend to spontaneously mention suggestions for new features in the product reviews or tweets about the product, and similarly recommendations for nearby places to eat are often spotted in the hotel reviews. Yet sentiment analysis remains the most popular opinion mining task performed on these data sources. In this thesis, we investigate the automatic extraction of suggestions from text, which is referred to as *Suggestion Mining*. Suggestion Mining is framed as a sentence classification task, where sentences in a given text are to be automatically labeled as suggestions and non-suggestions.

Given the very limited amount of related work, suggestion mining can be considered as a young research problem. Therefore, research questions investigated in this dissertation address some of the core aspects of suggestion mining. This includes, task definition where the scope of suggestion and non-suggestion classes is formally defined, benchmark datasets are developed, manually identified features for supervised learning methods, as well as representation learning are evaluated, and distant supervision approaches under the lack of domain specific training datasets are introduced. While covering these aspects, this thesis primarily revolves around two computational tasks, sentence classification and representation learning. The thesis adopts some of the popular deep learning concepts and methods for suggestion mining, like Word Embeddings and Long Short Term Memory Networks. This also opens up a young sentence classification task, and corresponding benchmark datasets to the deep learning community.

The contributions of this thesis are manifold. A formal task definition for suggestion mining is provided, which accompanies qualitative and quantitative analysis of suggestions and a formal definition of suggestions in the context of suggestion mining. Benchmark datasets from multiple domains are developed and released, accompanied by a robust data annotation methodology which balances the cost and quality of manual annotations. An in-depth evaluation of the method of using manually selected features with Support Vector Machine classifiers is performed in domain specific, domain independent, and cross domain training scenarios. It is demonstrated that a combination of features from the related work and our newly proposed features outperform the models which

use either of them. It is also discovered that the syntactic features consistently remain the top performing features in all the experiments. A major contribution of the thesis is creation of a large silver standard dataset composed of sentences from Wikihow and Wikipedia, and validation of a method to use this dataset for automatically learning features, i.e., representation learning. Experiments comparing manually selected features with automatic feature learning, i.e. word embeddings prove that the embeddings which represent part of speech tags outperform the state of the art pre-trained word embeddings for this task. This thesis performs an end to end exploration of suggestion mining, with all evaluations performed for domain specific, open domain and cross domain classification scenarios.

# *Acknowledgements*

The research presented in this thesis was financed by Science Foundation Ireland under grant no. SFI/12/RC/2289 (Insight), and the European Union financed projects EUROSENTIMENT (FP7-ICT-296277) and MixedEmotions (H2020-644632).

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Overview

Online text is becoming an increasingly popular source to acquire public opinions towards entities like persons, brands, products, services etc. Opinion mining is a research area which investigates the extraction of opinions towards a target entity from the text written by a set of people, for example, collecting opinions towards iPhone by performing opining mining on a large number tweets containing iphone related hashtags. State of the art opinion mining systems primarily process text to provide a summary of positive and negative sentiments towards the entities of interest, primarily by the means of sentiment analysis methods [35, 59] methods. We observe that opinionated text also contains descriptive information like advice, tips, warnings, recommendations etc, which is generally overlooked in a sentiment based summary, mostly because it does not contain any sentiment towards the target entity. On the other hand, this kind if information is explicitly sought by the stakeholders. For example, 'Be sure to pick up an umbrella at the concierge if you anticipate rain while sightseeing.' and 'An electric kettle would have been a nice addition to the room', are sentences found in a hotel review but does not express the sentiment of a reviewer towards the hotel. Therefore, in the case of commercial products and services, suggestions present in the reviews can convey ideas for improvements to the brand owners, or tips and advice to fellow customers. We collectively refer to these expressions as *suggestions*. Therefore, suggestion mining can be defined as *the extraction of suggestions from the* source text *where suggestions of interest are likely to appear*. We stress that suggestion mining can be seen as a research problem under the broader area of opinion mining, akin sentiment analysis. However, sentiments and suggestions are not completely disjoint, and we will elaborate this later in the thesis.

Suggestion mining is an emerging research area, and thus the problem definition is still in an early stage, as well as the labeled datasets are in scarcity. So far, only a few studies have been performed on suggestion mining, mostly from an application perspective, and therefore most of the studies are focused around collecting suggestions for improvement in products using product reviews as a source text, with approaches tailored only for this domain. We assert that suggestion mining is a generic language processing task, similar to sentiment analysis, which can be associated with common linguistic properties and challenges irrespective of the domain and application. Secondly, the previous works mostly proposed rule based methods, and a very few of them developed statistical classifiers which employed manually identified features. Suggestion mining haven't yet received the attention of applied deep learning research, where features are automatically learned [22]. Speaking of statistical learning approaches, there is a clear lack of training datasets for suggestion mining, either due to the proprietary nature of previous studies or due to the implementation of rule based systems.

This thesis focuses on some foundational aspects of the problem of suggestion mining, which includes qualitative and quantitative properties of suggestions and how these properties vary across domains, defining the scope of suggestions to formalize the task definition, and curating training and evaluation datasets. Moving to the computational aspects, suggestion mining in this thesis is defined as a problem of classifying sentences from a given source text into *suggestion* and *non-suggestion* classes. We propose and evaluate machine learning methods for suggestion mining, with a focus on applying them to a wider range of domains, as compared to the previous studies. These methods primarily include Support Vector Machines (SVM) and Long Short Term Memory Networks (LSTM) as classification algorithms, aided with a variety of features which are either manually identified or automatically learned. We also propose and evaluate a feature learning methods (representation learning) using LSTMs. The domain independence of different models, as well as cross domain training is also evaluated, keeping in mind the application scenarios where training datasets specific to a target domain are unavailable. The studied domains include, hotel reviews, electronics reviews, travel discussion forums, and software suggestion forum. We also develop a large and open domain silver standard dataset for suggestion mining, and investigate into different ways to exploit this dataset using representation learning.

## 1.2 Challenges

As mentioned earlier, suggestion mining is primarily defined as a sentence classification task, which is mostly applicable on text obtained from the web. Therefore, it faces

| **Source, Entity/Topic** | **Sentence** | **Sentiment Label** | **Linguistic Properties** |
|---|---|---|---|
| Reviews, Electronics | I would recommend doing the upgrade to be sure you have the best chance at trouble free operation. | Neutral | Subjunctive, Imperative mood |
| Reviews, Electronics | My one recommendation to creative is to get some marketing people to work on the names of these things | Neutral | Imperative |
| Reviews, Hotels | Be sure to specify a room at the back of the hotel. | Neutral | Imperative |
| Reviews, Hotel | The point is, don't advertise the service if there are caveats that go with it. | Negative | Imperative |
| Tweets, Windows Phone | Dear Microsoft, release a new zune with your wp7 launch on the 11th. It would be smart | Neutral | Imperative, subjunctive |
| Discussion thread, Travel | If you do book your own airfare, be sure you don't have problems if Insight has to cancel the tour or reschedule it | Neutral | Conditional, imperative |
| Suggestion forum, Software | Please provide consistency throughout the entire Microsoft development ecosystem! | Negative | Imperative |

TABLE 1.1: Examples of suggestions from different domains, about different entities and topics, and intended for different receivers.

similar language processing challenges as other sentence or short text classification tasks, like stance detection [44], tweet sentiment classification [63], specially when a task is newly introduced. These challenges include task formalization and data annotation, understanding sentence level semantics, dealing with extra propositional properties of sentences like mood and modality, emotions, sentiments, etc. Below, we elaborate some of the observed challenges in suggestion mining:

- **Problem definition and annotation agreement:** The previous rule based approaches assumed that suggestions can be expressed in a straightforward way using standard expressions like 'I recommend', 'I suggest that', 'You should', and created small evaluation datasets which were labeled in-house [11, 60]. However, we observe a low inter-annotator agreement in labeling sentences as suggestions and non-suggestions when no specific annotation guidelines are provided and layman annotators are simply asked to label sentences based on their understanding of the term suggestion. This is mainly due to differences in the perception of implicitly expressed suggestions and understanding of the context in which a sentence appeared. For example, 'Definitely come with a group and order a few plates to share.' was a sentence obtained from a restaurant review, which was labeled as a suggestion by 52% of 50 annotators.

- **Training data:** Most of the domains analyzed by us exhibit a sparse appearance of suggestion sentences among the text. Therefore, the class distribution in training

datasets remain highly imbalanced, which is unfavorable for supervised learning approaches. For example, hotel reviews tend to contain 6 - 13 suggestions in every 100 sentences. It also results in higher data annotation costs, since more data instances are required to be annotated in order to obtain a fair number of suggestion sentences in the datasets. Only 2 training datasets are available from previous studies, which cover travel discussion forum [72] and twitter domain [18], while the customer review datasets from the previous works remain proprietary [43, 60, 70].

- **Figurative expressions:** The text from social media and other sources usually contain figurative use of language, which demand pragmatic understanding from the automated systems. For example, 'Try asking for extra juice at breakfast - its 22 euros!!!!!' is more of a sarcasm than a suggestion. Therefore, a sentence in the form of suggestions may not always be a suggestion, and vice versa. The presence of a variety of linguistic strategies used in the suggestion expressing sentences makes this task interesting from a computational linguistics perspective.

- **Context dependency:** In some cases, context plays a major role in determining whether a sentence is a suggestion or not. For example, 'There is a parking garage on the corner of the Forbes showroom.' can be labeled as a suggestion (for parking space) when it appears in a restaurant review and the human annotator gets to read the full review which otherwise contains opinion about the restaurant, while the same sentence would not be labeled as a suggestion if it is present in the description of the locality of a Forbes showroom.

- **Long and complex sentences:** Often, a suggestion is only expressed in one part of a long sentence, or is elaborated in a very long sentence, like, 'I think that there should be a nice feature where you can be able to slide the status bar down and view all the push notifications that you got but you didn't view, just like android and IOS, but the best part is that it fixes many problems like when people wanted a short cut to turn WiFi on and off and data on and off so that would be a nice feature to have 2'. This increases the chances that same words appear in suggestion as well as non-suggestion sentences of the datasets from a particular domain, which might negatively effect the performance of classifiers using bag of words features. Also in the case of part of speech tag related features, the accuracy of tagger is lower for such sentences.

## 1.3 Terminology

**Suggestion:** Prior to this study, a few more studies performed experiments on suggestion mining. Some of them referred to the kind of information we identify as suggestions,

using different but related terms, like *advice* [72], *wishes for improvements* [60], *suggestions for improvements* [11]. All of these terms essentially referred to the expressions of advice, tips, warnings, and recommendations found in opinionated text.
Therefore, a suggestion can be referred to as a advice, tip, or suggestion, depending on the situation and context. However, none of the previous works have chosen their preferable term based on a strict linguistic definition for these terms. We choose to remain with the term *Suggestion*, since it was used in most of the related works. The Oxford dictionary defines *Suggestion* as "An idea or plan put forward for consideration", and lists *recommendation, advice, proposal, tips* as synonyms.

**Domain:** In this thesis we primarily refer to the term *Domain* as the source of the candidate text. Some example sources could be hotel reviews, electronics reviews, travel discussion forums, or tweets. Domains can further be classified into the sub-domains but still be referred to as domains. For example, we refer to the hotel reviews and electronic reviews as two different domains. It is possible that the two sample texts discuss same kind of entity but are obtained from two different domains. For instance, hotel reviews and travel discussion forums are two separate domains, but two sample texts obtained from these domains may talk about hotels.

**Cross Domain Training:** When a classifier is trained on a dataset which belongs to a different domain than the domain it is being evaluated on.

**Open Domain Training:** When a classifier is trained on datasets from multiple domains, including the domains on which it is being evaluated.

## 1.4 Research Questions and Contributions

All the previous works on suggestion mining framed it as a text classification problem. Following the same, and relying upon the assumption that a suggestion can be expressed within a single sentence, we broadly define the problem as:

**Problem Definition:** Given a set S of sentences $\{s_1,s_2,s_3,...,s_n\}$, predict a label $l_i$ for each sentence in S, where $l_i \in$ {suggestion, non suggestion}.
In this thesis, we aimed to further refine this problem definition by formally defining the scope of suggestion and non-suggestion classes.

Following three research questions form the guiding map of the research presented in this dissertation.

***RQ1: What are suggestions in the context of Suggestion Mining?***

So far only a small number of studies are conducted on suggestion mining, most of them focused on a specific application of suggestion mining. All these studies present a dictionary like definition of suggestion, aided with example sentences. At a granular level, we observe that such definitions vary across domains and use cases for suggestion mining. We therefore observe a lack of coherent and formal definition of the class *suggestion* in the previous works, unavailable annotation details, and inaccessible datasets. The process of defining suggestion and annotation guidelines also leads to investigation into the related questions, like:
*What is the unit of a suggestion?*
*What is the agreement between humans on perceiving a given text as a suggestion or non-suggestion?*
*Does the nature of suggestions vary across source domains?*
*What is the relation between suggestions and sentiments?*
Following contributions were made regarding the research question 1:

- Qualitative (linguistic) and quantitative analysis of suggestions across different domains.
- A method to formally define the scope of suggestion and non-suggestion classes, using certain parameters. Identification of *implicit* and *explicit* type of suggestions, where the thesis focuses on explicit suggestions.
- Preparation of benchmark quality human annotated datasets for different domains.

***RQ2: What are good features for supervised learning in suggestion mining?***

Most of the related works used linguistic patterns and rules for suggestion mining. Two of the works used statistical classifiers with hand picked features, many of which were dependent on domain specific properties and thus cannot not be used with a different domain. To the best of our knowledge, no investigation was made on automatic feature learning, and on the use of deep learning based classification methods for suggestion mining, which learn to represent data as a part of learning the model.
This research question also leads to the investigation into related questions, like:
*Which of the two, manually defined features or automatically learned features work better for suggestion mining?*
*What kind of features suit different training scenarios, domain specific, cross domain,*

*and open domain training.*
*Does the relative performance of different features depend on domains?*
Following contributions were made regarding the research question 2:

- Identification of new features for suggestion mining, and their evaluation for multiple domains, as well as domain independent training.
- Compilation of all features and rules from the existing works and a comparison of their performance with the proposed features.
- An in-depth comparison of manually identified features vs. automatically learned features. The learned features include pre-trained word vectors, as well as sentence vectors obtained by processing the pre-trained word vectors.
- Comparison of the performance of Support Vector Machine and Long Short Term Memory Network classifiers for suggestion mining.

***RQ3: How can distant supervision be employed in suggestion mining?***
The training datasets used for the investigation of previous research questions were small and highly unbalanced, in comparison to the benchmark datasets to which deep learning approaches are generally applied. Since the preparation of training datasets is an expensive process, we investigate into finding silver standard training datasets by looking for alternative sources of suggestion like sentences on the web. We also investigate into different methods to use such datasets for performance improvement of suggestion mining classifiers, specially in a domain independent setting. Therefore, this research question also covers questions like:
*What could be the possible sources for large silver standard training datasets ?*
*In what ways a silver standard dataset be used with the existing benchmark datasets ?*
*Is there a performance gain when a silver standard dataset is employed ?*
Following contributions were made under this research question:

- Identification of the wikiHow platform as a potential source of suggestion like sentences. Compilation of a silver standard training dataset using wikiHow and Wikipedia, which is an open domain dataset, and comprises of 1.3 million sentences.
- Evaluation of different methods to employ this silver standard dataset and improve the classification accuracy.
- Introduction of a method to learn vectors for part of speech tags, specifically developed for suggestion mining. These vectors perform comparably to the state of the art word vectors when employed with LSTM classifiers.

## 1.5 Applications

A key motivation behind the research in suggestion mining is the variety of applications it can be employed to, some of these are outlined below.

1. **Ideas for product improvement**: In addition to the online reviews and blogs, people are increasingly resorting to social networks like Twitter, Facebook etc. to instantly express their sentiments and opinions about the products and services they might have experienced in a recent time. Suggestions present among the opinions expressed on social media often convey ideas for improvements to a brand owner. For example, 'Dear Microsoft, release a new zune with your wp7 launch on the 11th. It would be smart'. These ideas are otherwise sought by the stakeholders through dedicated suggestion boxes or online forms.

2. **Customer to customer suggestions**: Reviews, blogs, and other opinionated text about commercial entities also carry tips, warnings, and advice for the future customers. Often future customers seek such advice through dedicated question answering forums, where they have to wait for a considerable amount of people to answer, with no answers at times. On the other hand, answers for these advice seeking queries may be present among unstructured text from a different source. For example, electronics reviews contain troubleshooting tips, or advice for accessories which go well with the reviewed product, eg., 'Btw , be sure to have your mp3 tags labeled correctly, as this will ensure a great organization', 'My only suggestion is to get a lens protector to help protect the shooting lens (the lens coating will wear out after so many clean wipes) and I'm getting the those (52 mm adapter and uv lens filter) at lensmateonline.com'. Another domain would be hotel reviews, which often contain room tips, i.e. which room should be preferred in a hotel. On the other hand, platforms like Tripadvisor which collect hotel and restaurant related opinions[1] request the reviewers to also fill up a separate room tips section, in addition to the hotel review (Figure 1.1). These kind of tips can be extracted using a suggestion mining system on the reviews of an entity collected from multiple sources.

3. **Suggestion summarisation:** Suggestion mining can also be employed for summarisation of dedicated forums or surveys used for collecting suggestions around a certain topic or entity. Suggestion providers often tend to provide context in their responses, which gets repetitive over a large number of responses relating to the same entity. A suggestion mining system can identify the exact sentence where a suggestion is conveyed in a larger piece of text, and thus help in summarising these responses. For example, in Figure 1.2, following sentences contain the actual suggestion, 'It would be

[1] https://www.tripadvisor.com

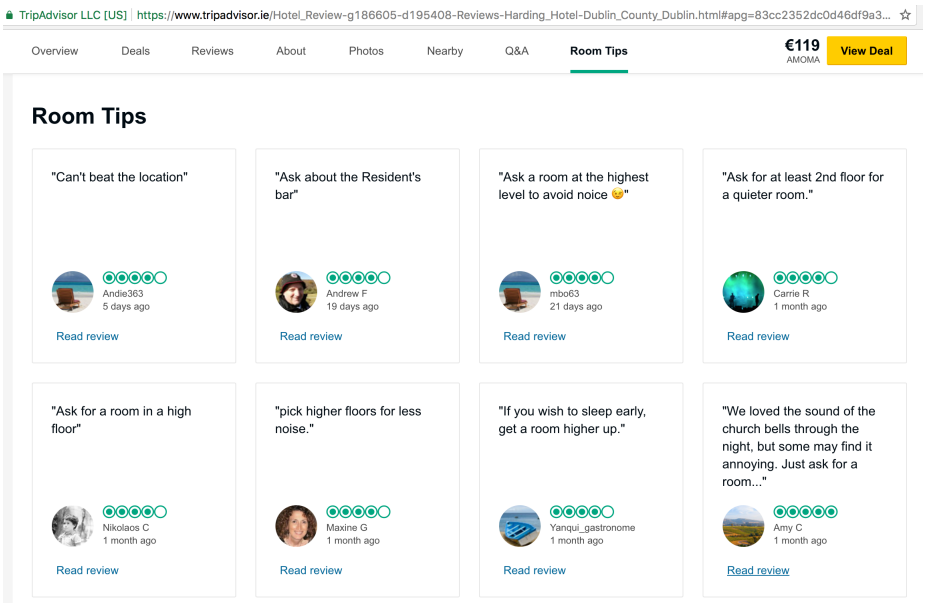



FIGURE 1.1: Room Tips on TripAdvisor

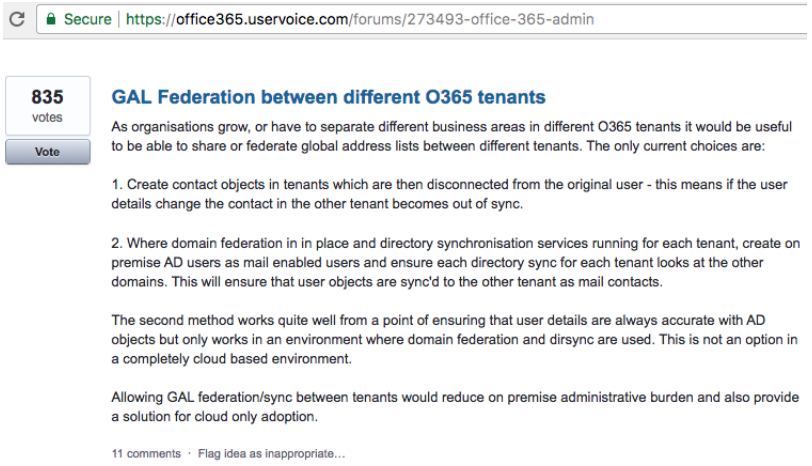



FIGURE 1.2: A post from the suggestion forum for Windows application developers

useful to be able to share or federate global address lists between different tenants.', and, 'Allowing GAL federation/sync between tenants would reduce on premise administrative burden and also provide a solution for cloud only adoption.', while rest of the text acts as a context.

4. **Improving sentiment detection:** A significant number of sentences in the benchmark sentiment analysis datasets are labeled as *neutral* or *objective*. Often sentiment analysis systems perform poorly in identifying sentiments of neutral and irrelevant sentences, or sentences with linguistic modalities like conditional and subjunctive mood [68]. Sentences which carry suggestions can often be neutral with respect to the target entity, and also tend to posses certain grammatical moods, along with the presence of positive and negative sentiment words (Table 1.1). Using suggestion label

as a feature may help improve the sentiment classification accuracies in such complex cases.

5. **Recommender systems**: Recommender systems predict user preference and recommend them entities. For example, Facebook recommends to connect with certain people with whom a user is not already connected on Facebook, and Youtube recommends videos to watch. There are different methods to make this prediction, which mainly utilise the content from a user profile and the entity profile. A common challenge for recommender systems is when a user profile or a candidate entity profile does not contain enough information. For example, a remotely situated or a new coffee shop may not necessarily have a web presence, or lacks the metadata typically required by recommender systems. However, recommendations for this shop may be found in a textual form among online reviews for the hotels near the shop, as well as travel blogs. For example, 'There are plenty of places to eat in the area around the hotel (try the baguette-sandwiches from Patisserie Paul in the tube-vaults)'. Suggestion mining systems will be capable of extracting such recommendations, which can then be utilized by recommender systems.

## 1.6 Thesis Outline

The remaining chapters of this thesis are organised as follows:

**Chapter 2:** This chapter broadly presents the background knowledge required to understand this thesis. This includes the terminology used in the thesis, where some terms are new while the others carry a specific sense in the present context. It also covers algorithmic details of existing machine learning classifiers employed in the experiments (SVM and LSTM), and the concepts of feature engineering and feature learning for text classification.

**Chapter 3:** This chapter covers the state of the art and the related work for suggestion mining. The state of the art corresponding to each research question is discussed. Other relevant studies in linguistics and psychology which studied and categorized suggestions in text are also described.

**Chapter 4:** This chapter investigates the first research question i.e. *What are suggestions in the context of suggestion mining?* It covers a detailed study of the properties of suggestions across different domains, annotation studies to prepare benchmark train and test datasets, and challenges observed during annotations. Most importantly, a formal

definition and scope for suggestions is defined in an attempt to reduce the context based ambiguity in the identification of a sentence as a suggestion. The datasets used in the rest of the thesis are detailed in this chapter.

**Chapter 5:** This chapter investigates the second research question i.e. *What are good features for machine learning based approaches to suggestion mining ?*. Features from the related work are aggregated, and additional features are also proposed based on the observations during the manual annotation. Experiments are performed using a support vector machine classifier, and evaluation are performed for both domain specific training, open domain training, and cross domain training of classifiers.

**Chapter 6:** This chapter continues the investigation of the second research question, but it focuses on automatic learning of features or representation learning. Pre-trained word embeddings (dense and continuous word vectors) are employed which were learned using state of the art methods on very large language corpora (like Wikipedia), in an unsupervised matter. SVM and LSTM classifiers are used with these pre-trained word embeddings. Since SVM classifiers are not designed to be used with word embeddings, two different methods of using word embeddings with SVM classifiers are evaluated.

**Chapter 7:** This chapter investigates into applying distant supervision to suggestion mining by means of a large silver standard training dataset. The chapter describes the development of this dataset at the first place, and then proposes different methods to use it. Distant supervision is combined with representation learning, in order to improve the classification accuracies for LSTM classifiers.

**Chapter 8:** Finally, we conclude the thesis with a summary of answers and observations obtained through our experiments and analysis, as well as the future directions for our research. Two openly available suggestion mining services developed by us are also described in this chapter.

Lastly, the appendices provide detailed results of the experiments, and are organised chapter-wise. Diagram 1.3 provides an overview of the flow and distribution of our contributions across the core chapters (Chapters 4-7).

## 1.7 Publications

Publications arising from the thesis are referenced in the respective chapters. Following is a list of publications were published during the course of this thesis, which cover the

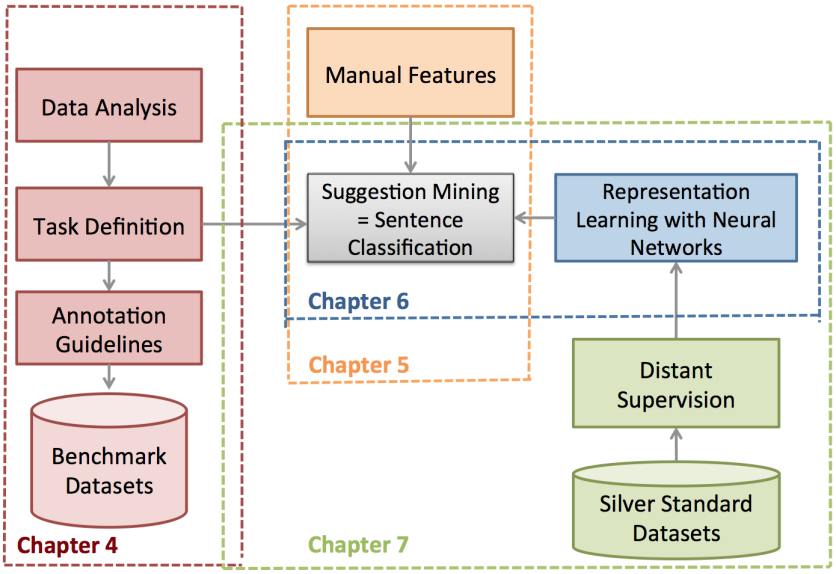



FIGURE 1.3: An overview of the core chapters

contributions mentioned above:

- **Open Domain Suggestion Mining: Problem Definition and Datasets**
  Sapna Negi, Maarten de Rijke, and Paul Buitelaar
  arXiv preprint arXiv:1806.02179, 2018

- **MixedEmotions: An Open-Source Toolbox for Multi-Modal Emotion Analysis**
  Paul Buitelaar, Ian D Wood, Sapna Negi, Mihael Arcan, John P McCrae, Andrejs Abele, Cecile Robin, Vladimir Andryushechkin, Housam Ziad, Hesam Sagha, Maximilian Schmitt, Bjorn W Schuller, J Fernando Sanchez, Carlos A Iglesias, Carlos Navarro, Andreas Giefer, Nicolaus Heise, Vincenzo Masucci, Francesco A Danza, Ciro Caterino, Pavel Smrz, Michal Hradis, Filip Povolny, Marek Klimes, Pavel Matejka, and Giovanni Tummarello
  IEEE Transactions on Multimedia, 2018

- **Inducing Distant Supervision in Suggestion Mining through Part-of-Speech Embeddings**
  Sapna Negi and Paul Buitelaar
  arXiv preprint arXiv:1709.07403, 2017

- **Suggestion Mining from Opinionated Text**
  Sapna Negi and Paul Buitelaar
  Sentiment Analysis in Social Networks, 2016

- **Suggestion Mining from Opinionated Text**
  Sapna Negi
  Association for Computational Linguistics, 119–125, 2016

- **A Study of Suggestions in Opinionated Texts and their Automatic Detection**
  Sapna Negi, Kartik Asooja, Shubham Mehrotra, and Paul Buitelaar
  Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics, 2016

- **Towards the extraction of customer-to-customer suggestions from reviews**
  Sapna Negi and Paul Buitelaar
  Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015

- **Curse or boon? presence of subjunctive mood in opinionated text**
  Sapna Negi and Paul Buitelaar
  Proceedings of the 11th international conference on computational semantics, 2015

- **INSIGHT galway: syntactic and lexical features for aspect based sentiment analysis**
  Sapna Negi and Paul Buitelaar
  Proceedings of the 8th International Workshop on Semantic Evaluation, 2014

# Chapter 2

# Background

In this chapter, we provide an overview of the existing terms, concepts, and algorithms which relate to this thesis. This includes the standard or thesis specific definition of key terms like domain and opinion mining. We also explain some technical concepts like inter-annotator agreement, feature engineering, feature learning. Finally we also review the classifiers used in this work, namely Support Vector Machines and Long Short Term Memory Networks.

## 2.1 Sentiment Analysis and Opinion Mining

The problem of Sentiment Analysis [52] was introduced in 2002, which dealt with the classification of a textual unit not with regard to its topic but with regard to the sentiment expressed in the text. Sentiment analysis is primarily used to refer sentiment classification of a text unit into positive, negative, objective or neutral categories. This gradually led to the growth of the broader area of Opinion Mining [35] which initially dealt with the application of sentiment classification to various domains and use cases. Till date, opinion mining is a well established and evolving research area in the field of natural language processing and computational linguistics. The benchmark datasets from sentiment analysis are also popularly used for evaluating machine learning algorithms for text classification.

In this thesis, we view suggestion mining akin to sentiment analysis. Suggestion mining can be applied on different domains and use cases related to opinionated texts. Suggestions can generally be categorized as opinions [2], and therefore suggestion mining can be placed under the umbrella of opinion mining.

In this work, when we talk of suggestions, we consider sentences as the unit of suggestion, Chapter 4 discusses this in detail, and also sheds some light on the relation between sentiments and suggestions. Consider the below hotel review:

> *Breakfast was typical hotel fare, not especially exciting, but fresh, varied and plentiful. Room-service fast and delicious, great selection of food.* **Ate dinner at Georgetown - highly recommended.** *There are plenty of places to eat in the area around the hotel.* **Try the baguette-sandwiches from Patisserie Paul in the tube-vaults.**

The text in bold are suggestions which appear among the review sentences. These suggestions do not convey any sentiment towards the reviewed hotel, but convey opinions about other entities, and recommendations related to one's stay in the hotel.

## 2.2 Linguistic Properties of Sentences

**Part of Speech tags**: The Part of Speech (POS) categories represent the role a word plays in a sentence. There are eight main parts of speech in English language: nouns, pronouns, adjectives, verbs, adverbs, prepositions, conjunctions and interjections. The most common POS tag category schemes are from Penn Treebank and Brown Corpus. For our work, we would be using the POS tag scheme developed for Penn Treebank, as shown in figure 2.1 [69].

**Dependencies**: In this thesis, we will later use Stanford typed dependency as one of the features for classification. Typed dependencies give a simple representation of the grammatical relations existing between the words in a sentence, considering that the syntactic structure of a sentence contains binary asymmetrical relations between the words [17]. Stanford typed dependencies contain approximately 50 grammatical relations. All dependencies explain binary grammatical relations described by a head and a dependent. Thus, a typed dependency form a triplet: name of the relation, head/governor and dependent. Figure 2.2 shows the Stanford typed dependencies for an example sentence 'Be sure to specify a room at the back of the hotel'.

## 2.3 Preparing Datasets: Inter-Annotator Agreement

It is very often that the linguistic/data annotation tasks are subjective, and can not be defined with all the necessary conditions for required mark ups. During a manual

| Tag | Description | Example | Tag | Description | Example |
|---|---|---|---|---|---|
| CC | coordin. conjunction | *and, but, or* | SYM | symbol | +,%, & |
| CD | cardinal number | *one, two* | TO | "to" | *to* |
| DT | determiner | *a, the* | UH | interjection | *ah, oops* |
| EX | existential 'there' | *there* | VB | verb base form | *eat* |
| FW | foreign word | *mea culpa* | VBD | verb past tense | *ate* |
| IN | preposition/sub-conj | *of, in, by* | VBG | verb gerund | *eating* |
| JJ | adjective | *yellow* | VBN | verb past participle | *eaten* |
| JJR | adj., comparative | *bigger* | VBP | verb non-3sg pres | *eat* |
| JJS | adj., superlative | *wildest* | VBZ | verb 3sg pres | *eats* |
| LS | list item marker | *1, 2, One* | WDT | wh-determiner | *which, that* |
| MD | modal | *can, should* | WP | wh-pronoun | *what, who* |
| NN | noun, sing. or mass | *llama* | WP$ | possessive wh- | *whose* |
| NNS | noun, plural | *llamas* | WRB | wh-adverb | *how, where* |
| NNP | proper noun, sing. | *IBM* | $ | dollar sign | *$* |
| NNPS | proper noun, plural | *Carolinas* | # | pound sign | *#* |
| PDT | predeterminer | *all, both* | " | left quote | ' or " |
| POS | possessive ending | *'s* | " | right quote | ' or " |
| PRP | personal pronoun | *I, you, he* | ( | left parenthesis | [, (, {, < |
| PRP$ | possessive pronoun | *your, one's* | ) | right parenthesis | ], ), }, > |
| RB | adverb | *quickly, never* | , | comma | , |
| RBR | adverb, comparative | *faster* | . | sentence-final punc | . ! ? |
| RBS | adverb, superlative | *fastest* | : | mid-sentence punc | : ; ... – - |
| RP | particle | *up, off* | | | |

FIGURE 2.1: Penn Treebank PoS tag scheme. Figure taken from [29].

```
cop(sure-2, Be-1)
root(ROOT-0, sure-2)
mark(specify-4, to-3)
xcomp(sure-2, specify-4
det(room-6, a-5)
dobj(specify-4, room-6)
case(back-9, at-7)
det(back-9, the-8)
nmod(specify-4, back-9)
case(hotel-12, of-10)
det(hotel-12, the-11)
nmod(back-9, hotel-12)
```

FIGURE 2.2: Typed dependencies for an example sentence.

annotation task, a person might need to decide whether a certain observation/example belongs to one or to another class. This decision can be subjective and contexual depending upon the perception, background and knowledge of the people involved in the annotation task. Therefore, we require some metric indicating the difficulty of the task and quality of the annotations. This metric is referred to as inter-annotator agreement which gives a measure of how well two (or more) annotators can make the same annotation decision for a certain data instance.

A high inter-annotator agreement confirms that task is not highly subjective, and unambiguous annotation guidelines can be designed for the same. While a low inter-annotator agreement would mean that the annotators found it difficult to agree on the classification of the given items in the data. It would be hard to utilize such dataset (with low inter-annotator agreement) for quantitative use-cases, for example, training a machine learning based classifier. For our annotation tasks, we use Cohen's kappa coefficient (k) statistic which measures inter-rater agreement for qualitative (categorical) items. It is a more robust measure than simple percent agreement/overlap calculation, as it incorporates the probability of the agreement occurring by chance. Cohen's kappa (k) can be defined as below:

$$k = \frac{p_o - p_e}{1 - p_e}$$

where $p_o$ is the relative observed agreement among annotators, and $p_e$ is the hypothetical probability of chance agreement. Value of k is 1 if the annotators are in complete agreement while k $\approx$ 0 signifies no agreement except what occurred by chance. A general rule of thumb is that any kappa value over 0.8 is considered very good for many tasks [1].

Consider two annotators A and B and two labels 1 and 2 they can assign to each data instance. a, b, c, and d are the counts of four possible cases of annotations as seen below:

| | | Annotator B | |
|---|---|---|---|
| | | **Label 1** | **Label 2** |
| Annotator A | **Label 1** | a | b |
| | **Label 2** | c | d |

The observed agreement will be,

$$p_o = \frac{a + d}{a + b + c + d}$$

The chance agreement for Label 1 will be,

$$p_{e,1} = \frac{a + b}{a + b + c + d} * \frac{a + c}{a + b + c + d}$$

The chance agreement for Label 0 will be,

$$p_{e,0} = \frac{c + d}{a + b + c + d} * \frac{b + d}{a + b + c + d}$$

Therefore, the total chance agreement will be,

$$p_e = p_{e,1} + p_{e,0}$$

## 2.4 Text Classification using Machine Learning

A number of natural language processing tasks have been framed as automatic text classification problems, where the categories vary according to the task under consideration. For example, sentiment analysis is the classification of text into sentiment polarities (positive, negative, or neutral), or language detection task where text is categorized into one of the language from a predetermined set of languages. In text classification, different units of text can be dealt with, like words, phrases, sentences, paragraphs, etc. In this thesis, suggestion mining has also been identified as a sentence classification task, where the classes are suggestion and non-suggestion.

Automatic text classification is mainly performed using two approaches, rules derived from linguistic observations, and supervised learning. The former involves manual observation of patterns in the text and associating the patterns with the classes, while the latter uses computational algorithms to do the same. In a supervised learning task, a function mapping certain features to a set of pre-defined classes is automatically learned from the data instances which are already labeled with their class. Features are discussed in more detail in the next section. In this thesis we particularly focus on two types of algorithms for supervised learning, Support Vector Machines (SVM), and Long short Term Memory (LSTM) networks. Both the algorithms are discussed in more detail towards the end of the chapter.

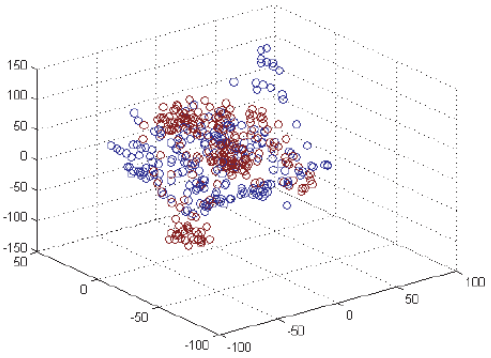



FIGURE 2.3: Data points belonging to the red and blue classes plotted in a three dimensional feature space.

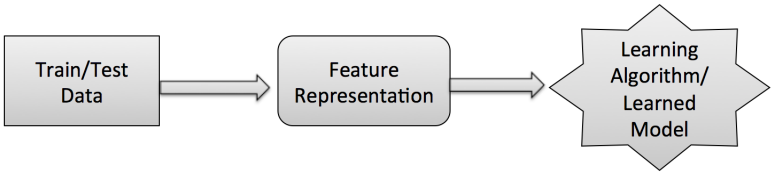



FIGURE 2.4: Steps involved in text classification or using a learned classification model

### 2.4.1 Features

A crucial step for statistical classification is to represent data in the form of numerical, and informative features which capture the discrimination between the classes. A feature quantifies an attribute or property observed in a data instance. Most of the supervised learning algorithms assume data instances to be a point in a high dimensional space, where each dimension corresponds to a single feature (see Figure 2.3).

Thus, a feature vector is an n-dimensional vector of features that represents a data instance. Ideal features are the ones which are useful in distinguishing the instances of one class from the others. Sometimes, feature extraction or feature selection step is used to further reduce the feature space. Feature extraction takes the initial set of features (measured from the data itself), and derive a complex feature space intended to be more informative by using general dimensionality reduction techniques. On the contrary, in feature selection a subset of original features is selected which best suits for the required task. A major part of the application of supervised learning algorithms deals with feature engineering which uses domain knowledge of the problem and data to come up with good features. Until the recent upsurge of representation learning primarily by deep neural networks, features were mostly manually identified by closely observing multiple data instances and using domain knowledge, and then applying feature extraction or feature selection. Text classification features may belong to all levels of a language, i.e. morphological, lexical, syntactic, semantic, and pragmatics. For example, a manually observed feature for suggestion mining is the presence of imperative mood in a given sentence.

**Manually determined features:** The accuracy of the learned function depends strongly on how the input data is represented. Typically, the data instance is transformed into a feature vector, which contains a number of features that are informative and discriminative. The number of features should not be too large, because of the curse of dimensionality; but should contain enough information to accurately predict the output.

In the case of text classification, feature space can easily become very high dimensional because of the usage of training data vocabulary and n-grams as features. A number of dimensionality reduction techniques can be employed on top of this feature space, in order to reduce the size of feature vector. In the past, a large amount of manual effort had gone into designing problem specific features to allow machines learn. Coming up with features is labour intensive and generally require expert knowledge, for instance linguistics knowledge in the case of text classification problems. The default features for

text classification are n-grams. Further improvement in the performance can be achieved by designing additional features which are suited to a specific problem. For example, depending on the task, some words can be be grouped into certain syntactic or semantic categories and those categories can prove better features than using the individual words.

One major issue about features from limited labeled training data is that only few features might be observed in the test data. In the recent years, there is an upsurge in studying methods to automatically learn feature representation or feature vectors from large amount of readily available unlabeled text. Some of the proposed feature learning methods based on neural networks have beaten the state of the art hand crafted features for many popular NLP tasks like machine translation.

## 2.5 Classification Algorithms

### 2.5.1 Support Vector Machines

A Support Vector Machine (SVM) is a discriminative classifier that is based on the concept of decision boundaries being defined by planes which are then known as decision planes. A decision plane creates a separation between a set of objects that belong to different classes. It requires the learning algorithm to find an optimal decision plane which can classify new examples. Each data item is plotted in a multidimensional space where the feature values are mapped to the values of the corresponding coordinates. SVM can support both regression and classification, and can handle multiple continuous and categorical variables. For a linearly separable dataset, there can be a number of possible hyperplane decision separators, as shown in figure 2.5.

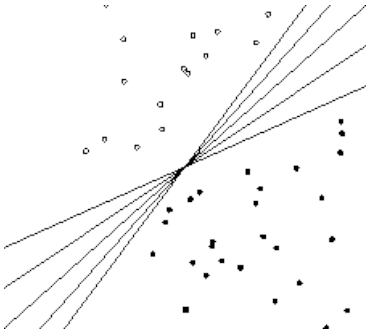


FIGURE 2.5: Infinite hyperplanes separating two linearly separable classes. Image taken from [38])

SVM employs an iterative training algorithm to minimize the error function for optimization. It is intuitive to assume that a decision hyperplane in the middle of the void between data points of both the classes would server classification purpose better than the one closer to data points. Thus, the SVM defines the optimization criterion to be looking for a decision boundary that is maximally far away from any data point. In contrast, perceptron algorithm finds just any hyperplane separator, and algorithms like Naive Bayes, search for a better decision boundary following some criterion [38].

Margin of the classifier is defined as the distance from the decisive hyperplane to the closest data point. As SVM tries to maximize this margin, it is categorized as a maximum margin classifier. This essentially implies that the SVM decision function is dependent only on a small number of data points of the data required to define the position of the hyperplane separator. Considering a vectorial space, these key data points are referred to as the support vectors, as shown in figure 2.6. SVM follows a convex optimization function to determine the model parameters, and so any local solution is also a global optimum.

SVM is effective in high dimensional spaces including the cases where number of dimensions is greater than the number of samples. As it uses only a subset of data points (support vectors) in the decision boundary function, it is quite memory efficient. The real data can be a mess, and not necessarily linearly separable; SVM employs the kernel trick to convert a low dimensional input feature space and transform it to a higher dimensional space, where the data maybe linearly separable. Kernels are mostly useful in non-linear separation problem, and makes SVM quite versatile. However, if the number of features is much greater than the number of training data points, it is important to choose kernel functions and regularization properly to avoid over-fitting.

The two general types of classification SVMs are as below:

SVM Type 1 (C-SVM):
Optimization function: $\frac{1}{2}w^T w + C\sum_{i=1}^{n}\xi_i$

subject to the constraints: $y_i(w^T\phi(x_i)+b) \geq 1-\xi_i$ and $\xi_i \geq 0, i = 1, ...., N$

SVM Type 2 (nu-SVM):
Optimization function: $\frac{1}{2}w^t w - v\rho + \frac{1}{N}\sum_{i=1}^{n}\xi_i$
subject to the constraints: $y_i(w^T\phi(x_i)+b) \geq \rho-\xi_i$, $\xi_i \geq 0$, $i = 1, ...., N$ and $\rho \geq 0$

In the above equations, C is the capacity constant, w is the vector of coefficients, b is a constant, and $\xi_i$ represents parameters for handling non-separable data. The index

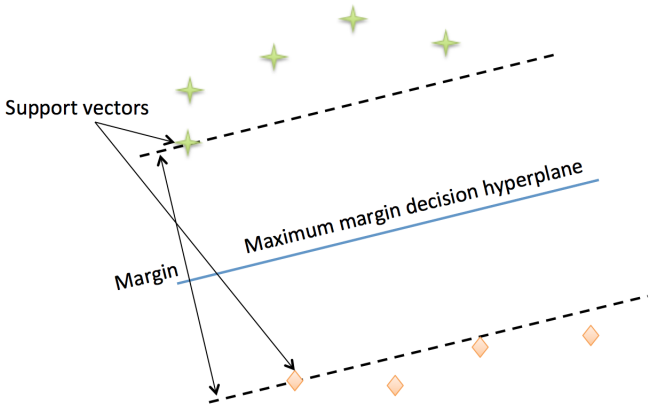



FIGURE 2.6: Support vectors positioning the maximum margin hyperplane.

i labels the N training cases. $y\epsilon \pm 1$ represents the class labels and $x_i$ represents the independent variables. The kernel is used to transform data from the input (independent) to the feature space. It should be noted that the larger the C, the more the error is penalized. Essentially, C controls the trade off between smooth decision boundary and classifying the training points correctly. Thus, C should be chosen with care to avoid over fitting.

The kernel function represents a dot product of input data points mapped into the higher dimensional feature space by transformation. The popular kernel functions are as follows:

1. Linear kernel: $K(X_i, X_j) = X_i \cdot X_j$
2. Polynomial kernel: $K(X_i, X_j) = (\gamma X_i \cdot X_j + C)^d$
3. Radial basis function (RBF) kernel: $K(X_i, X_j) = exp(-\gamma|X_i - X_j|^2)$
4. Sigmoid kernel: $K(X_i, X_j) = tanh(\gamma X_i \cdot X_j + C)$

Gamma or $\gamma$ is an adjustable kernel coefficient for RBF, polynomial and sigmoid kernels. A higher value of gamma will try to exact fit the training data set, and thus can induce generalization error and cause over-fitting problem.

**SVMs for Suggestion Mining**

Given that the task at hand is the binary classification of sentences, in the later chapters we perform experiments with Support Vector Machines (SVM) classifier with a Radial Basis Kernel. SVMs have been widely accepted as a good choice of algorithm for text and binary classification [13, 28, 31], since it often performs comparable or better than its counterparts. It is also known to require a relatively small amount of parameter tuning for parameters like number of features [28]. In addition, all the related works on statistical suggestion mining used Support Vector Machines [18, 43, 72], excluding the ones which approached suggestion mining as a sequence classification task.

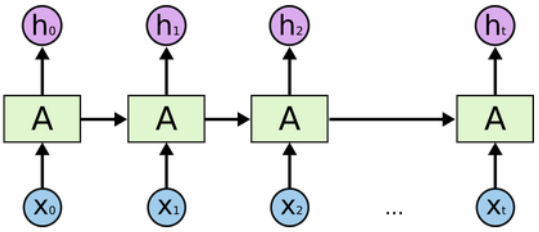



FIGURE 2.7: A generic RNN architecture.[1]

The RBF kernel defines a function space much larger than the linear or polynomial kernel, which allows the complexity of SVM classification models with RBF kernels to grow with the data. While polynomial kernels also offer a much larger function space than linear kernels, a number of comparative studies prove RBF kernels equal or at-par with the polynomial kernels [13, 28]. In the further chapters, we also employ neural networks for suggestion mining which themselves are able to learn complex non-linear functions, and therefore RBF kernels seemed to be a plausible choice to compare the performance of SVMs and Neural Nets.

### 2.5.2 Long Short Term Memory Networks

Long Short Term Memory Networks (LSTMs) refer to a type of artificial neural network modeled to utilize patterns occurring in data sequences to solve required tasks. The generic term for such networks is Recurrent Neural Networks (RNNs). In contrast to feedforward networks, RNNs have a feedback loop connecting their past decisions, thus ingesting their earlier outputs as input. LSTMs are arguably the most commonly used type of RNNs. LSTMs have consistently been shown highly successful at many NLP tasks. They make use of sequential information allowing operations over sequences of vectors. The sequences can exist in the input, output or both. Traditional neural networks consider all inputs independent of each other. On the other hand, RNNs do consider a sequential dependency between them. It can also be considered as having a "memory" recording a summary of the previous events in the sequence. This allows the information to persist throughout the network, and making every sequence output to be conditioned on the previous computations as well. However, naive RNNs are not able to handle long-term dependencies in practise, due to the exploding and vanishing gradient problem [25]. In comparison to vanilla RNNs, LSTMs are much better at capturing long-term sequential dependencies. We would briefly discuss initial RNN architecture, and then see how LSTMs extend them.

Figure 2.7 shows a generic RNN architecture. Here, *A* shows the recurrent hidden

[1]Image taken from http://colah.github.io/posts/2015-08-Understanding-LSTMs/
[2]Image taken from http://colah.github.io/posts/2015-08-Understanding-LSTMs/

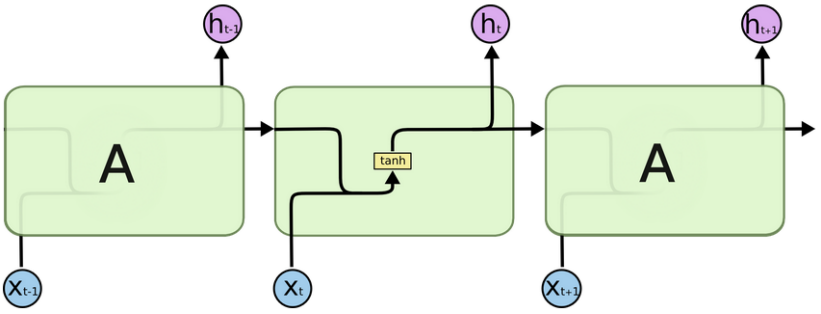



FIGURE 2.8: NN module contains a single layer in RNN.[2]

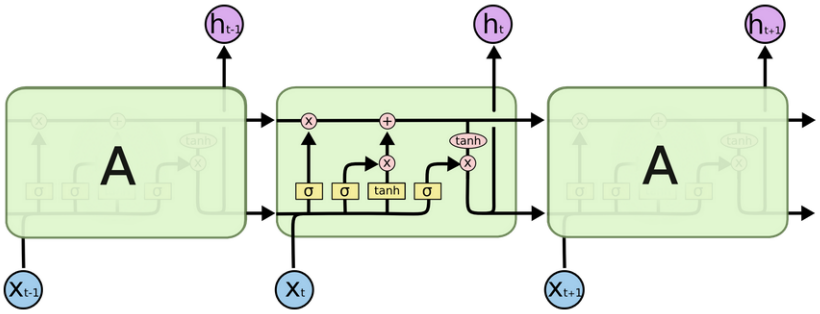



FIGURE 2.9: NN module contains four interacting layers in LSTM.[3]

neural network module repeating over the full input sequence, $X_i$ represents a vector from the input sequence, and $h_i$ is the corresponding output. Following the diagram, the current output value can be written as:

$$h_t = f(h_{t-1}, X_t)$$

It shows that the current state output $h_t$ is dependent upon the previous state output $h_{t-1}$ and current state input $X_t$. The function f is a non-linear activation function such as tanh or sigmoid as shown in the figure 2.8.

Due to the inherent sequential nature, RNNs require an extension of back propagation considering the temporal aspect called Back propagation through time (BPTT). In the back-propagation phase of classic RNNs, the gradient interactions between words that are several steps apart in a sequence can get lost/weaken due to vanishing or exploding gradient problem [54]. However, this fundamentally is not a problem with RNNs or neural networks as such. It is actually caused by the gradient based learning methods, which learn how to tune the parameter values in very small steps optimizing the network's output. In the case of vanishing gradient, a change in a parameter's value is unable to make any notable change in the network's output, so it becomes harder to

[3]Image taken from http://colah.github.io/posts/2015-08-Understanding-LSTMs/

optimize the parameter value effectively. This is highly dependent upon the choice of activation function used for learning. General activation functions like tanh or sigmoid squash the input values to a very small value, thus this effect is much stronger in such functions. However, recently introduced activation functions like Rectified linear units (ReLU) are able to handle these problems.

These issues led to the development of LSTM model, which introduces a memory cell, a new structure to handle the past events in the sequence. The NN module of LSTM memory cell contains four main elements: an input gate, a neuron with a self-recurrent connection, a forget gate and an output gate, as shown in the figure 2.9. Comparing figures 2.8 and 2.9, we can notice the differences between the neural network (NN) modules of both RNN and LSTM. The recurrent connection in the LSTM NN module uses a weight value of 1.0 to ensure that the state of a memory cell can remain constant moving from one timestep to another in the sequence. The forget gate allows the memory cell to remember or forget its earlier state if required. The input gate controls the altering of the memory cell state following the current state incoming signal. On the contrary, the output gate controls the effect of the memory cell state on other neurons.

**Determining Hyperparameter Values**

Typical hyper-parameters for LSTM networks are same as the general neural nets, like, number of layers, number of neurons per layer, activation function, dropout regularization, number of epochs, batch size, optimization algorithm, and learning rate. Because of the ability of deep neural nets to learn complicated non-linear functions, it may be easy to achieve high accuracy during the train time by parameter tuning as per train time model performance. However, such complex functions are highly prone to over-fitting and may not generalize well to the test set. Therefore, the choice of hyper-parameters should consider the likelihood of over-fitting. *Vanishing gradient* problem as explained in the previous section is another issue commonly faced by recurrent neural nets, where the gradient becomes very small by the time backpropogation reaches to the initial layers of the network, which leads to model stopping learning. Choosing an appropriate activation function can address this problem. Hyper-parameter optimisation is relatively less computationally expensive for SVMs as compared to recurrent neural networks, due to lesser number of hyperparameters as well as the lower number of computations required for each run. Some studies have evaluated hyper-parameter values on benchmark problems and datasets, and have recommended a set of values to experiment with [9].

**LSTMs for Suggestion Mining**

In general, the choice of depth and complexity of a neural network architecture should depend on the size of data used for training [9], smaller the dataset, less complex the network. In this thesis, we experimented with the recommended hyperparameter values

by Bengio [9], which falls in the category of *hand tuning or trial and error* approach for hyperparameter selection. The loss function on which we intend to do our optimization is chosen as categorical cross-entropy because the problem at hand is binary classification.

## 2.6 Word Representation Learning for Text Classification

Since the advent of internet, there has been an increasing availability of online text from blogs, social networking sites, online news, and encyclopedic web pages. Past few years of research on text classification has focused on making effective use of this unlabeled text data. This open domain unlabeled data may not be directly useful for specific type of text classification, but it contains generic information and knowledge about the world or a a particular domain in the cases where a large amount of data is available about a certain topic or domain.

As mentioned earlier, words are important features for text classification tasks. In the traditional text classification methods, each word generally corresponded to one dimension of the feature vector. However, classification algorithms based on neural network classifiers (e.g. Recurrent Neural Networks) accept an n-dimensional vector for each word, and therefore the feature vector for a text instance becomes a *tensor*. Representing each word as a dense vector gives the ability to provide a semantic representation of words, which can be used to compare words with each other. This is in contrast to the one hot vectorial or sparse representation of the word, where the feature value would either be the count of the word in a given text or simply a binary value representing its presence or absence.

Word vectors can be used directly (for computing similarities between words), as well as for computing text representations for higher level NLP tasks like text classification, part of speech tagging, named entity recognition, sentiment analysis, etc. There is an ongoing trend of learning low dimensional word vectors (50-200 dimensions) or distributed representations from very large unlabeled language corpora, using neural network architectures. These pre-computed vectors are popularly referred to as *word embeddings*, and can be used in a number of NLP tasks. Although the term word embeddings have become recently popular, the concept of representing words in a low dimensional space is not new and have been previously referred to distributional semantic representations like latent semantic analysis (LSA) and latent Dirichlet allocation (LDA). Distributional semantic representations are based on the distributional hypothesis for words i.e. words that are used and occur in the same contexts tend to represent similar meanings. In the recent papers on word embeddings or distributed representations, it has been shown

that both distributed and distributional representations model the word co-occurrences. However, with distributed representations, we get the flexibility of playing with the optimization function to model the word representation according to a specific task.

In this thesis, we are mainly concerned with learning word embeddings and using them for suggestion mining. In the current research scenario, a number of studies have been performed on evaluating word vectors which include modern word embeddings and traditional word vectors obtained through distributional semantics. Two state of the art models, GloVE and Word2Vec tend to significantly outperform other models, and are themselves very close in their performances across a number of tasks.

**Word2Vec:** Mikolov et al. [41, 42] proposed simpler and less expensive architectures for learning word embeddings using neural networks, as compared to the approaches which were designed to learn language models and produce word embeddings as side products and were much more computationally expensive. The Word2Vec approach comprises of two models with slight variations. The general notation for these models include a training corpus T with words, $w_1, w_2, w_3, ..., w_T$, which constitute a vocabulary V with size $|V|$. The input embedding for a word w is $V_w$ with d as the number of dimensions. Each model optimizes an objective function $J_q$ with regard to the model parameters q and the model outputs a score $f_q(x)$ for an input x.
The first model referred to as the continuous bag of words (CBOW) model, uses n words before and n words after a target word $w_t$ to predict $w_t$. Equation (2.1) shows the objective function.

$$J = \frac{1}{T} \sum_{t=1,T} logp(w_t | w_{t-n}, ..., w_{t-1}, w_{t+1}, ..., w_{t+n}) \tag{2.1}$$

The second model is known as the skip-gram model, where given a word, task is to predict the surrounding words. The objective function for this model sums the log probabilities of the n words to the left and n words to the right of the word $w_t$ (Equation (2.2)).

$$J = \frac{1}{T} \sum_{-n \leq j \leq n, \neq 0} logp(w_t + j | w_t) \tag{2.2}$$

**Global Vectors for Word Representation (GLoVE):** This method for obtaining word embeddings explicitly performs the operations inherently performed by Word2Vec. Word2Vec ends up capturing co-occurrences of the words in the form of word embeddings. However, GloVe focuses on this co-occurrences of two words as its optimization function directly, and encodes the ratio of the co-occurrence probabilities of two words.

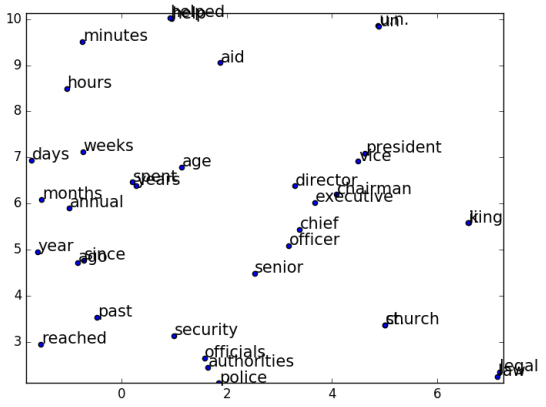



FIGURE 2.10: A sample of two dimensional representation of 200 dimensional word vectors learned from unlabeled data using the GLoVE method.

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T w_j \tilde{w}_j + b_i + \tilde{b_j} - logX_{ij})^2 \tag{2.3}$$

where, $w_i$ and $b_i$ are respectively the word vector and bias of word i. $\tilde{w}_j$ and $b_j$ are the context word vector and bias of word j, and $X_{ij}$ is the number of times word i occurs in the context of word j, and f is a weighting function that assigns relatively lower weight to rare and frequent co-occurrences. (2.3)

*Learning task specific embeddings*: When embeddings are employed in different tasks, they may lack in domain specific knowledge and/or task specific knowledge. At times, the application data for which pre-trained embeddings are to be used is different from the data used to learn those embeddings at the first place. In such cases, if an equally large amount of unlabeled documents are available from the target domain, embeddings can be learned from scratch or fine-tuned using one of the popular methods, and then can be further used for the task at hand. If such amount of target domain data is not available, a common approach is to adapt the pre-trained embeddings on the new domain while retaining the previously learned knowledge. Several methods have been used to perform this kind of adaptation. To enrich the embeddings with task specific knowledge, one method is to augment the word embeddings with additional dimensions and explicitly provide task specific information in those dimensions.

# Chapter 3

# Related Work

In this chapter, we summarize and review previous works which studied suggestion mining. We also include the studies which do not refer to their work as *suggestion mining*, but are closely associated with our research questions. Sections 3.1, 3.2, 3.3 correspond to the three research questions of this thesis. We also briefly highlight how our contributions are alternative or complementary to the existing works. At the end of the chapter, table 3.3 provides a summary of the state of the art for suggestion mining.

## 3.1 Defining Suggestions

This section describes the related work for our first research question, i.e. *How do we define suggestions in the context of suggestion mining.* Defining what kind of text is covered under the term *suggestions* is as crucial as the algorithmic part of the problem, since it brings consistency in the dataset preparation, and defines the scope of the problem for robust evaluation and comparison. We observe that suggestions have been defined in the literature in two kinds of research contexts. Suggestions are a type of information in the context of suggestion mining, which draws the attention towards the semantics of the text. On the other hand, in linguistic studies, suggestions have been mostly referred to as a type of speech act or an illocutionary act, under which syntax and pragmatics of the text are the centre of attention.

### 3.1.1 Defining Suggestions for Suggestion Mining

To the best of our knowledge, not all the related works in suggestion mining explicitly defined what they referred to as *suggestions*. Those who attempted to define suggestions, did so by providing a gloss like definition. The related works are sporadic in the sense

that they do not follow a coherent problem definition with regard to what kind of text is considered as a suggestion.

Wicaksono and Myaeng performed the extraction of what they refer to as the *advice revealing sentences* from weblogs and online forums [72–74]. They describe advice as: 'a sentence made by person, usually as a suggestion or a guide to action and/or conduct relayed in a particular context'. The authors also provide some linguistic observations in the sentences which they manually annotated with *advice* and *non-advice* labels [73]. They report the following observations:

- Specific vocabulary aids, like, *make sure you, be careful with, be sure to, be aware, you have to, i suggest, i strongly recommend, advice,* etc.
- Use of modal verbs, *could, might, must, shall, should, will, and would.*
- Presence of imperative mood, marked by unconjugated form of verb without any attached subject.
- Presence of sentiment words, like, *It is nice idea to bring medicine wherever you go.*

These observations are in-line with what have been already explored as the linguistic markers for suggestions, as described in the next section. Wicaksono and Myaeng did not provide the formal annotation guidelines for their advice dataset [74]. However, the kappa statistics for inter-annotator agreement was calculated as 0.76, and only those sentences were included in the datasets on which the two annotators agreed. Their dataset has been made available for research.

Ramanand et al. [60] investigated the detection of suggestions for product improvements from reviews, and provided a domain specific definition: 'sentences where the commenter wishes for a change in an existing product or service'. They also provide some example sentences which should or should not be considered as suggestions for improvements (see Table 3.1). The annotation agreement was not reported and the dataset is not made available for research.

Viswanathan et al. [70] provided a broad definition of suggestions: 'a suggestion made by a user indicates what one should know about something being discussed before an informed decision can be made by the reader'. They also focused on the domain of online reviews, and provided examples of the kind of information these suggestions should convey. Annotation details including the inter annotator agreement were not provided.

Dong et al. [18] performed suggestion detection on tweets about the Microsoft Windows 7 phone. They do not define suggestions, but mention that the objective of collecting suggestions is to improve and enrich the quality and functionality of products, services, and organizations. The dataset prepared by them is publicly available, however the formal annotation details are not available. They did not calculate the inter-annotator agreement, and retained only those labeled tweets in the dataset where all the annotators agreed on the label.

Moghaddam [43] performed suggestion mining at the document level. This work aimed to identify suggestion containing reviews in a large number of reviews about the eBay App. They define such reviews as, 'Feedback that explicitly suggests/requests company to add/change/improve/stop specific aspects. In other words, the customer proactively proposes a product change/improvement to the company'. They labeled suggestions at review level as well as sentence level. They report that 5 human annotators were employed, and a 96% agreement was observed in their annotations, where only agreed upon instances were retained in the dataset. However the dataset is proprietary to the company *eBay*, and is therefore unavailable for open research.

Table 3.3 provides a summary of related works and the available datasets. The above mentioned previous works on suggestion mining did not investigate the definition and scope of suggestions as a research question. They mostly provided a loose definition of suggestions, and examples of the type of suggestions they wish to mine from the domain being addressed. The details on the annotation process of suggestions, like the annotation guidelines, analysis of disagreements, and in some of the cases inter annotator agreement, largely remain unaddressed or unreported.
However, these works did remove the disagreed instances from the train and test datasets if manual labeling was performed using the broad definitions as guidelines. Although, this may be one way to compensate for a missing annotation study, models trained on such unambiguous datasets may result in a lowered performance on real life data, as compared to the unambiguous test datasets. In Chapter 4, we perform a study of different perceptions of the term *suggestion*, formalise the definition of suggestions, and propose a *two pass* annotation method in order to create benchmark datasets.

### 3.1.2 Defining Suggestions for Linguistic Studies

The previous works on suggestion mining provided their own preliminary definitions and characteristics of suggestions, where there is a scope to build upon the existing linguistic studies dedicated towards characterizing and defining suggestions. Discourse analysis, and speech acts are two such areas of study.

Asher et al. [2] perform a qualitative and quantitative study of the kinds of opinion expressions found in discourse. They study explicit and lexical expressions of affective content, and how these expressions are related to each other within a discourse in written texts. They provided a categorisation of expressions, out of which one was *Advice*, and suggestions were listed as a type of advice expressions. They also listed different kinds of opinion verbs which fall under each category (see figure 3.1).

| CATEGORIES | GROUPS | EXAMPLES |
|---|---|---|
| REPORTING | a) Inform | *inform, notify, explain* |
| | b) Assert | *assert, claim, insist* |
| | c) Tell | *say, announce, report* |
| | d) Remark | *comment, observe, remark* |
| JUDGMENT | e) Think | *think, reckon, consider* |
| | f) Guess | *presume, suspect, wonder* |
| | g) Blame | *blame, criticize, condemn* |
| | h) Praise | *praise, agree, approve* |
| | i) Appreciation | *good, shameful, brilliant* |
| ADVISE | j) Recommend | *advise, argue for* |
| | k) Suggest | *suggest, propose* |
| | l) Hope | *wish, hope* |
| SENTIMENT | m) Anger/CalmDown | *irritation, anger* |
| | n) Astonishment | *astound, daze* |
| | o) Love, Fascinate | *fascinate, captivate* |
| | p) Hate, Disappoint | *demoralize, disgust* |
| | q) Fear | *fear, frighten, alarm* |
| | r) Offense | *hurt, chock* |
| | s) Sadness/Joy | *happy, sad* |
| | t) Bore/Entertain | *bore, distraction* |

FIGURE 3.1: Opinion categories by Asher et al., 2009. Table taken from [2]
.

Studies in the area of pragmatics and psychology have identified suggestions as a type of speech act. Searle [64] mentions: *In a typical speech situation involving a speaker, a hearer, and an utterance by the speaker, there are many kinds of acts associated with the speaker's utterance. He will also have performed acts within the class which includes making statements, asking questions, issuing commands, giving reports, greeting, and warning.* The studies on speech acts relate to the pragmatics of a language, i.e. for what purpose is a sentence being used by the speaker. All the theoretical linguistics works on speech acts focused on defining the typologies or improving the existing typologies, in order to classify commonly found speech acts.
Following is a detailed classification provided by Bach and Harnish [5], which was built upon the previously published works by Austin[3] and Searle[65]:

- **Constatives:** affirming, alleging, announcing, answering, attributing, claiming, classifying, concurring, confirming, conjecturing, denying, disagreeing, disclosing, disputing, identifying, informing, insisting, predicting, ranking, reporting, stating, stipulating.
- **Directives:** advising, admonishing, asking, begging, dismissing, excusing, forbidding, instructing, ordering, permitting, requesting, requiring, suggesting, urging, warning.
- **Commissives:** agreeing, betting, guaranteeing, inviting, offering, promising, swearing, volunteering.
- **Acknowledgments:** apologizing, condoling, congratulating, greeting, thanking, accepting (acknowledging an acknowledgment).

These categories and subcategories are very fine grained and some of them are separated by minor distinctions. For example, advising, instructing, suggesting, warning. Linguistic experts would be required to label sentences as per the characteristics defined for these classes of speech acts.
We argue that the speech act theories cannot be followed in their original state for defining suggestions in suggestion mining due to the following reasons:

1. Speech act theories study well formed and standard text, and assume that the sentences would have lexical clues of the corresponding speech acts. For example, in the case of suggestions, 'I suggest ...'. However, this is not the case with the user generated text on the internet.
2. The suggestions in suggestion mining may comprise of not just the *suggest* speech act, but also the acts like advicing, requesting, instructing etc. What speech act theories refer to as *suggestion* is a subset of what suggestion mining can refer to as *suggestions*.
3. Unlike speech act theories, suggestion mining views suggestions from a data mining perspective. Therefore, what is said, and in what context are important to define suggestions. For example in the suggestion, "I suggest you to go for a room at the back of the hotel", what is suggested is the required information and the act of suggestion serves as an important feature to identify it. But not all the sentences which may possess a suggestion or advice speech act are suggestions for suggestion mining, for example, "I only suggest to go for this hotel if you have plenty of money to waste".

If the categories defined in these studies are mapped to the sentences considered as suggestions in suggestion mining, lexical and syntactic features for machine learning approaches can be determined. However, the linguistic studies and suggestion mining studies have remained mutually exclusive in the past. We make an effort to bridge this gap between the two; we employ the findings of linguistic studies to define the scope of suggestions in suggestion mining, as well as provide some qualitative and quantitative analysis of datasets which can be employed by the future linguistic studies.

The state of the art in suggestion mining mainly focused on the research question 2, which we discuss in the following section.

## 3.2 Feature Engineering

This section describes the related work for our second research question, i.e. *What are useful features for detecting suggestions?* As mentioned in the previous section, some experiments have already been performed to automatically detect suggestions in text, where each of them formulated and evaluated their method for a single domain. These works mostly focused on the domains of product and service reviews, with the exceptions of weblogs, and twitter. All of these works defined suggestion mining as a binary text classification problem, where the positive and negative classes represent *suggestion* and *non-suggestion* respectively. Most of the works considered sentence as unit of suggestion. The employed methods fall in two categories, manually identified rules based on observed linguistic patterns and vocabulary, and supervised machine learning with manually determined feature types.

### 3.2.1 Rule Based Classification

In rule based classification, manually identified linguistic patterns can be considered as features given that the classifier algorithm is to simply check their presence or absence in the text.
Ramanand et al. [60] performed the extraction of *wishes to see improvement in the products and services*, and referred to them as *Suggestion Wishes*. They evaluated a set of hand crafted rules (see Table 3.1) on manually labeled datasets comprising of review or feedback about different entities, like iPod (˜21100 sentences), banking services (˜15400 sentences), financial services (˜11000 sentences), etc. On average less than 1% of sentences were labeled as suggestions. Precision varied between 83% and 57% for different datasets, and recall varied between 60% and 39%. Some of their rules can not be replicated due to the unavailability of the hand crafted lexicons.

| **Example of target suggestions** |
|---|
| In Scope: Request for new features, feature may not necessarily be mentioned<br>• I'd love for the iPod shuffle to also mirror as a pedometer.<br>• It would be much better if they had more ATMs in my area.<br>• I wish they'd do this.<br>• My wish list would be as follows<br><br>Out of Scope: Negative opinion, improvement could be inferred<br>• My only gripe is the small size of the camera body.<br>• The rubber flap that covers the usb port seems flimsy. |
| **Linguistic patterns** |
| Rules based on modal verbs:<br>• ⟨modal verb⟩ ⟨auxiliary verb⟩ {window of size 3} ⟨positive opinion word⟩<br>• ⟨modal verb⟩ {window of size 3} ⟨preference verb⟩<br>• "should be able" or "should come with" or "could come with"<br><br>The *needs to* rules:<br>• Interrogative sentences or those with a negation word to the left of "need to" are not wishes.<br>• If the product attribute is present (usually as the subject), the sentence is a wish.<br>• If the subject of the sentence is one of "this, that, these", the sentence is likely to be a wish. When the subject is one of "I, you, one", the sentence is not a wish.<br><br>Other rules:<br>• "I wish": along with filters such as the subject ("they, you, product") etc. can be matched as wishes.<br>• "hopefully" or "I hope"<br>• "should be able to" or "should come with" |

TABLE 3.1: Examples of suggestion wishes and patterns from Ramanand et al. [60]

Viswanathan et al. [70] performed suggestion mining on customer reviews for mobile phones, aiming to extract suggestions for improvement. They built a multi-component system where one of the components comprised of linguistic patterns with explicitly identified keywords. In their rules, they refer to a product or a brand or a company as *Product*, features of these entities as *Feature*, and *Attribute* refers to the characteristic of the feature for which a suggestion is made. Examples of these patterns are, ⟨ *'suggest' (Product) 'for' (Features)* ⟩, ⟨ *'I wish' (Product) 'had/contain/include' (Feature)* ⟩, ⟨ *(Product) 'should have' (Feature)* ⟩, etc. The other components comprised of a linguistic pre-processor for normalizing the text, a syntactic engine (statistical parser), a semantic processor (named entity detection), and a post processor to convert the elements extracted from suggestions into standard frames (see Figure 3.2). They built ontologies for terms relating to mobile phone parts and features, which were used by multiple components of their system. They evaluated their rules on their hand labeled dataset, comprising of 360 customer reviews.

Another line of work by Brun et al. [11] also employed manually crafted linguistic patterns, and evaluated it on a small dataset of 60 manually labeled reviews about printers. They report the precision, recall, and F-measure values as 77%, 70%, 73% respectively. Their system used four main components:

- A structured terminology of the target topic which encoded concepts and associated terms. Considering a product like printer, examples of structured terminology would be: 'part-of' concepts of the product (*paper tray*), the manufacturer of the

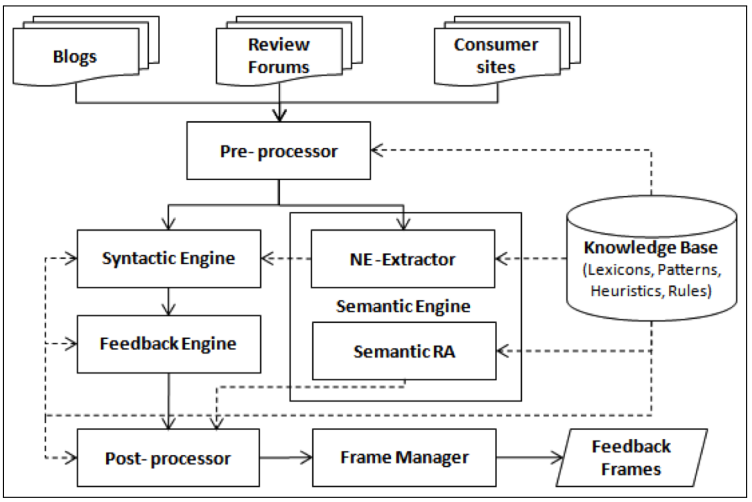



FIGURE 3.2: Suggestion mining system by Viswanathan et al. (2011). Figure taken from [70]

$$\begin{bmatrix} \text{SUBJECT} & [\text{SEMANTIC-TYPE} = \text{MANUFACTURER}] \\ \text{PREDICATE} & \begin{bmatrix} \text{TENSE} = & \text{PAST} \\ \text{ASPECT} = & \text{PERFECTIVE} \\ \textit{Modal} = & \text{YES} \end{bmatrix} \end{bmatrix}$$

FIGURE 3.3: One of the suggestion patterns by Brun and Hagege, 2013. Figure taken from [11].

product (*Xerox*), physical characteristics (*weight*), and commercial/technical characteristics (*speed, price*). These are potential targets on which suggestions may apply. The authors identified some main concepts in the domain and extracted most frequent noun modifiers for those concepts.

- A thesaurus of vocabulary related to the expressions of suggestions. It contained words which are commonly used in wishes, regrets and misses. It was obtained by using the words *wish, regret, miss, lack*, as seed and gathering verbs and nouns which convey the same or related semantics.

- A fine grained linguistic parser. An in-house parser was used to tag the morphological features of words, in particular modality and aspect. Dependencies such as SUBJECT, OBJECT, MODIFIERS, named entities were also tagged.

- An extractor for suggestions for improvement. This was the core component of their system, and comprised of general syntactico-semantic patterns that were matched against the data processed using the previous components. The arguments used in the patterns were domain specific terminology. For example, the pattern in Figure 3.3 can be phrased as, "a manufacturer entity, which is subject

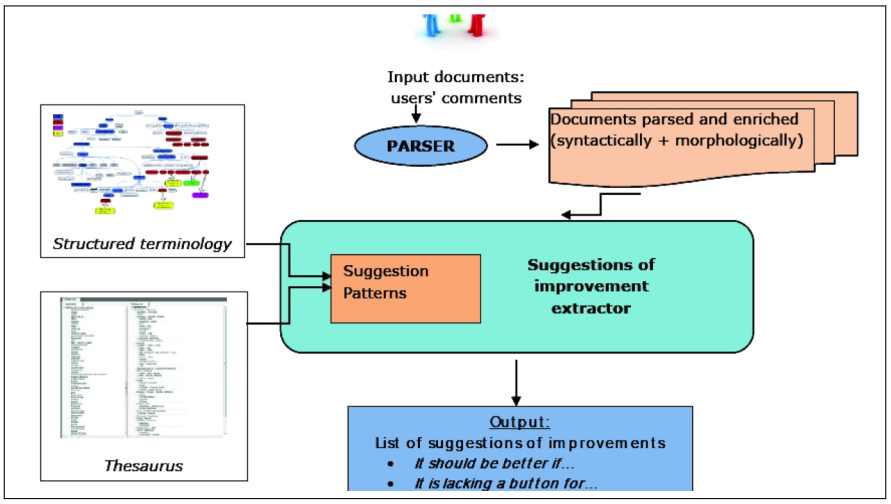



FIGURE 3.4: Suggestion Mining system by Brun and Hagege, 2013. Figure taken from [11].

of a modal verb used in the past tense and perfective aspect', and can identify a sentence like, *HP should have made the bin deep enough*, but does not identify: *HP should be ashamed*.

A few other works adapted similar approaches of identifying linguistic patterns and extracting the text which match this pattern from a large number of reviews [27]. Some of the linguistic patterns are similar across these works, while others remain restrictive to the domain. It is difficult to replicate these systems or compare them with the our results mainly due to the unavailability of one or more of the following: linguistic resources like lexicons, ontologies, in-house preprocessing components like parser and name entity extractor, and the datasets used for evaluation.
However, these studies give an idea of the classification performance obtained using rule based systems. It is observed in the previous works that the rule based systems tend to provide a higher precision than recall.

### 3.2.2 Supervised Machine learning

To the best of our knowledge, the very first attempt to perform suggestion mining using machine learning was performed by Wicaksono and Myaeng [72]. They experimented with two different types of classification algorithms, SVM and HMM [72–74]. However, they referred to the task as *Advice Mining*, and targeted the extraction of advice sentences in weblogs and discussion threads. In their first work, Wicaksono and Myaeng [72] identified the characteristics of advice sentences appearing in the dataset and defined relevant feature types for their classifier. The features were: a set of manually picked

expressions which tend to appear in suggestions, a proper noun directly attached to a modal verb as a nominal subject, a set of clue verbs constructed by expanding a seed set of verbs (suggest, recommend, advice etc.) using Framenet, appearance of *imperative mood expressions*, and presence of an *opinionated copula* in the sentence. The imperative mood expressions were identified using a set of rules. A copula is the relation between the complement of a copular verb (is, am, are) and the copular verb itself, and an opinionated copula is where the complement has a subjectivity score (using SentiWordNet) is above the threshold of 0.69. They used these features with a Support Vector Machine (SVM) classifier, and a labeled dataset of travel blogs. In order to prepare the dataset, they first filtered out a sample of blog entries by using clue words like *suggest, advice, recommend, tip*, etc. In our opinion, this approach of filtering the real data might bias the results towards the proposed features since the authors also used these clue words as one of the features. The dataset used in this work is not available for further investigation. The classification results were 0.523, 0.665, 0.586 for the average precision, recall and F-1 score respectively, for both the classes. Evaluation was performed on a held out test dataset which was 27% of the total annotated dataset.

In their subsequent work [74], Wicaksono and Myaeng defined the advise mining problem by proposing the concept of Advice-revealing text units (ATUs). They defined ATUs as comprising of two elements: *advice-revealing sentences*, and *context sentences*, latter being the sentence which explain or clarifies the advice revealing sentence usually describing when, where, or in what situation people would find the advice relevant. An example of ATU would be:

Advice revealing sentence: *But you are advised to exchange only a small amount at the airport as you won't get a good rate there*
Context sentence: *I am planning a 2 week trip to Bangalore and am not sure whether it is better to change my Euros here or to wait until I get to India.*

In this work, they performed experiments on the extraction of both kind of sentences using Conditional Random Field (CRF) based models. They consider the extraction of advice sentences from travel forums as a sequential labeling problem, based on their observation that advice revealing sentences as well as context sentences tend to appear contiguously. They use 2D-CRF model for the same. They also present a solution to extract both kind of sentences in a single pass by using two new CRF based models referred to as Multiple Linear CRF (ML-CRF) and 2 Dimensional CRF Plus (2D-CRF+). For the dataset, they crawled travel forums and randomly selected 150 threads comprising of 5199 sentences. They manually labeled ATUs with the help of two annotators. They used the intersection of annotation judgments from the two annotators for the

| **Syntactic** |
|---|
| 1. Whether or not a target sentence contains an imperative mood expression |
| 2. Discovered class sequential rules (CSRs) |
| 3. List of a target sentence's typed dependencies |
| 4. Presence of forum-specific cue phrases such as "thank you", "enjoy your trips", etc. (characterizing non-advice) |
| **Context Features** |
| 1. Jaccard similarity between a target sentence and its N preceding sentences |
| 2. Jaccard similarity between a target sentence and its M succeeding sentences |
| 3. Whether a target sentence and its N preceding sentences are in the same post |
| 4. Whether a target sentence and its M succeeding sentences are in the same post |
| **Semantic Features** |
| 1. Sentence informativeness score |

TABLE 3.2: Features used in the model of Wicaksono and Myaeng [74]

advice sentences, and union of judgments for the context sentences. On average each advice revealing sentence had 3 context sentences. The dataset from this work was made available by the authors upon request, and is used for the experiments in this thesis.
For advice sentence extraction, the identified features fall into the category of Syntax, Context, and Semantics (Table 3.2). Some of their context features are only suitable for the domains which are structured as discussion threads. Same applies to the semantic feature of sentence informativeness, which is a summation of informativeness scores of all the nouns contained in a sentence. The informativeness score of a noun is further calculated using the weighted sums of the notions introduced by them as *local* and *global informativeness*. Local scope refers to a sentence, and global scope refers to a thread.

For context sentence extraction, it was assumed that all advice revealing sentences have been extracted. Context sentences are located within the same thread, and an advice revealing sentence may act as a context sentence for other advice sentences. Each advice revealing sentence is paired with every other sentence of the thread, and a prediction is made for each pair whether the two sentences hold an advice-context relation. The methodology mainly comprised of performing a binary classification of each advice-context tuple by training a SVM and Maximum Entropy based classifier. The features comprised of jaccard similarity and semantic similarity between the two sentences, location of context sentence with respect to the advice sentence, whether the context sentence has been previously identified as an advice sentence, and whether the context sentence candidate contained a proper noun.
In Chapter 4, we also include context in our problem definition, but our notion of context is broader as compared to that of Wicaksono and Myaeng.

To the best of our knowledge, the work by Dong et al. [18] is the only attempt to perform suggestion extraction from tweets. Their objective was to extract suggestions for product improvements from the tweets which mention the product. They modeled a classifier to predict whether a given tweet about Microsoft Windows' phone expresses

a suggestion for improvement or not. They compared the performance of Support Vector Machines (SVM) [15] and Factorisation Machines (FM) [61] for building this the classifier. They prepared a dataset by collecting the tweets which mentioned *Windows Phone 7*, and manually annotating them with the labels *suggestion and non-suggestion*. A total of 3000 tweets which were given the same label by the annotators were retained in the final dataset. 238 out of these 3000 tweets were labeled as suggestions.
In addition to the unigram features, Dong et al. used two additional feature types: hashtags, and frequent phrases which tend to appear in suggestions. The frequent phrases were automatically mined using a pattern mining algorithm (PrefixSpan) on the suggestions obtained from Windows phone's official feedback website. They evaluated both SVM and FM with and without the two additional feature types, and used oversampling and thresholding to overcome the class imbalance in the datasets. They demonstrated that Factorisation Machines with class balancing and all the features gives the highest performance against the rest of the configurations. The best results were obtained using Factorisation Machines with all the features, which were 71.06, 67.86, and 69.42 for precision, recall, and F-1 score respectively, for the suggestion class. SVM also performed best when all the features were used as compared to the bag of words baseline. The evaluation was performed using a 5 fold cross-validation on the entire dataset.

Overall, in both rule based and machine learning approaches, the works so far focused on explicitly identifying the patterns or features to improve classification performance over simple baselinesf f like unigram features. Some of the approaches employ domain specific patterns, which are to be manually identified each time the target domain changes. In the case of machine learning approaches, only non-neural classifiers have been employed so far, among which SVM was a popular choice.

The state of the art in machine learning and opinion mining is moving towards representation learning approaches which use large unlabeled datasets and neural network architectures to automatically learn suitable feature vectors [42]. Such approaches have been shown to outperform the n-gram and manually determined feature engineering approaches in a number of popular tasks like sentiment analysis [67]. In the case of suggestion mining, there is a lot of scope to investigate into neural network based representation learning methods and neural architectures in order to minimise manual feature engineering efforts, as well as to evaluate the state of the art machine learning algorithms on the problem of suggestion mining.

In this thesis, we perform an extensive evaluation of manually determined features for suggestion mining (Chapter 5), as well as employ neural network based representation learning approaches (Chapter 6, 7).

| Pattern | Example (selected sentence segment) |
|---|---|
| there should be (DT) | There should be a reply to all button so I can do 100 items in just 2 minutes. |
| (allow\|let) USER to | Allow us to open and pay the invoices. |
| VB (DT) option | Give me the option to search for auctions in Europe. |
| I wish COMPANY | I wish eBay would make my eBay emails open in the app instead of safari. |
| MD be (ADV) ADJ | From the standpoint of store owner would be very helpful to be able to create sale put on and off vacation mode and edit categories. |
| MD (like\|prefer\|love) to | i would love to have one button to remind all my buyer to leave feedback with my message and link to the feedback page. |
| stop VBG | Please stop sending me an email every time a bid happens. |
| ability (to\|of) | Only thing I would change is to have the ability to do searches by years |

FIGURE 3.5: Linguistic patterns used for distant supervision by Moghaddam, 2015 [43]

## 3.3 Distant Supervision

To the best of our knowledge, the study conducted by Moghaddam [43] related to suggestion mining has focused on using distant supervision for extracting suggestions for improvement from customer feedback. This study first defines a set of what they refer to as *lexical-Part of Speech* patterns to identify suggestion reviews in a large dataset of about 50,000 eBay App reviews on App store and Google Play. The predicted suggestions were then used as positive instances to train a SVM based classifier which used bag of words features. They performed distant supervision at the review level, rather than the sentence level. They compared the results with a SVM classifier trained on a manually labeled dataset. The classifier trained on manually labeled reviews obtained higher precision, recall and F-measure values (0.38, 0.78, 0.51 respectively) on a test dataset, than the distant supervision method (0.32, 0.74, and 0.46 respectively). In order to extract the exact suggestion sentence within the reviews identified in the first stage, they simply used a SVM classifier with bag of words features trained on a manually annotated sentence dataset. In this thesis, we propose alternative methods of distant supervision by employing large silver standard datasets collected from the web, and learning feature representations from this dataset using neural network architectures (Chapter 7).

## 3.4 Summary

The table below summarizes the state of the art for the research questions addressed in this thesis.

| Related Work | Domain | RQ | Method | Dataset |
|---|---|---|---|---|
| Rules Based Approaches | | | | |
| Ramanand et al. (2010) [60] | Product reviews | RQ 1,2 | Linguistic patterns, domain lexicon | Unavailable, 47,500 sentences (1% suggestions) |
| Viswanathan et al. (2011) [70] | Product Reviews | RQ 1,2 | Linguistic patterns, domain structured terminology | Unavailable |
| Brun and Hagege (2013)[11] | Product reviews | RQ 2 | Linguistic patterns, domain structured terminology | Unavailable, 60 reviews (% suggestions unavailable) |
| Iacob and Harrison (2013) [27] | Product reviews | RQ 2 | Linguistic patterns | Unavailable, 3000 sentences (% suggestions unavailable) |
| Supervised Machine Learning | | | | |
| Wicaksono and Myaeng (2012)[72] | Weblogs | RQ 2 | Binary sentence classification. Support Vector Machines with manually defined features. | Unavailable, 7239 sentences (10% advice) |
| Wicaksono and Myaeng (2013) [73] [74] | Discussion Forums | RQ 2 | Sequence labelling of sentences. Conditional Random Fields and Hidden Markov Models. Manually defined features. | Available, 5199 sentences (44% advice) |
| Dong et al. (2013)[18] | Tweets (product related) | RQ 2 | Binary sentence classification. Support Vector Machines, and Factorization Machines. Manually defined features. | Available, 3000 tweets (8% suggestions) |
| Rule Based + Supervised Machine Learning | | | | |
| Moghaddam (2015)[43] | Mobile Application (eBay App) user reviews | RQ2, RQ 3 | Binary review classification. Support Vector Machines trained on dataset which was automatically labeled using manually identified patterns. Bag of words (unigram) features. | Unavailable, 50,000 reviews (15% suggestions) |

TABLE 3.3: A summary of the state of the art in suggestion mining. Rule based and machine learning methods have been equally popular. Only two datasets are openly available.

# Chapter 4

# Defining Suggestions for Suggestion Mining

## 4.1 Introduction

In this chapter we answer the first research question of this thesis: *What are suggestions in the context of suggestion mining?* The need for this research question arises from our observation that human perception of the term *suggestion* is subjective, and this effects the preparation of hand labeled suggestion datasets. The related works minimally focused on the formalization of suggestions in the context of suggestion mining, and also most of the previous manually labeled datasets remain proprietary. In order to compare approaches across different studies on suggestion mining, a formal definition of the scope of suggestions as well as benchmark datasets are required.
In this chapter we first present a qualitative analysis of suggestions, and based on our observations propose a formal definition of suggestions and annotation procedure for the benchmark datasets. Parts of this chapter have been published in [47–49]

## 4.2 Qualitative Analysis

As a part of qualitative analysis, we investigate the linguistic properties of suggestion sentences, relationship between sentiments and suggestions, and a layman's perception of suggestions. These observations are close to the existing related work in suggestion mining and related linguistic studies. In the next chapter, this analysis forms a basis of manually proposing features for suggestion mining.

### 4.2.1 Linguistic Properties

We observe that suggestions are often expressed by means of lexical cues, and grammatical moods.

**Lexical Cues:** Suggestions tend to contain certain keywords and phrases, like *suggest, suggestion, recommendation, advice, I suggest, I recommend, etc.* Most of the previous works created a hand made list of these words and phrases to use them as features with the classifiers. Our observations of the lexical expressions in suggestions are in-line with the previous works on suggestion mining, as well as the related linguistic studies. For example, Asher et al. [2] have performed an in-depth study of how affective content is explicitly and lexically expressed in written texts (Chapter 3). They list a category of verbs called *Advise*, and define it as, 'expressions which urge the reader to adopt a certain course of action or opinion'. Some examples of advise verbs are: advise, argue for, suggest, propose, wish, hope, etc. Martinez [39] also identify these lexical clues as a part of their study of suggestions (Table 4.1).

However, not all the suggestions contain these keywords and phrases. Some example

| **Type** | **Example** |
|---|---|
| Direct | I suggest that you ...<br>I recommend that you ...<br>I advice you to ...<br>My suggestion would be ...<br>Try using ...<br>Don't try to ... |
| Conventionalised forms | Why don't you ...?<br>How about ...?<br>Have you thought about ...?<br>You can ...<br>You could ...<br>If I were you, I would ... |
| Indirect | One thing (that you can do) would be ...<br>Here's one possibility ...<br>There are a number of options that you ...<br>It would be helpful if you...<br>It might be better to ...<br>It would be nice if ... |

TABLE 4.1: Three types of expressions of suggestions as defined by Martinez (2005) [39] with the examples of prevalantly used surface structures.

of such sentences are shown in Table 4.2. Also, Table 4.3 lists the top 10 unigrams and bigrams in the sentences tagged as advice and suggestion in the dataset used by Wicaksono et al. [73] and Dong et al. [18]; the ranking excludes the stopwords.

**Modality:** Modality is a grammatical category that allows the expression of aspects related to the attitude of a speaker towards his statement, in terms of degree of certainty, reliability, subjectivity, sources of information, and perspective [45]. Grammatical mood is a grammatical feature of verbs, used for signaling modality. Mood and

| Source | Sentence | Linguistic Properties |
|---|---|---|
| Electronics Reviews | I would recommend doing the upgrade to be sure you have the best chance at trouble free operation. | Subjunctive, Imperative, Lexical clue: *recommend* |
| Electronics Reviews | My one recommendation to creative is to get some marketing people to work on the names of these things | Imperative, Lexical clue: *recommendation* |
| Hotels Reviews | Be sure to specify a room at the back of the hotel. | Imperative |
| Tweets to Microsoft | Dear Microsoft, release a new zune with your wp7 launch on the 11th. It would be smart. | Imperative, Subjunctive |
| Discussion thread, Travel | If you do book your own airfare, be sure you don't have problems if Insight has to cancel the tour or reschedule it | Conditional, Imperative |

TABLE 4.2: Examples of similar linguistic properties in suggestions from different domains

| Travel | | Microsoft Tweets | |
|---|---|---|---|
| **Unigrams** | **Bigrams** | **Unigrams** | **Bigrams** |
| You | credit card | Microsoft | Windows Phone |
| tour | you will | Windows | Dear Microsoft |
| if | if want | WP7 | Microsoft needs |
| just | http www | phone | Come Microsoft |
| travel | make sure | need | Microsoft make |
| time | Europe board | nokia | Microsoft WP7 |
| like | tour director | make | If Microsoft |
| hotel | post Europe | needs | Microsoft really |
| did | travel tips | apps | really needs |
| need | United States | want | hope Microsoft |

TABLE 4.3: Top 10 unigrams and bigrams in the sentences labeled as advice and suggestion in the dataset provided by Wicaksono et al. [73] and Dong et al. [18] respectively

modality have been investigated in a vast amount of linguistic literature, and different kinds of typologies and terminologies have been suggested for moods. We observe that suggestion expressions often contain what may be referred to as *subjunctive, and imperative* moods.

Subjunctive mood is a commonly occurring language phenomenon in Indo-European languages, which is typically used in subordinate clauses to express action that has not yet occurred, in the form of a wish, possibility, necessity etc. [24]. The presence of these moods pose challenges for detecting sentiment polarity since the text containing subjunctive mood is likely to express sentiments towards a possible event rather than an actual event [8, 46]; however, it can be a good feature to detect suggestions. Some typical examples are: *If the Duke were here he would be furious* [19], *It is my suggestion that the students be sent to Tibet* [24]. The Imperative mood is used to express the requirement that someone perform or refrain from an action. A typical example would be, *Take an umbrella*, *Pray everyday* [58]. In the context of suggestion mining, Dong et al. [18] define some rules to detect the presence of imperative mood in a sentence, and use it as a feature. However, subjunctive mood was never associated with suggestions in any of the previous suggestion mining studies. Table 4.2 lists the examples of these

moods present in suggestion sentences.

### 4.2.2 Suggestion Mining and Opinion Mining

Opinion mining and sentiment analysis studies evolved from the earlier studies on subjectivity analysis and automatic detection of subjective and objective expressions in text [62, 75]. A large number of advice to fellow customers in online reviews tend to express suggestions based on factual information. We observe that suggestions can be both factual statements and subjective or opinionated expressions. Examples of suggestions expressing facts are, *'Top tip for frequent visitors is the Okura Club International which you can join for nothing and entitles bearers to free usage of the pool, showers and o-furo (bath) which you would normally have to pay for'*, *'As a general tip, if you need to use the internet whilst in Berlin, look up the Easy Internet cafes before you go, there's a huge one in Kurfurstendam'*, etc. Examples of suggestions largely containing subjective information are, *'I do recommend Kaffee Einstein.'*, *'Highly rated apps should appear near the top more and they should appear less and less frequently and the rating of the app dwindles.'*. These observations may call for a deeper investigation to derive a formal relationship between suggestions and subjectivity, however it currently out of scope for this thesis.

In the context of sentiment analysis, suggestion mining can be applied to very similar domains or data sources, for example, reviews, blogs, discussion forums, twitter etc. Sentiment expressions and suggestion expressions may also appear together in the same text, specially in the case of recommendation type of suggestions.

**Opinion classes:** In sentiment analysis, text can be generally categorised into three or more sentiment classes, while in suggestion mining a text is either a suggestion, or a non-suggestion. For a sentence classification task in suggestion mining, unlike sentiment analysis, only the positive class holds importance in order to extract suggestions from a given text, while in sentiment analysis both positive and negative classes are important in order to generate sentiment summaries. We observe that suggestion expressing texts are not limited to a particular class of sentiment. Figure 4.1 and Table 4.4 shows some examples from online reviews, highlighting how sentiments towards a reviewed entity can vary across different suggestions. While sentiment sentences are always defined a subjective, suggestions can be found in both objective and subjective type of text. Sentiments can minimally appear at the phrase level, while suggestions can only appear at or above clause level. Therefore, a suggestion bearing sentence may contain multiple sentiments.

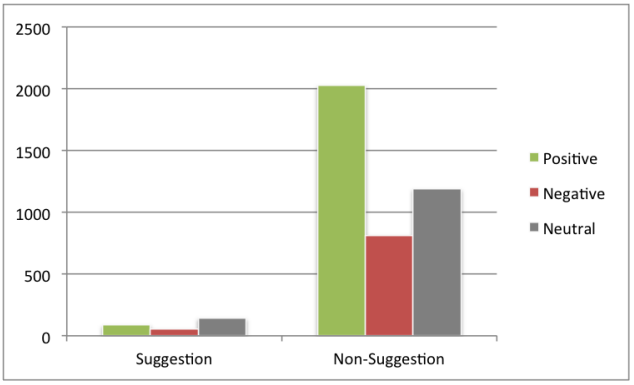



FIGURE 4.1: Distribution of sentiment polarities towards the hotel in suggestion and non-suggestion sentences for a hotel review dataset.

**Opinion Target:** In the case of reviews, some sentiment analysis benchmark datasets like SemEval datasets [57] exclude the text which is not about the opinion target, even though the text is found within the same review. The guidelines from SemEval 2015 Sentiment Analysis task [57] read: *Quite often reviews contain opinions towards entities that are not directly related to the entity being reviewed, for example, restaurants/hotels that the reviewer has visited in the past, other laptops or products (and their components) of the same or a competitive brand. Such entities as well as comparative opinions are considered to be out of the scope of SE-ABSA 15. In these cases, no opinion annotations were provided.*

We observe that some of the non-relevant sentences in review datasets are potential suggestions on closely related points of interest. Suggestions in hotel reviews may contain tips and advice about the reviewed entity, suggestions and recommendations about the neighbourhood, transportation, and things to do. Similarly, in product reviews, suggestions can be about how to make a better use of the product, accessories which go with them, or availability of better deals, etc. Table 4.4 gives some example of such sentences. Sentiment and Suggestions can have more than one type of target readers. Sentiments in reviews can convey useful information both to the brand owners, as well as the future consumers. Similarly, suggestions may convey ideas of improvement to the brand owners, or tips to the future consumers.

### 4.2.3 Layman Perception of Suggestions

We study the perception of the term *suggestion* by layman users, by performing a trial annotation where no detailed annotation guidelines were provided, and the annotators were simply asked to choose between the labels *suggestion* and *non-suggestion* for a given sentence. We collected 100 sentences, 20 sentences each from 4 different domains:

| Sentence | Domain | Sentiment | Relevant |
|---|---|---|---|
| One more thing- if you want to visit the Bundestag it is a good idea to book a tour (in English) in advance | Hotel | Neutral | No |
| Be sure to pick up an umbrella (for free) at the concierge if you anticipate rain while sightseeing. | Hotel | Positive | Yes |
| You are going to need to buy new headphones, the stock ones suck | MP3 player | Negative | Yes |
| If you strictly use the lcd and not the view finder , i highly recommend the camera . | Camera | Positive | Yes |
| For those of you who already bought this camera , I suggest you buy a hi-ti dye-sub photo printer | Camera | Neutral | No |
| If only it played stand alone avi files | DVD player | Neutral | Yes |
| Would be really good if they have given an option to stop this auto-focusing | Camera | Neutral | Yes |

TABLE 4.4: Examples of suggestions from review datasets for sentiment analysis. *Sentiment* refers to the sentiment towards the reviewed entity, while *Relevant* indicates if the suggestion is relevant for calculating the sentiment towards the reviewed entity.

hotel reviews, restaurant reviews, software developers' suggestion forum, and tweets. Each of the 20 sentences for each domain were picked from different documents, and it was made sure that at-least 5 potential suggestion sentences were picked. In the case of tweets, entire tweet was used instead of a sentence. Each sentence was annotated by 50 annotators. Crowdflower[1], a paid crowdsourced data annotations service platform was used for the same. More details on Crowdflower platform are provided in section 4.4. We also employed two variations in the process:

1. Providing candidate sentences with and without context. Context here refers to the information about the source domain of the text, as well as the text of the entire document from where a sentence is extracted.

2. Two vs four options for labels. Two options include *Suggestion, Non-Suggestion*; while four options include *Positive sentiment, Negative sentiment, Suggestion, None*, where multiple labels could be chosen in the case of four options.

Following patterns were observed in the annotations:

- For customer reviews, when only two options were provided for the labels, there was a tendency to label the sentiment sentences like, 'They serve really nice breakfast.' as suggestions.

- Similarly, annotators tend to choose *Suggestion* if they inferred a suggestion from the sentence, even if the sentence did not contain the expressions typically used in a suggestion.

- When four options were provided, the above mentioned sentences (positive sentiments and implicit suggestions) were only labeled as *Sentiment*. Therefore, the

[1] https://www.crowdflower.com

sentences which were labeled as suggestions were mostly explicitly expressed suggestions.

- When the context was provided, annotators tend to label a sentence only for the sentiment. It appeared that they were biased from the overall sentiment of the text or neighbouring sentences. This tendency is lower in the case of four options.

Tables 4.5 and 4.6 provide examples of the sentences which were perceived as suggestions and non-suggestions by more than 50% of the total annotators. In order to generalise the type of sentences considered as suggestions by these annotators, we also match these sentences with their potential opinion expression category according to Asher et al [2], namely *Inform, Assert, Tell, Remark, Think, Guess, Blame, Praise, Appreciation, Recommend, Suggest, Hope, Anger, Astonishment, Love, Hate, Fear, Offence, Sadness/Joy, Bore/Entertain.* A sentence with multiple clauses may contain more than one expression. We design annotation procedure and guidelines based on the observations from this annotation study. Except the *Suggestion*, *Recommendation*, and *Request* opinion expressions, suggestions containing rest of the expression types are in an implicit form. Since humans inherently infer suggestions, the layman annotators tend to consider both implicit and explicit form of suggestions as suggestions. We limit the scope of this thesis to the detection of explicitly expressed suggestions, and deal with the challenges associated with the explicit form of suggestions first. However, inferring suggestions from their implicit forms can be a potential future investigation in this area.

Identifying suggestions on the basis of the type of opinion expression in a sentence may require certain level of linguistic understanding by the annotators. We also observe that workers on crowdsourcing platform tend to ignore detailed guidelines. On one hand, crowdsourcing with layman annotators may not be the best suited method for creating suggestion mining benchmark datasets, while on the other hand, trained annotators are either slower (volunteers) or expensive. In the next section, we formally define the scope of this thesis. Based on this definition, we propose a two-phase annotation method which involves both layman annotators as well as expert annotators.

## 4.3 Defining Suggestions for Suggestion Mining

The Oxford dictionary defines suggestion as, *An idea or plan put forward for consideration.* Some of the listed synonyms of suggestions are *proposal, proposition, recommendation, advice, hint, tip, clue.* As mentioned before in Chapter 3, many linguistic studies have studied and defined suggestions as a speech act. They perform a fine grained analysis of syntactic and pragmatic characteristics of suggestions. Searle [66] mentions that,

| **Sentence** | **Source** | **Context Given** | % **Annotators** | **Category** |
|---|---|---|---|---|
| We did not have breakfast at the hotel but opted to grab breads/pastries/coffee from the many food-outlets in the main stations. | Hotel Review | Yes | 80 | Tell |
| If you do end up here, be sure to specify a room at the back of the hotel. | Hotel Review | Yes | 100 | Suggest |
| I recommend going for a Trabi Safari | Hotel Review | Yes | 90 | Recommend |
| Definitely come with a group and order a few plates to share. | Restaurant review | No | 52 | Recommend, Suggest |
| Please provide consistency throughout the entire Microsoft development ecosystem!" | Suggestion Forum | Yes | 70 | Request |
| We need a supplementary fix to the issues faced by existing affected users without resorting them to contacting Microsoft customer support | Suggestion Forum | No | 60 | Need/ Necessity |
| RT larrycaring: "every time someone send you hate send them this back" this is v good advice I'll use it every time antis insult me | Twitter | NA | 84 | Suggest, Recommend, Appreciation |

TABLE 4.5: Examples of sentences accepted as suggestions by layman annotators. % Annotators in each table refers to the percentage of total annotators which labeled a given sentence as a suggestion.

| **Sentence** | **Source** | **Context Given** | % **Annotators** | **Category** |
|---|---|---|---|---|
| This is much safer and far more convenient than hailing taxis from the street. | Hotel Review | No | 70 | Assert |
| There is a very annoying bug in the Windows 10 store that hides apps from listing. | Suggestion Forum | Yes | 64 | Tell |
| Just returned from a 3 night stay. | Hotel Review | No | 94 | Tell |
| The internet was free but I do think people should only be allowed say 30 mins and its written in a book, thats a fair way I think. | Hotel Review | Yes | 72 | Tell, Think, Suggest |
| I had so much fun filming all kinds of tips and advice (and some yoga!) yesterday | Twitter | NA | 84 | Inform, Tell, Entertain |
| There is a parking garage on the corner of Forbes so its pretty convenient. | Restaurant Review | No | 62 | Inform, Appreciation |

TABLE 4.6: Examples of sentences accepted as non-suggestions by layman annotators. % Annotators in each table refers to the percentage of total annotators which labeled a given sentence as a non-suggestion.

*Suggestions belong to the group of directive speech acts, which are those in which the speaker's purpose is to get the hearer to commit him/herself to some future course of action.* These studies mainly define how suggestions should be expressed in a standard use of language. In the context of text mining, we are dealing with user generated text on the web, which can be associated with multiple contexts, like the end user, domain etc. While the previous studies on suggestion mining fall under text mining, they followed a generic definition of suggestions and did not formally define the scope of suggestions. As

we saw in our layman annotation study, context may affect an annotator's judgment. In the absence of context, different annotators associate different contexts to a candidate sentence. We observe that the following concepts form an integral part of defining a suggestion in the context of suggestion mining.

We now propose an empirically driven and context-based definition of suggestions. Given that:

- $s$ denotes the surface structure of a sentence,
- $C$ denotes additional context provided with $s$, where the context can be a set of values corresponding to certain variables, and
- $a(s, C)$ denotes the annotation agreement for the sentence, and $t$ denotes a threshold value for the annotation agreement,

we write $S(s, C)$ to denote the *suggestion function*, which is defined as

$$S(s,C) = \begin{cases} Suggestion, & \text{if } a(s,C) \geq t \\ Non\text{-}suggestion, & \text{if } a(s,C) < t. \end{cases} \tag{4.1}$$

***Surface structure.*** Different surface structures [14, 16] can be used to express the underlying intention of giving the same suggestion. For example, *The nearby food outlets serve fresh local breakfast and are also cheaper* and *You can also have breakfast at the nearby food outlets which are cheaper and equally good.*

Linguistic studies in the past studied the way suggestions are expressed (see Table 4.1) and provided the examples of lexical elements used in the surface structures for suggestions.

***Context.*** When dealing with specific use cases, context plays an important role in distinguishing a suggestion from a non-suggestion. Context may be present within a given sentence. It can be a set of values corresponding to different variables that are provided explicitly and in addition to a given sentence. One or more of the following variables can constitute the context:

***Domain.*** We refer to the source of a text as *domain*. In the process of dataset annotation, we closely studied some of the domains; Table 6.3 shows how the distribution of suggestions varies with the domains.

***Source text.*** The text in the entire source document to which a sentence belongs may also serve as a context, giving an insight into the discourse where the suggestion appeared.

***Application or use case.*** Suggestions may sometimes be sought only around a specific topic, for example, mining room tips from hotel reviews. Suggestions can also be selectively mined for a certain class of users, for example, mining suggestions for future customers. All non-relevant suggestions in the data may be regarded as non-suggestions in this case. Previous studies on suggestion mine from online reviews operated on this kind of context. For example, only suggestions for improvement were identified from the product reviews, whereas suggestions meant for the fellow customers [48] were considered as non-suggestions.

We now propose an empirically driven and context-based definition of suggestions. Given that

- $s$ denotes the surface structure of a sentence,
- $C$ denotes additional context provided with $s$, where the context can be a set of values corresponding to certain variables, and
- $a(s, C)$ denotes the annotation agreement for the sentence, and $t$ denotes a threshold value for the annotation agreement,

we write $S(s, C)$ to denote the *suggestion function*, which is defined as

$$S(s,C) = \begin{cases} Suggestion, & \text{if } a(s,C) \geq t \\ Non\text{-}suggestion, & \text{if } a(s,C) < t. \end{cases} \tag{4.2}$$

Depending on the choice of $C$, and, hence, on the value of $a(s, C)$, we identify four categories of sentences that a suggestion mining system is likely to encounter.

***Explicit suggestions.*** *Explicit suggestions* are sentences for which $S$ always outputs *Suggestion*, whether $C$ is the empty set or not. The suggest, recommend, request, need/necessity category of opinion expressions in Table 4.5 are mostly found in the explicit expressions of suggestions. They are like the *direct* and *conventionalised* forms of suggestions as defined by Martínez Flor [39] (Table 4.1). It may also be the case that such sentences have a strong presence of context within their surface form, as in illustrated by *If you do end up here, be sure to specify a room at the back of the hotel.*

***Explicit non-suggestions.*** These are the sentences for which $S$ always outputs *Non-suggestion*, whether $C$ is the empty set or not. For example, *Just returned from a 3 night stay.*

***Implicit suggestions.*** These are sentences for which $S$ outputs *Non-suggestion* only when $C$ is the empty set. Typically, implicit suggestions do not posses the surface form of suggestions but the additional context helps the readers to identify them as suggestions. For example, *There is a parking garage on the corner of Forbes, so its pretty convenient* is labeled as a suggestion by the annotators when the context is revealed as that of a restaurant review. A sentence such as *Malahide is a pleasant village-turned-dormitory-town near the airport* can be considered as a suggestion given that it is obtained from a travel discussion thread for Dublin. These kind of sentences are observed to have a lower inter annotator agreement than the above two categories.

***Implicit non-suggestions.*** These are sentences for which $S$ outputs *Suggestion* only when $C$ is an empty set. Typically, an implicit non-suggestion posses the surface form of suggestions but the context leads readers to identify them as non-suggestions. Such sentences may contain sarcasm. Examples include *Do not advertise if you don't know how to cook* appearing in a restaurant review and *The iPod is a very easy to use MP3 player, and if you can't figure this out, you shouldn't even own one* appearing in a MP3 player review.

Based on the above four categories, we can define the scope of the suggestion and non-suggestion classes for open domain suggestion mining. For open domain suggestion mining, we propose to limit the scope of suggestions to the *explicit suggestions*. Therefore, we set the task definition for *open domain* suggestion mining to be:

> Let $s$ be a sentence. If $s$ is an explicit suggestion, assign the label *Suggestion*. Otherwise, assign the label *Non-suggestion*.

The proposed categories provide the flexibility to change the scope of classes in a well defined manner, as well as to define context as per the application scenario.

**Unit of suggestions**: Within the scope of this research, we consider sentence as the unit of suggestion, which is also the case with the previous studies. In order to perform text classification based suggestion mining, a pre-processing step would be to split the text into the desired units. Some pros and cons can be associated with the sentential units.

Clause is the smallest unit within which a suggestion can be expressed. A sentence may contain one or more clauses, and therefore using sentence as a unit increases the chances of more details on the suggestion to be present within the sentence and not to be explicitly sought from the neighbouring text. The sentence boundary detection is

done using part of speech taggers, and is therefore prone to some errors, specially when the text is obtained from online platforms, like reviews and social media. We observe that automatically split sentences can be either incomplete, or even if correctly split they may refer to an entity mentioned in the preceding text. For example, *But if you are coming here for the service that you expect for what they claim to be..*, and, *We left it too late.* However, we also observe that majority of sentences which were labeled as suggestions in the absence of an additional context, contained sufficient content for human annotators to decide its label.

**Domains in Suggestion Mining**: We refer to the source of a text as *domain*. In this thesis, we perform domain specific, cross domain, and open domain experiments for suggestion mining. In the process of dataset annotation, we closely studied some of the domains; Table 6.3 shows how the distribution of suggestions varies with the domains. Below are some potential domains for suggestion mining:

1. **Online Reviews:** We observe that reviews can contain two kinds of suggestions, suggestions to brand owners, and suggestions to consumers. The two type of suggestions can vary in their proportions depending on the type of entity being reviewed. For example, hotel and electronics reviews tend to have a large proportion of suggestions targeted to future customers as compared to the hotel authorities. But book reviews hardly contain any suggestions to the fellow readers, however, suggestions are mostly present in the form of wishes about what the readers would have liked to see in the book. In general, suggestions may not be present in every review, therefore, suggestions appear sparsely in review datasets.

2. **Suggestion Forums:** Suggestion forums are dedicated forums to provide suggestions for improvement in an entity. Often people tend to provide the context in such posts, which gets repetitive in the case of large number of posts under the same topic. Suggestion mining methods can extract the exact sentence in the post where a suggestion is expressed. There is at-least one suggestion per post, which results in larger number of suggestions in these datasets.

3. **Tweets:** Tweets about a commercial entity also contain suggestions to the brand owners as well as to the customers. The dataset provided by Dong et al. [18] only contain the tweets addressed to Microsoft and not the tweets which merely mention Microsoft, therefore the suggestions in this dataset are mostly the suggestions for product improvement. We also studied the appearance of open domain suggestions on twitter, and they mostly contain expressions of advice and recommendations targeted to an open audience. For example, *Advice to consider: Hand Exercises for*

*Knitters, Crocheters, and other Handcrafters with Col... https://t.co/pkV4zXUc3S via @YouTube.* Suggestions tend to appear sparsely on Twitter.

**Problem definition revisited:** After proposing our definition and scope of suggestions, the problem definition for this thesis can be refined as:
Given a set S of sentences $\{s_1,s_2,s_3,...,s_n\}$, predict a label $l_i$ for each of statement in S, where $l_i \in$ {suggestion, non suggestion | suggestion = {explicit suggestions}, non-suggestion = {explicit non-suggestions, implicit suggestions, implicit non-suggestions}}.

## 4.4 Creating Benchmark Datasets

In this section, we detail the development process of manually annotated datasets used in this thesis, as well as the takeaways from the dataset development process. We performed two phases of manual annotation for each dataset. The first phase is performed using paid crowdsourcing where each sentence is annotated by multiple annotators, and the second phase is performed by a single yet expert annotator.

### 4.4.1 Phase 1: Crowdsourcing Annotations

**Crowdflower Platform:** Crowdflower[2] is a crowdsourcing and data mining company which provides full fledged services to individual customers and organisations, under which their clients can access an online workforce to manually clean, label, and enrich data. Various options are provided in the platform to ensure the reliability of this workforce, and pay them accordingly. The platform supports many different use cases, like sentiment analysis, search relevance, and data categorization. The total cost involved in the annotation process includes a platform fees and direct payments to the workers.
A proper understanding of the platform is required before an annotation job is publicly launched, which includes the platform functionality and terminology. In this regard, a very useful feature of Crowdflower is that it allows for a trial launch of the annotation job to a customer's internal team, where everything remains same as the actual task, except that no payment is made to the annotators. Once approved and tried by the internal team, the job can be released to the workforce. Only those workers can see the annotation job and choose to do it whose profile has the characteristics chosen by the customer. There is a provision of testing the contributors against a set of *test questions* with known answers. A graphical editor allows the user to easily upload the data and set up various aspects of the annotation process, like the instructions, desired annotator

[2] https://www.crowdflower.com/

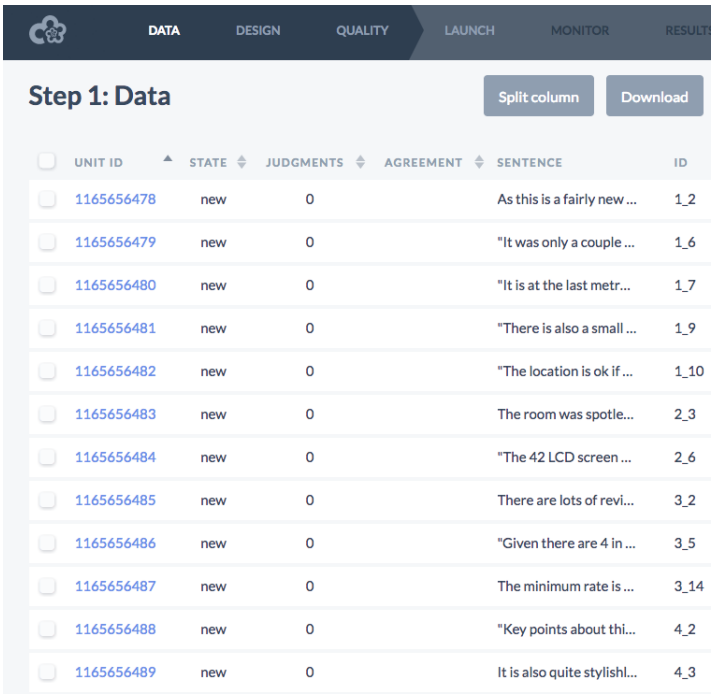



FIGURE 4.2: A screenshot of how the raw data looks upon uploading on crowdflower platform

profile, test questions, payment per annotation, number of minimum and maximum data instances to be annotated by each annotator, etc. More details on the above mentioned aspects of annotation process are provided below.

**Job Design:** The very first step to create a new annotation job is to load the data into the job. Crowdflower accepts raw data in .tsv, .csv, .xlsx, and .ods formats (Figure 4.2). Rows of data represent individual data points that can be presented to the annotators to work on. In our case, the data was in a .csv format with the fields 'id', 'sentence', and 'source', which stand for the sentence id, sentence, and full text from the document from where the sentence is extracted. While working on a job, annotation job appears in batches of rows to work on at a time, which are referred to as 'Pages' (see Figure 4.3). Annotators are not paid per individual row, rather per page. In our case, each page has 8 sentences.

**Annotators:** Crowdflower offers a large pool of annotators (referred to as *workers*) with a varying experience of performing data labeling tasks (referred to as *jobs*) on crowdflower. Each worker has a profile on crowdflower, where his experience is quantified into three levels, level 1 being a new worker who haven't yet performed any job. Level 1 may also mean that a worker could not perform any job because he/she did not achieve a threshold score in the respective test questions. A worker's country and

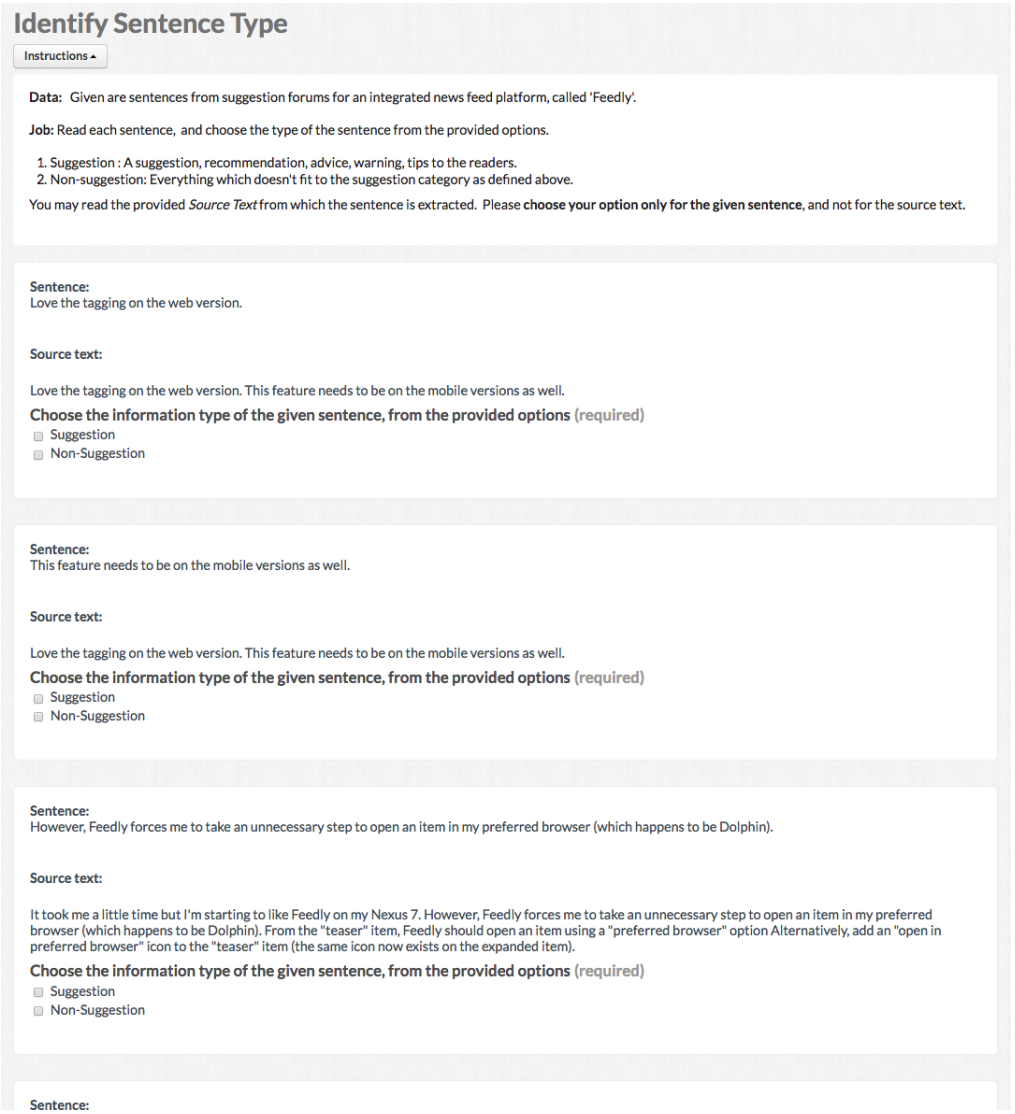



FIGURE 4.3: A screenshot of how each page of the annotation job appears to the annotators on the crowdflower platform

language proficiency is also recorded in his profile.

**Quality Control:** In the trial phase of annotations, we observed that under certain scenarios the credibility and quality of labeled data can be compromised.

1. The annotation instructions lack clarity.
2. Since the annotators are paid according to the number of annotations completed by them, it can be the case that the annotators choose a random option without actually taking time to understand the instructions, and reading individual data point.

3. The annotators do not possess the level of language understanding required for the task.

4. The annotators are very new to the platform and data labeling tasks.

We adapt the following solutions which are provided with the Crowdflower platform to overcome the above mentioned quality issues, respectively.

1. The job was launched to our internal team members first, and their feedback was incorporated on the clarity of instructions.
   Before being allowed to perform a job, the annotators are presented with a set of test sentences which are similar to the actual questions except that their answers have already been provided by us to the system. We also submit the explanation behind the correct answer. A worker is shown the right answer and explanation upon his submission of the answer. Workers can also provide feedback referred to as *contentions*, if they think that a wrongly answered test question was unfair as per their understanding of the instructions. This way the test questions serve two purposes: test the annotators competency and understanding of the job, and train the annotator for the job. Crowdflower recommends certain best practices to prepare effective test questions [3].
   We submit 30 test questions for each dataset. Each starting annotator is presented with 10 test questions, and only the annotators who score 70% or more are allowed to proceed with the job.

2. If an annotator passes the test and starts the job, the remaining unseen test questions are presented to them in between the regular sentences, and without being notified. In our case, every 1 question out of 8 questions on a page is a hidden test question for the annotator. The accuracy score of a contributor on test questions is referred to as *Trust score* in a job. If an annotator's trust score drops down a certain threshold during the course of the annotation, system does not allow them to proceed further with the job. This threshold score in our case was set to 70%.
   In addition to the hidden test questions, a minimum time for each annotator to stay on one page of the job (8 questions in our case) can be set. If annotators appear to be faster than that, they are automatically removed from the job. We set this time to 40 seconds (5 seconds on average for each sentence) for our jobs.

3. Crowdflower allows the customers to restrict or choose the countries from where an annotator can join. Only 15 countries could be selected; we considered the following countries: Australia, Bangladesh, British Indian Ocean Territory, Canada,

[3] https://success.crowdflower.com/hc/en-us/articles/213078963-Test-Question-Best-Practices

Germany, Hong Kong, Ireland, Malta, Sri Lanka, India, Israel, Nepal, United States, United Kingdom, British Virgin Islands, U.S. Virgin Islands. We tried to choose the countries where English is a popular language, but which is also likely to have a large crowdsourcing workforce. It was observed that most of the annotators came from: Canada, Australia, Germany, India, United Kingdom, Ireland, and USA.

4. Crowdflower assigns the workers with experience levels based on the number of times they successfully passed the test questions and performed a job. i.e. level 1, 2, 3, where level 1 is assigned to the absolute newcomers with no job experience. Usually the workers with higher experience level do not pick up low paying annotation jobs. We only allowed workers with experience level 2 or more for our annotations. Therefore, the annotation reward was required to be competent with other jobs. We paid 10 cents for each page, i.e. 8 sentences, which means a maximum of 10 cents for 40 seconds of work, which is 9$ per hour.

**Annotation Agreement:** Multiple judgments are collected for each sentence. However, the number of annotators or judgments to be collected for each sentence is not fixed. This is due to the reason that Crowdflower doesn't simply decide the right label for a given sentence based on a majority count, instead the label with highest *confidence* is chosen, effectively returning the *most trusted judgment*, for a given unit of data.
Confidence is a score between 0 and 1, which describes the level of agreement between multiple contributors and the confidence in the validity of the result at the same time. For example, for two possible answers for each question in our task i.e. *suggestion* and *non-suggestion*, the confidence would be calculated as follows:

1. Sum the trust scores of the contributors responsible for each response. As explained previously, *Trust Score* is the accuracy score of a contributor on test questions.
   sum of trust(suggestion) = 4.47
   sum of trust (non-suggestion) = 1.80

2. Sum the trust scores for all responding contributors.
   Sum of trust(all) = 6.27

3. Divide each in #1 by #2 to find the confidence score for each response.
   Confidence(suggestion) = 0.713
   Confidence(non-suggestion) = 0.287
   Confidence scores for all the possible answers should sum to 1.

We provide a threshold confidence score (0.6), which one of the answers has to achieve in order to consider the sentence as successfully annotated. However, it can be the case

that a sentence is very ambiguous and cannot achieve the confidence score even after a large number of workers answered it. A maximum limit to the number of annotators is set in such case, and no further judgements are collected even if the threshold confidence is not reached. We set this limit to 5 workers in our case. Such sentences are counted as non-suggestions in the final dataset if the confidence score still remain to be below 0.6.

### 4.4.2 Phase 2 Annotations

As discussed in previous sections, we follow a 2 phase annotation strategy, where one phase includes the context and the other excludes the context of a sentence. Our scope of suggestions is limited to the sentences which are labeled as suggestion in both the phases, i.e. explicit suggestions. In the crowdsourced annotations, the context was provided to the annotators in the form of domain information and source text. The second phase of annotations is only required to be performed on the sentences which were labeled as suggestions in the previous phase. This drastically reduces the number of annotations to be performed in the second phase. The inter-annotator agreement was calculated by making two annotators label a subset of sentences for each domain (50 sentences). Cohen's kappa coefficient (see Chapter 2) was used to measure the inter-annotator agreement. Rest of the data instances were annotated by only one annotator.

Following guidelines were provided:

- The intention of giving a suggestion and the suggested action or recommended entity should be explicitly stated in the sentence. For example:[[Try]$_{intention}$[the cup cakes at the bakery next door]$_{entity}$]$_{action}$. Other explicit forms of this suggestion could be: *I recommend the cup cakes at the bakery next door*, or, *You should definitely taste the cup cakes from the bakery next door.* An implicit way of expressing the suggestion could be, *The cup cakes from the bakery next door were delicious.*
- The suggestion should have the intention of benefiting a stakeholder, and should not be mere sarcasm or joke. For example, *If the player doesn't work now, you can run it over with your car.*

Following are some of the scenarios of conflicting judgments observed in this phase of annotations:

1. In the case of suggestion forums for specific domains, like software developer forum, domain knowledge is required to distinguish an implicit non-suggestion from an explicit suggestion. For example the two sentences, 'It needs to be an integrated

part of the phones functionality, that is why I put it in Framework.', and, 'Secondly, you need to limit the number of apps that a publisher can submit with a particular key word.'. The first sentence is a description of already existing functionality and is a context sentence in the original suggestion post, while the second is suggestion for a new feature.

2. No concrete mention of what is being advised. 'It'd be great if you would work on a solution to improve the situation'.

3. 'I would go in fall.' This sentence was annotated as a suggestion in phase 1, as a part of travel discussion forum, with a likely interpretation as 'If I were you, I would go in fall'. However, when viewed without context, it can be perceived as reporting one's travel plans.

4. At times, there was a confusion between information (fact) or suggestion (opinion). For example, 'You can get a ticket that covers 6 of the National Gallery sites for only about US$10.'

## 4.5 Released Datasets

This section lists all the manually labeled datasets used in this thesis. In final versions of the datasets prepared by us, the sentences which are labeled as suggestions in the second phase of annotations are labeled as suggestions, while all other sentences are labeled as non-suggestions.

Following two datasets were available from the previous works. We consider the labels provided from the previous works as phase 1 annotations, and performed phase 2 annotations on these datasets, i.e. re-labeling of instances which were previously labeled as suggestions.

- **Travel forum dataset:** Wicaksono et al. [73, 74] crawled several web forum threads from two well-known travel forums InsightVacations[4] and Fodors [5]. Originally, an inter-annotator agreement of 0.76 (Cohen's kappa) was reported for this dataset. The provided dataset only comprises of the instances where both the annotators agreed.

- **Microsoft tweets dataset:** The dataset was initially released by Dong et al. [18]. While no inter-annotator agreement was reported by the authors, only those tweets were retained in the dataset where the annotators mutually agreed upon the label. In

[4] http://forums.insightvacations.com
[5] http://www.fodors.com

this case, the unit for suggestions is a tweet instead of a sentence. All of the tweets previously labeled as suggestions in the Microsoft tweet dataset were accepted as suggestions in phase 2 annotations. A 100% inter-annotator agreement was observed between the two annotators. This could be due to the fact that full tweet is available to the annotators rather than a single sentence, while hashtags also help in reducing ambiguities.

We prepare new datasets for the following domains, using the two phase method. This also includes a new dataset for the previously studied travel forum domain.

- **Hotel reviews**: Wachsmuth et al. [71] provide a large dataset of hotel reviews from the TripAdvisor[6] website. They segmented the reviews into statements so that each statement has only one sentiment label and have manually labeled the sentiments. Statements are equivalent to sentences, and comprise of one or more clauses. These statements have been manually tagged with *positive, negative, conflict, and neutral* sentiments. We take a smaller subset of these reviews, where each statement is considered as a sentence in our dataset. We also retain the the original review text and sentiment label for each sentence.

- **Electronics reviews:** Hu et al. [26] provide a dataset comprising of reviews of different kinds of electronic products obtained from the website of Amazon[7]. Amazon is an e-commerce company, and its website collects and displays online reviews of the listed products. Hu et al. also split the reviews into sentences, and sentiment for each sentence has been manually tagged.

- **Travel forum:** The data is obtained from the same source as the previous travel forum dataset by Wicaksono et al. This domain appears to exhibit a wide variety of expressions employed in suggestions, with relatively lower grammatical and spell errors. However, this domain also shows a relatively lower inter-annotator agreement.

- **Software suggestion forum:** The sentences for this dataset were scraped from the Uservoice[8] platform. Uservoice provides customer engagement tools to brands, and therefore hosts dedicated suggestion forums for certain products. Though most of the forums for commercial products are closed access, we discovered two forums which are openly accessible: Feedly mobile application, and Windows developer. A sample of posts were scraped and split into sentences using Stanford CoreNLP toolkit [32]. The class ratio in the dataset obtained from suggestion forums is more balanced than the other domains. Many suggestions are in the form of requests, which is less frequent in

[6] https://www.tripadvisor.com/
[7] https://www.amazon.com/
[8] https://www.uservoice.com/

| **Dataset identifier** | **Source** | **Class ratio (Suggestion : Non-Suggestion)** | **Inter-annotator agreement (phase 2)** |
|---|---|---|---|
| Existing datasets | | | |
| Microsoft tweets (original, re-tagged) | Twitter | 238/2762 ≈ 8:100 | 1.0 |
| Travel forum (original) | InsightVacations, Fodors | 2192/3007 ≈ 72:100 | 0.76 [73] |
| Travel train (re-tagged) | InsightVacations, Fodors | 1314/3869 ≈ 34:100 | 0.72 |
| New datasets | | | |
| Hotel train | Tripadvisor | 448/7086 ≈ 6:100 | 0.86 |
| Hotel test | Tripadvisor | 404/3000 ≈ 13:100 | 0.86 |
| Electronics train | Amazon | 324/3458 ≈ 9:100 | 0.83 |
| Electronics test | Amazon | 101/1070≈ 9:100 | 0.83 |
| Travel test | Fodors | 229/871≈26:100 | 0.72 |
| Software train | Uservoice suggestion forum | 1428/4296≈33:100 | 0.81 |
| Software test | Uservoice suggestion forum | 296/742≈39:100 | 0.81 |

TABLE 4.7: Manually annotated datasets prepared during this thesis. The datasets will be referred by their identifiers in the remaining chapters. The source text was openly available through different web platforms or existing publicly available datasets for sentiment analysis.

other domains. The text contains highly technical vocabulary related to the software which is being discussed, which may effect the classifier performance when this dataset is used for training or evaluation in the cross domain train-test settings, specially when bag of word features are employed. This will be validated in the next chapters.

Table 6.3 provides the details of these datasets. The datasets are selectively used in different experiments, based on the requirement of the experimental setup. Therefore, the experiments reported in the next chapters do not include all the available datasets as listed in this section. Some experiments only use a subset of a dataset, which will also be made publicly available in order to replicate the results.

## 4.6 Summary

In this chapter, we investigate a foundational question in suggestion mining, i.e., *What are suggestions in the context of suggestion mining?* To begin with, we perform a qualitative analysis of suggestions, which includes the study of two prominent linguistic properties of suggestions: lexical cues and grammatical moods. While most of the previous works heavily used the lexical cues as features for suggestion mining, our count of highly frequent unigrams and bigrams in suggestions miss many of these lexical cues.

While the frequent unigrams and bigrams appear to be heavily relying on the domain, grammatical moods tend to remain consistent across suggestions from different domains. Imperative and subjunctive mood are the two grammatical moods frequently observed in suggestions.

Sentiment analysis and suggestion mining are closely related areas of research in the sense that they are both applied to opinionated source texts, and are likely to be coupled for various applications. We study the relationship between sentiments and suggestions, and observe that suggestions do not always appear with a specific sentiment polarity. An interesting observation was that some of the sentiment analysis datasets which were sourced from online reviews choose to exclude those sentences which do not discuss the reviewed entity. Often, such sentences are advice and recommendations to the future customers on closely related points of interest.

We also report our study of the perception of the term *suggestion* among layman annotators. We map the sentences labeled as suggestions and non-suggestions by layman annotators to some predefined categories of expressions which tend to appear in opinion discourse. These categories were thoroughly studied and defined by an existing work [2]. Some of the categories of expressions were present in both suggestion and non-suggestion sentences, which forms the basis of ambiguities associated with the preparation of benchmark datasets. Based on the observations, we propose a context dependent method to define four categories of sentences encountered a the source text, explicit suggestions, implicit suggestions, explicit non-suggestions, an implicit non-suggestions. We also study the possible contexts which can play a significant role in determining whether a sentence should be regarded as a suggestion or not. We set the scope of suggestions to the *explicit* suggestions for this thesis and the unit of suggestion as sentences, and will refer to explicit suggestions as *suggestions* in the rest of the thesis. Based on this theory, we develop benchmark datasets using a two-phase annotation method. A detailed report is presented on the methodology adapted by us to prepare the benchmark datasets using a combination of crowdsourced and expert annotators. We introduce datasets which will be used throughout this thesis, and will be publicly available for research purposes.

A possible future direction with regard to the qualitative analysis of suggestions is to extend the study of suggestions to more domains, like, youtube comments, stack overflow discussions, blogs, etc.

# Chapter 5

# Manual Feature Engineering for Suggestion Mining

## 5.1 Introduction

This thesis approaches suggestion mining as a binary classification problem for sentences, based on if they contain explicit expressions of suggestions or not. This chapter investigates one of the central research question of a statistical classification task, i.e., *What are good features for suggestion mining?* In this chapter, we evaluate manually identified features for data representation in suggestion mining. Since features may perform variably with different classification algorithms, we employ Support Vectors Machines with a radial basis kernel [15] as the classification algorithm and experiment with employing different features with this classifier. We propose additional features to those which have been already proposed in the previous works, and present an extensive evaluation of all the features across different domains and datasets. These features represent the lexical, syntactic, semantic, and sentiment aspects of a text. Suggestions are observed to possess similar linguistic properties across domains. Therefore, feature evaluation is performed in both domain specific and domain independent training scenarios for the statistical classifiers, which has never been studied in the past with regard to suggestion mining.

This chapter addresses two aims. First is to provide insights into manual feature engineering for suggestion mining, which identify useful features as well as the limitations of manual feature engineering. Second is an in-depth evaluation of existing approaches for suggestion mining on the evaluation datasets which are labeled according to our proposed definition of suggestions. A rule based classifier is also used as a baseline in

order to evaluate the results obtained from the statistical classifiers. Early versions of this chapter have been published in [48, 49].

## 5.2 Features

The evaluated features are categorized into four classes, baseline features, related work features, proposed features, and a combination of the previous three.

### 5.2.1 Baseline features: n-grams

Unigrams, bigrams, and trigrams are considered as baseline features. All other features described below are used in combination with the n-gram features. The n-gram features are case insensitive, and do not exclude any stop words.

### 5.2.2 Related work features

The two main line of works which performed statistical suggestion mining were, Wicaksono and Myaeng [72–74] and Dong et. al [18]. Unigrams were used as features by all of these works, which we have already included in our baseline classifiers. The other features employed by these works can be categorised into domain independent and domain specific features. We use an identifier $R_n$, where $n$ is a numeric id, for each of the features proposed by these related works and which we re-implement for our experiments.

**Domain independent features**

These are the features whose value can be calculated independent of the domain, and which do not rely on any domain specific property of the data. However, not all of these features could be re-implemented by us due to unavailability of implementation details in the papers. Following are the features which could be re-implemented based on the available information:

1. **Template features ($R_1$):** These features were proposed by Dong et. al [18], who performed suggestion mining using a training dataset of tweets which were addressed to Microsoft, and which discussed Windows phone. The features include automatically extracted indicator words and phrases, which they refer to as *suggestion templates*. These templates were extracted from the feedback data crawled from Windows phone's official feedback web site. With an aim to extract domain-independent templates from these feedbacks, they first filtered out the domain-related words (e.g.

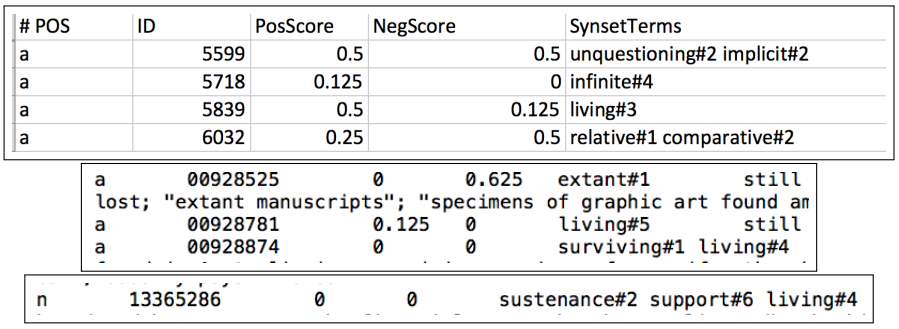
| Group# | Templates | Group# | Templates |
|---|---|---|---|
| 1 | need, require, want | 6 | wish, would, should, ‘d |
| 2 | dear, please, come on | 7 | ‘d better, had better, couldn’t be more, couldn‘t be better |
| 3 | i would like, i’d like, i would love, ’d love, love to see | 8 | we want, we don’t want, we need, we don’t need, i want, i don’t want, i need, i don’t need, want to, wants to |
| 4 | i hope, i don’t hope, we hope, we don’t hope, i wish, i don’t wish, we wish, we don’t wish, wish to, hope to | 9 | how about, what about, what if, why not, why couldn’t, why can’t, why wouldn’t |
| 5 | need to, needs to, require to, requires to | 10 | might help, may help, i have an idea |

TABLE 5.1: Template features from Dong et. al [18]

“windows”, “phone” and “lumia”) from the feedback data. They employed the PrefixSpan (Prefix-projected Sequential Pattern mining) [55] algorithm which is used to find frequent sequential patterns in sequential data. They removed the patterns which only contained stop words. To overcome sparsity in the feature vector, they did not use only one template as one feature, instead they grouped a number of templates as a single feature. This grouping was performed manually, by selecting templates which only differed in interchangeable semantically similar words, like “I would like” and “I would love”. They selected 45 highest frequency templates and grouped them as ten different features (Table 5.1). The feedback dataset and their implementation of the template extraction algorithm is not available and the templates are not listed in their paper, although the templates were directly provided to us from the authors upon request. These are implemented by us as ten different boolean valued features.

2. **Proper noun and modal verb ($\mathbf{R}_2$):** This features was proposed by Wicaksono and Myaeng [72]. It is a string representing the combination of a proper noun and modal verb which appear in a sentence as the arguments of a ‘nominal subject’ (*nsubj*) dependency relation. A nominal subject is a noun phrase which is the syntactic subject of a clause. Stanford dependency parser [17] is used to identify the linguistic dependencies in a sentence. Each pair of such proper nouns and modal verbs appearing in the data are treated as a single feature.

3. **Presence of imperative mood ($\mathbf{R}_3$):** This feature was used by Wicaksono and Myaeng [72, 73] as a boolean valued feature whose value depends on the presence of imperative mood in the sentence. They proposed that if any one of the following conditions holds true, the imperative mood is present in a sentence:



| # POS | ID | PosScore | NegScore | SynsetTerms |
|---|---|---|---|---|
| a | 5599 | 0.5 | 0.5 | unquestioning#2 implicit#2 |
| a | 5718 | 0.125 | 0 | infinite#4 |
| a | 5839 | 0.5 | 0.125 | living#3 |
| a | 6032 | 0.25 | 0.5 | relative#1 comparative#2 |

```
a       00928525        0       0.625   extant#1        still
lost; "extant manuscripts"; "specimens of graphic art found am
a       00928781        0.125   0       living#5        still
a       00928874        0       0       surviving#1 living#4
```

```
n       13365286        0       0       sustenance#2 support#6 living#4
```

FIGURE 5.1: Screenshots from SentiWordNet. The field *Synset Terms* comprises of words and their synset id from Wordnet separated by a # symbol. Word *living* appears in different sentiment synsets.

(a) First word's POS tag is either VB (verb, base form) or VBP (verb, non-3rd person singular present).

(b) First word's POS tag is RB (adverb) and second word's POS tag is either VB or VBP.

(c) Starting from any word position whose POS tag is either SYM (symbol) or punctuation mark, if #1 and #2 apply.

4. **Specific linguistic dependencies ($\mathbf{R}_4$):** Proposed by Wicaksono and Myaeng [73], this is again a boolean valued feature depending on the presence or absence of any of the typed dependencies, 'conj', 'csubj', and 'nsubj' in the sentence.

We were unable to implement the following domain independent features because of the unavailability of required details:

1. **Opinion copula:** Proposed by Wicaksono and Myaeng [72], this boolean valued feature indicates the presence of an opinionated copula, where a copula is the relation between the complement of a copular verb and the copular verb (is, am, are). In an opinionated copula, the governor of the relation should have a subjectivity score above the threshold 0.69. The subjectivity score of a word was determined using SentiWordNet [4]. The screenshots from SentiWordNet shown in Figure 5.1 show that there are two types of scores provided for each sense of a word in SentiWordnet. There could be multiple senses of the same word, with a different set of values corresponding to the positive and negative scores. It was not mentioned how exactly a single subjectivity score was obtained, given SentiWordNet's multiple senses and different values of positive and negative scores for each sense.

2. **Clue verbs:** This feature was proposed by Wicaksono and Myaeng [72]. It is a function of three elements: a set of clue verbs found in the sentence, proper-noun

attached to the clue verbs as nominal subject, and POS tags of those clue verbs. In order to obtain clue verbs, they used some seed verbs and expanded them using Framenet [6]. However, these seed verbs or the final expanded set of verbs are unavailable in order to replicate this feature.

**Domain dependent features**

Apart from the above described features, domain specific features were also employed in the previous works, as each work only addressed one domain. In the case of discussion forums, features which where dependent on the structure of the discussion thread were employed [73, 74], for example, the position of a given sentence in the thread, and hashtag based features were proposed for suggestion mining from tweets [18]. We do not re-implement and compare the performance of domain specific features, since these are not applicable to the training datasets from different domains, as well as are not suitable for domain independent training of the classifiers.

### 5.2.3 Proposed Features

These are the new features proposed by us, which are either based on our observations, or are variations of some features from the previous works which could not be replicated. We refer to these features through an identifier $P_n$ for each of the feature types, where $n$ is a numerical id. Similar to the related work features, we propose to use these features in addition to unigram features, which is a popular baseline. Below, we group the new features into lexical, syntactic, sentiment, and semantic features.

**Lexical features:** These are word level features, which depend on the vocabulary

| |
|---|
| hope, desire, trust, believe, want,require, advice, warning, advise, suggest, recommend, recommendation, insist, demand, necessity, requirement, advise, tip, ask, request, urge, expect, propose, proposal, think, dream, imagine, covet, wish, beg, crave, fancy, implore, need, press, avoid, refrain, do not, dont́, do, warn, caution, desirable, suitable, important, imperative, urgent, beneficial, effective, necessary, essential |

TABLE 5.2: The extended set of suggestion keywords from proposed features ($P_1$)

appearing in the data. Unigrams also belong to this category of features.

1. **Suggestion keywords ($P_1$):** These include the verbs: *suggestion, advise, request, ask, warn, recommend, do, do not*; their corresponding nouns: *suggestion, advice, request, warning, tip, recommendation*, and the synonyms of this set of verbs and nouns, obtained using WordNet [21]. Suggestion keywords constitute a single feature, which is boolean valued, and its value is determined by the presence of any of the suggestion keywords in the sentence.

**Syntactic features:** These features represent the grammatical properties of sentences.

1. **Part of Speech n-grams ($P_2$)**: These features are similar to the word n-gram features, with the only difference that words are replaced by their respective part of speech tags. Consider an example of a PoS tagged sentence, 'For_IN a_DT lovely_JJ breakfast_NN ,_, turn_VB left_VBN out_IN of_IN the_DT front_NN entrance_NN -_: on_IN the_DT next_JJ corner_NN is_VBZ a_DT cafe_JJ with_IN fresh_JJ baked_JJ breads_NNS and_CC cooked_JJ meals_NNS ._.'. The POS tag bigram corresponding to the bigram 'For a' would be 'IN DT'. POS n-gram features include unigrams, bigrams, and trigrams of POS tags.

2. **Sequential patterns of POS tags:** Sequential pattern mining are methods which find statistically relevant patterns between data examples where the data points are present in a sequence. Sequential pattern mining was traditionally applied to applications like mining purchase patterns of customers, where ordering of products on shelves can be based on the order of mined purchasing patterns. State of the art sequential pattern mining algorithms require the dataset to be converted into an ordered list of *events*. An event is a non-empty unordered collection of *items*, which in turn is the smallest unit of sequences. The output of these algorithms are sequential patterns, which comprise of sequences of items, and is defined to be frequent if its support (a measure of frequency) is equal or more than a user-defined threshold.
   Consider $(a_1, a_2, ..., a_q)$ a sequence, where $a_i$ is an event. A sequence $(a_1, a_2, ..., a_n)$, is a subsequence of another sequence $(b_1, b_2, ...b_m)$, if there exist integers $i_1 < i_2 < ... < i_n$ such that $a_1 \subseteq b_{i1}$, $a_2 \subseteq b_{i2}, ..., a_n \subseteq b_{in}$. For example, given a sequence (AB,E,ACD), where B is an item and AB, E and ACD are events. (B,AC) is a subsequence, since B $\subseteq$ AB and AC $\subseteq$ ACD and the order of events is preserved. On the other hand, (AB,E) will not be a subsequence of the sequence (ABE). We map our pos tag sequences to sequential pattern mining by considering each POS tag as an event, which means that the length of all events is 1 item.
   Several algorithms exist for pattern mining. We use the Clospan [76] algorithm to obtain the frequent patterns of PoS tags appearing in imperative mood sentences. Closed pattern mining algorithms are believed to produce a significantly less number of sequences than the previously introduced algorithms, while preserving the same expressive power. We use sequences of lengths 2 and 3 with a relative support greater than 0.4, where maximum relative support is 1. Features ($P_3$) and ($P_4$) explained below use sequence pattern mining. In Chapter 4, we showed that suggestions tend to contain imperative and subjunctive mood, therefore we perform pattern mining on sentences containing imperative and subjunctive mood, and use these patterns to indirectly identify if the sentence contains these moods.

| Imperative Patterns | Subjunctive Patterns |
|---|---|
| $< NN >< NN >, < VB >< NN >$ | $< NN >< NN >,$ |
| $< VB >< PRP >, < VB >< VB >,$ | $< VB >< NN >,$ |
| $< VB >< VB >< NN >$ | $< NN >< VB >$ |

TABLE 5.3: Frequent POS patterns observed in sentences containing imperative and subjunctive mood

**Sequential patterns vs. n-grams:** Sequential patterns are different from the POS n-gram features in the sense that sequential patterns are like templates with place-holders, where the items in a pattern are ordered but do not necessarily appear immediately next to each other, while items in n-grams are immediate neighbors.

**Imperative Mood Sequential Patterns ($\mathbf{P}_3$):** Wicaksono et al. [72] also observed the presence of imperative mood in advice sentences. They used an imperative mood detector, based on a set of handmade rules to determine whether a sentence is imperative. PoS sequential pattern extraction is our alternative to such rule based mood detection. The sentences often contain more than one clause, and more than one mood expression. This feature aims to detect if any part of a given sentence contains at-least one expression of imperative mood. In order to extract the patterns, we prepare a small dataset of 300 example sentences of imperative mood. These sentences were short sentences of lengths between 3-8 words, and are manually collected from webpages related to English grammar, and linguistic research papers on mood and modality.

**Subjunctive Mood Sequential Pattern ($\mathbf{P}_4$):** None of the previous works emphasized the presence of subjunctive mood in the suggestion sentences. Based on our observation, we extract POS sequence features from a dataset of sentences containing the subjunctive mood. Similar to the imperative mood dataset, this dataset also contains 300 sentences, and is collected from English grammar webpages, and linguistic research papers on mood and modality.

Each of the two feature types ($P_3$, $P_4$) comprise of a number of patterns, each of which represents a single boolean valued feature depending on the presence or absence of the pattern in a given sentence. A total of 8 features (see Table 5.3) were extracted for both the moods.

3. **Information about the subject/s ($\mathbf{P}_5$):** This feature captures the details of *nsubj* dependency appearing in a sentence. It comprises of pairs of POS tags of the arguments of the nsubj dependency, or the label *no_nsubj* in case nsubj dependency does not appear. This feature is based on the observation that often a reviewer addresses

the reader when giving a suggestion. In the sentence, 'If you do end up here, be sure to specify a room at the back of the hotel', the nsubj dependency is nsubj(do, you), and the feature in this case would be *VBP-PRP*. On the other hand, this suggestion could also appear in a shorter form, 'Be sure to specify a room at the back of the hotel.', in which case the feature value would be *no_nsubj*. If more than one nsubj dependency is present, the POS pair for each of them will be included in the feature set.

**Sentiment features:** Given that suggestions are mostly extracted from opinion and feedback containing texts, which are also used for sentiment analysis studies, there may be a co-relation between sentiments and suggestions. In Chapter 4, we saw that suggestions or non-suggestions in reviews do not belong to any particular category of sentence level sentiment targeted towards the entity being reviewed. Also, it is a complex natural language processing task on its own to determine the human intended sentiment of a given sentence. Therefore, our sentiment features do not involve the estimation of a sentiment label for the full sentence, but a numerical aggregation of the amount of positive and negative sentiment associated with each word of a sentence, which may/may not be a direct indicator of the actual sentiment.
Previous works also used some rule based sentiment features, where numeric sentiment values were calculated using SentiWordNet, but only for words appearing at certain places or which have a relationship to other pre-determined words. We employ the following two sentiment related features:

1. **SentiWordNet score sum ($\mathbf{P_6}$):** A sum of sentiment scores of all the words in a sentence is obtained using SentiWordNet 3.0 [4, 20]. We already saw in Figure 5.1 how each word can appear in SentiWordNet among different sentiment synsets representing different senses of a word. Even on restricting a word to a particular part of speech, multiple sentiment synsets may exist. Different senses of a word may have a distinct sentiment score pair (positive, negative). We calculate a single sentiment value for each word using the formula:
   $sentiment_w = \sum(1/n)(PosSentiment(w_n^p) - NegSentment(w_n^p))$
   where $sentiment_w$ is a single sentiment score for each word w, $w_n^p$ is a sense of the word which appears as a p part of speech in the sentence, n is the sense rank of the word which is attached with the word using a # symbol. For example, 1,2,3 are sense ranks of the word *living* in the senses living#1, living#2, living#3 respectively. Thus, a single sentiment score for a word is a weighted sum of the difference in its positive and negative scores from each sense's sentiment. This method of calculating a single sentiment value for each word is adapted from

the sample algorithm mentioned on SentiWordNet's homepage [1]. Finally, the sentiment score for the whole sentence is the sum of sentiment scores of individual words.

2. **SentiWordNet score normalized sum ($P_7$):** This is the sentiment score sum as calculated above, but normalised over the number of words in the sentence.

**Semantic Features:** These features relate to the meaning of the text.

- **Named Entities ($P_8$):** This feature represents the presence and type of any named entity in the sentence. This is based on the observation that suggestions often contain mention of named entities. For example, 'I would have 2 nights in Killarney as a touring base and drive the Dingle figure of eight on a long drive between Killarney and Ennis.' Stanford named entity recognizer [2] is used for the same [23]. The *Stanford NER* model is used, which is a 7 class NER model, where the classes are: Location, Person, Organization, Money, Percent, Date, Time.

## 5.3 Rule Based Classifier

An intuition for suggestion mining is that a set of hand picked rules should work well for identifying suggestions, which was the approach adopted in some of the previous studies. Therefore, we also evaluate a rule based classifier for the same, in order to compare its performance with a statistical classifier. This classifier comprises of an aggregation of rules suggested in the existing works on rule based suggestion mining. These rules comprise of certain keywords, phrases, and lexico-syntactic patterns which in turn comprise of certain keywords/phrases and patterns in which these words appear in suggestion sentences within a particular domain. The patterns also contain placeholders for domain related entities or expressions. For example, the keywords/phrases 'suggest' and 'I wish', are used in combination with product specific terms in the following patterns from Viswanathan et al. [70]: ⟨*I wish* (Product) *had/contain/include* (Feature) ⟩, ⟨*Suggest* (Product) *for* (Feature) ⟩.

Since we are currently evaluating the classifiers for multiple domains, which can be used or trained likewise on datasets across different domains, these patterns from the existing works cannot be directly used for our rule based system. Therefore, we only collect the domain independent lexical cues from these patterns and form a set of keywords and phrases. In addition, there are some existing syntactic patterns which do not

[1] http://sentiwordnet.isti.cnr.it/code/SentiWordNetDemoCode.java
[2] https://nlp.stanford.edu/software/CRF-NER.shtml

| Related Work | Rules |
|---|---|
| **Keywords and phrases** | |
| Ramananed et al.[60] | needs to, need to |
| Viswanathan et al. [70] | suggest, recommend, if, i wish, go for,should have, would, could have been |
| Viswanathan et al. [70] | i would like, i'd like, i would love, I'd love, love to see |
| Moghaddam [43] | there should be, I wish, allow us to |
| **Syntactic patterns** | |
| Ramananed et al.[60] | Modal verbs followed by another verb, for example: I would prefer the unit to have a simple on off switch. |
| Moghaddam [43] | there should be (DT), VB (DT) option, I cannot (VB), MD be (ADV) ADJ, MD (like/prefer/love) to, stop VBG, ability(to—of) |

TABLE 5.4: Rules for the rule based classifier

comprise of domain specific vocabulary. The rule based classifier then simply checks the presence of at-least one of the lexical cues or syntactic patterns in order to identify a sentence as a suggestion. All the rules are listed in Table 5.4.

## 5.4 Classification algorithm

Given that the task at hand is the binary classification of sentences, in this chapter, we use Support Vector Machines (SVM) classifier with a Radial Basis Kernel. SVMs have been widely accepted as a good choice of classification algorithm for text classification and binary classification [13, 28, 31], since it often performs comparable or better than its counterparts. It is also known to require a relatively small amount of parameter tuning for parameters like number of features [28]. In addition, all the related works on statistical suggestion mining used Support Vector Machines [18, 43, 72], excluding the ones which approached suggestion mining as a sequence classification task. Wicaksono and Myaeng [73, 74] used HMM and CRF classifiers when they considered posts in discussion forums as sequences and labeled each sentence as an item in the sequence.

**Classifier configuration:** We choose classifier parameters using 5 fold cross validation of training datasets. The two critical parameters associated with SVMs are C and $\gamma$. Various studies have provided certain range of values for C and $\gamma$ (see Chapter 2) for best classification performances [3] [28]. Based on these studies and several runs conducted by us with different C and $\gamma$ values, we use the values 1 and 0.01 respectively. Another configuration parameter is the number of features to be used. We perform a comparison of different no. of unigram features to be used in a cross validation setting, and observe that there is no significant change in the F-score of the positive class, when the number

[3] http://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html

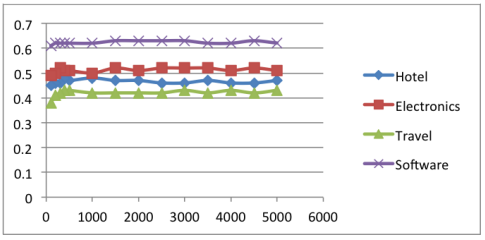



FIGURE 5.2: Performance comparison on varying the no. of unigram features to be used with the classifier

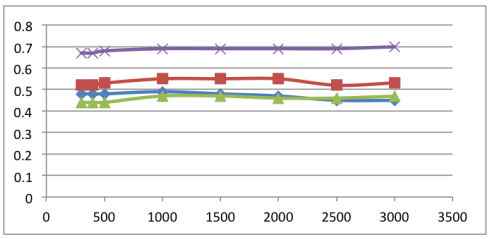



FIGURE 5.3: Performance comparison on varying the no. of features used by the classifiers when all kinds of features are used.

of unigram features are above 500 (Figure 5.2). We also perform the same test when all the features (n-grams, proposed, related) are used (Figure 5.3), and observe that feature dimension between 1000-2000 features gives best F1 score across all the domains. Therefore, we use 1000 dimensions for all the models using the Information Gain (Infogain) [77] feature selection method. Infogain is one of the widely used feature selection methods which can also be used to rank the features. Infogain value of a feature is the measure of reduction in entropy of the class variable, after the value for a feature is observed. The information gain value of each feature is calculated as follows:

Entropy:
H(i) $= -p_i \log p_i$
$p_i$ is the probability of $i^{th}$ class, i.e. the proportion of $i^{th}$ class in the dataset.

$$H = -\sum_{n=1}^{i} p_i \log p_i$$

Information gain or change in entropy from a feature f:
I(f)= H - H(i|f)

Class weight is another important parameter, which we set to 1: 5 in all the experiments. This parameter is used as an alternative to oversampling. Since SVMs work by finding the support vector instances from both the classes, performing oversampling would duplicate the instances which won't bring any change to the choice of support vectors (see Chapter 2).

## 5.5 Evaluation

The purpose of the set of experiments and evaluation presented in this chapter is to determine which of the manually defined features are effective for the underlying classification task in suggestion mining, and if they perform equally well when domain specific training data is unavailable or training data is a mix of datasets from different domains. Before proceeding to the results section, we present the following aspects of evaluation performed in this chapter.

**Evaluation metrics:** In this thesis, classification performance of a model has been reported as the Precision, Recall, and F-1 score for the positive class, i.e. the *suggestion* class. The best value for all of these measures is 1 and the worst is 0. In some of the real life applications, only one out of precision and recall for the positive class may be preferred in order to compare different models, and not the F-1 score. This is due to the fact that some applications may find it useful to retrieve all suggestions even though some non-suggestions are also identified as suggestions, than to miss some suggestions; while for other applications vice versa may be required.

| | | Actual Label | |
|---|---|---|---|
| | | **Suggestion** | **Non-Suggestion** |
| Predicted Label | **Suggestion** | True Positives | False Negatives |
| | **Non-Suggestion** | False Negatives | True Negatives |

Precision suggestion ($P_{sugg}$): It measures the fraction of instances which are actually suggestions out of the ones which are predicted as suggestions.
$P_{sugg}$ = True Positives / (True Positives + False Positives)

Recall suggestion ($R_{sugg}$): It measures the fraction of suggestion class instances which are correctly identified out of the total number of suggestions.
$R_{sugg}$ = True Positives / (True Positives + False Negatives)

F1 score suggestion ($F1_{sugg}$): It combines precision and recall of the suggestion class by calculating their harmonic mean.
$F1_{sugg} = 2 * (P_{sugg} * R_{sugg}) / (P_{sugg} + R_{sugg})$

**Evaluation data:** A test set is used for evaluation for each of the four domains, hotel reviews, electronics reviews, travel discussion forum, and software developer feedback forum. As discussed before, suggestion datasets have a skewed class distribution in all the studied domains, and therefore these test sets were originally unbalanced. However, for the evaluation we use a balanced version prepared by random under-sampling of

| Dataset | sugg./non-sugg. | Dataset | sugg./non-sugg. |
|---|---|---|---|
| **Training datasets** | | **Test datasets** | |
| Hotel train | 448/7086 | Hotel test | 404/404 |
| Electronics train | 324/3458 | Electronics test | 101/101 |
| Travel train | 1314/3869 | Travel test | 229/229 |
| Software train | 1428/4296 | Software test | 296/296 |
| **Related work datasets** | | | |
| Microsoft Tweets | 238/2762 | NA | NA |
| Travel-original | 2192/3007 | NA | NA |

TABLE 5.5: Datasets used in this chapter

negative instances, while counter methods for class imbalance are applied during the training of models. This is to ensure that results for cross domain and open domain training are not affected by different class distributions in training and test datasets. Table 5.5 provides an overview of the datasets used in this chapter.
As we already discussed, in the absence of a development dataset, we performed parameter optimisation using a 5 fold cross validation on the training datasets. In next chapters we deal with much heavier and complex neural network based training methods, for which cross validation is a much more computationally expensive method for evaluation. Therefore, separate train and test datasets suit better for our experimental setup, where the main objective is to observe the relative performance of different kinds of features.

**Open domain and cross domain evaluation:** We refer to using training datasets from multiple domains including the evaluation domain as *open domain*, and using training datasets from domains other than the evaluation domain as the *cross domain* training scenario. Keeping in mind that it would be expensive to manually label training datasets for each application domain, we investigate the effectiveness of features in *open domain* and *cross domain* training scenarios as well.

**Comparison with the related work:** As mentioned before in the Related Work chapter, not all the previous datasets are available, and not all the features proposed in the previous works are replicable, either due to the missing details or dependency on domain specific metadata. However, two of the three most relevant works on statistical suggestion mining have made their dataset publicly available (Dong et al. [18], and Wicaksono and Myaeng [73, 74]). Therefore, we compare our best performing feature set with the results obtained by these works, where they have also employed SVM classifiers with a set of hand picked features. Comparison is performed by using their version of training dataset (and not the one re-tagged by us), and evaluating the classifier through a 5-fold cross validation method (Table 5.11), which is the same as followed in both of these works. The parameter values for SVM remain the same as used in rest of the

| **Features/Classifier** | **Hotel Test** | | | **Electronics Test** | | |
|---|---|---|---|---|---|---|
| | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ |
| **Baseline** | | | | | | |
| Unigrams | 0.974 | 0.653 | 0.782 | 0.938 | 0.612 | 0.741 |
| Uni,Bi-grams | 0.996 | 0.589 | 0.74 | 0.957 | 0.459 | 0.621 |
| Uni, Bi, Tri-grams | 0.992 | 0.597 | 0.745 | 0.978 | 0.449 | 0.615 |
| **Related Work** | | | | | | |
| Rule based classifier | 0.790 | 0.475 | 0.593 | 0.802 | 0.564 | 0.662 |
| Unigrams, $R_1$-$R_4$ | 0.982 | 0.686 | 0.808 | 0.928 | 0.653 | **0.766** |
| **Proposed** | | | | | | |
| Unigrams,$P_1$-$P_8$ | 0.986 | 0.686 | 0.809 | 0.878 | 0.663 | 0.756 |
| **All** | | | | | | |
| Unigrams,$P_1$-$P_8$,$R_1$-$R_4$ | 0.989 | 0.693 | **0.815** | 0.877 | 0.653 | 0.749 |

TABLE 5.6: Performance of different features on hotel and electronics test datasets when the classifiers are trained on the same domain training data.

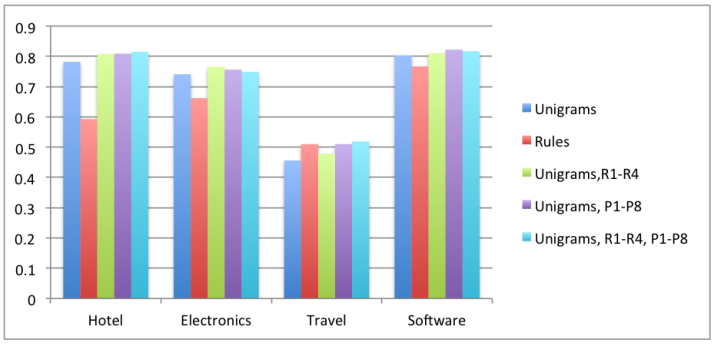



FIGURE 5.4: Comparison of the performance of different feature sets using when training and test datasets belong to the same domain

experiments.

## 5.6 Results and Analysis

The results presented in this section compare the performance of the classifier with different sets of features across all four domains. We test the baseline features in three settings, using only unigrams, using unigrams and bigrams, and using unigrams, bigrams and trigrams. It was observed that using only unigrams performed best with the majority of datasets. The second category of features comprises of features collected from the related work, which also include unigrams in addition to the manually chosen domain independent features. The third category of features are a combination of unigrams and the features proposed by us. Finally, we also present the classification results of using unigrams, related work features, and the proposed features, which tend to be the best setting with most of the domains. All the experiments are repeated for the three different training scenarios, domain specific, open domain, and cross domain.

**In-domain training:** Tables 5.6, and 5.7 show the classification results when a

| Features/Classifier | Travel Test | | | Software Test | | |
|---|---|---|---|---|---|---|
| | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ |
| **Baseline** | | | | | | |
| Unigrams | 0.505 | 0.416 | 0.456 | 0.769 | 0.84 | 0.803 |
| Uni,Bi-grams | 0.56 | 0.412 | 0.474 | 0.803 | 0.805 | 0.804 |
| Uni, Bi, Tri-grams | 0.551 | 0.429 | 0.483 | 0.799 | 0.799 | 0.799 |
| **Related Work** | | | | | | |
| Rule based classifier | 0.799 | 0.375 | 0.510 | 0.686 | 0.871 | 0.767 |
| Unigrams, $R_1$-$R_4$ | 0.536 | 0.434 | 0.479 | 0.776 | 0.85 | 0.811 |
| **Proposed** | | | | | | |
| Unigrams,$P_1$-$P_8$ | 0.558 | 0.469 | **0.510** | 0.748 | 0.911 | **0.822** |
| **All** | | | | | | |
| Unigrams,$P_1$-$P_8$,$R_1$-$R_4$ | 0.568 | 0.478 | **0.519** | 0.765 | 0.877 | 0.817 |

TABLE 5.7: Performance of different features on travel and software test datasets when the classifiers are trained on the same domain training data

classifier is trained and evaluated on the datasets belonging to the same domain. Figure 5.4 presents a comparative view of the F1 score produced by different feature sets across the datasets. It is observed that in 3 out of 4 datasets, inclusion of proposed features in the feature set resulted in an improvement in the F1 score for the suggestion class. In the case of Electronics domain, related work feature set gave the best performance, and the proposed feature set gave the second best. Overall, the proposed and related work features were close in their performances, except in the case of Travel domain, where proposed features performed significantly better. The classification accuracy remains quite low for the travel domain in comparison to the other domains. Only software domain has higher recall than precision with all types of features.

**Open-domain training:** Tables 5.8 and 5.8 show the classification results when

| Features/ Classifier | Hotel Test | | | Electronics Test | | |
|---|---|---|---|---|---|---|
| | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ |
| **Baseline** | | | | | | |
| unigrams | 0.955 | 0.78 | 0.858 | 0.874 | 0.847 | 0.86 |
| unigrams, bigrams | 0.965 | 0.752 | 0.846 | 0.847 | 0.847 | 0.847 |
| unigrams, bigrams, trigrams | 0.968 | 0.748 | 0.844 | 0.856 | 0.847 | 0.851 |
| **Related Work** | | | | | | |
| Rule based classifier | 0.790 | 0.475 | 0.593 | 0.802 | 0.564 | 0.662 |
| unigrams, $R_1 - R_4$ | 0.964 | 0.785 | 0.865 | 0.881 | 0.908 | **0.894** |
| **Proposed** | | | | | | |
| unigrams, $P_1 - P_8$ | 0.96 | 0.824 | 0.887 | 0.832 | 0.908 | 0.868 |
| **All** | | | | | | |
| unigrams, $R_1 - R_4$, $P_1 - P_8$ | 0.963 | 0.829 | **0.891** | 0.856 | 0.908 | 0.881 |

TABLE 5.8: Performance of different feature sets in an open domain training scenario evaluated on hotel and electronics test datasets

a classifier is trained on a pool of all the available datasets, which also includes the dataset belonging to the same domain as the test dataset. Which means that the training dataset remains same for all the test sets in this set of experiments. Figure 5.5 presents a comparative view of the F1 score obtained through different feature sets. In hotel and travel dataset, proposed features bring better improvement over the baseline,

| Features/ Classifier | Travel Test | | | Software Test | | |
|---|---|---|---|---|---|---|
| | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ |
| **Baseline** | | | | | | |
| unigrams | 0.73 | 0.372 | 0.493 | 0.717 | 0.829 | 0.769 |
| unigrams, bigrams | 0.748 | 0.341 | 0.468 | 0.742 | 0.795 | 0.768 |
| unigrams, bigrams, trigrams | 0.757 | 0.345 | 0.474 | 0.739 | 0.792 | 0.764 |
| **Related Work** | | | | | | |
| Rule based classifier | 0.799 | 0.375 | 0.510 | 0.686 | 0.871 | 0.767 |
| unigrams, $R_1 - R_4$ | 0.746 | 0.416 | 0.534 | 0.733 | 0.87 | **0.796** |
| **Proposed** | | | | | | |
| unigrams, $P_1 - P_8$ | 0.748 | 0.46 | **0.57** | 0.70 | 0.877 | 0.779 |
| **All** | | | | | | |
| unigrams, $R_1 - R_4$, $P_1 - P_8$ | 0.754 | 0.46 | **0.571** | 0.711 | 0.874 | 0.784 |

TABLE 5.8: Performance of different feature sets in an open domain training scenario evaluated on travel and software test datasets

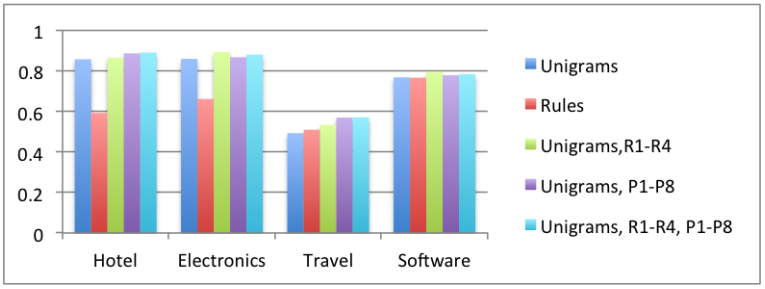



FIGURE 5.5: Comparison of the performance of different features sets in an open domain training scenario

| Features/ Classifier | Hotel Test | | | Electronics Test | | |
|---|---|---|---|---|---|---|
| | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ |
| **Baseline** | | | | | | |
| unigrams | 0.694 | 0.928 | 0.794 | 0.814 | 0.714 | 0.761 |
| unigrams, bigrams | 0.729 | 0.886 | 0.8 | 0.866 | 0.724 | 0.761 |
| unigrams, bigrams, trigrams | 0.753 | 0.896 | 0.818 | 0.886 | 0.714 | 0.791 |
| **Related work** | | | | | | |
| Rule based classifier | 0.790 | 0.475 | 0.593 | 0.802 | 0.564 | 0.662 |
| unigrams, $R_1 - R_4$ | 0.753 | 0.891 | 0.816 | 0.844 | 0.776 | 0.809 |
| **Proposed** | | | | | | |
| unigrams, $P_1 - P_8$ | 0.773 | 0.894 | 0.829 | 0.833 | 0.816 | 0.825 |
| **All** | | | | | | |
| unigrams, $R_1 - R_4$, $P_1 - P_8$ | 0.785 | 0.903 | **0.84** | 0.871 | 0.827 | **0.848** |

TABLE 5.9: Performance of different models using a combination of training datasets which excludes the test domain.

while in electronics and software datasets related work features bring more improvement over the baseline features. An interesting observation is that electronics and software also show better recall than precision, which may indicate that the related work features bring better recall. Overall, in all the datasets, using all features results in the highest or second highest F score. Classifier performance improves for all the domains except software when open domain training dataset is used.

**Cross-domain training:** Tables 5.9 and 5.10 show the results of when the train-

| Features/ Classifier | Travel Test | | | Software Test | | |
|---|---|---|---|---|---|---|
| | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ |
| **Baseline** | | | | | | |
| unigrams | 0.774 | 0.288 | 0.419 | 0.643 | 0.253 | 0.363 |
| unigrams, bigrams | 0.784 | 0.257 | 0.387 | 0.651 | 0.235 | 0.346 |
| unigrams, bigrams, trigrams | 0.789 | 0.248 | 0.377 | 0.63 | 0.215 | 0.321 |
| **Related work** | | | | | | |
| Rule based classifier | 0.799 | 0.375 | 0.510 | 0.686 | 0.871 | **0.767** |
| unigrams, $R_1 - R_4$ | 0.785 | 0.323 | 0.458 | 0.708 | 0.413 | 0.522 |
| **Proposed** | | | | | | |
| unigrams, $P_1 - P_8$ | 0.798 | 0.367 | 0.503 | 0.701 | 0.512 | 0.592 |
| **All** | | | | | | |
| unigrams, $R_1 - R_4$, $P_1 - P_8$ | 0.796 | 0.381 | **0.515** | 0.725 | 0.577 | 0.643 |

TABLE 5.10: Performance of different models using a combination of training datasets which excludes the test domain.

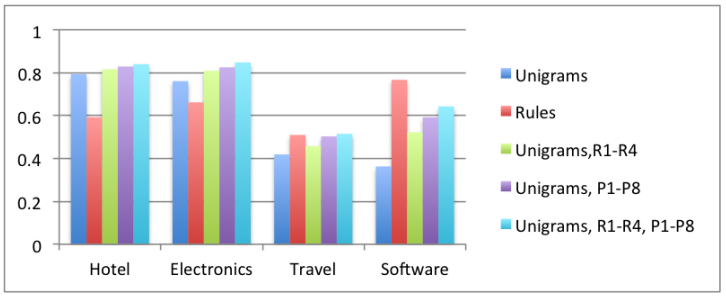



FIGURE 5.6: Comparison of the performance of different features sets in a cross domain training scenario

ing data excludes the test domain, and includes all other available domains. Figure 5.6 shows a comparative overview of Fscores across different datasets using different feature sets in a cross domain training scenario. The proposed features perform better than the related work features for all the datasets, and the best F score was produced when all the features were used. Interestingly, in the case of software dataset, the rule based classifier outperforms the best performing statistical classifier i.e the one which uses all the features. Also, rules bring higher recall than precision in this case. Again, the classifier performance improves in this setting as compared to the domain specific training, except for the software domain.

**Comparison with the previous works:** As discussed previously, only two of the related works made their datasets available (Wicaksono and Myaeng [72–74], and Dong et al. [18]). The classifiers employed by these systems are not completely replicable on all the datasets due to the previously stated reasons. Therefore, we perform a comparison of our features with the results produced by them by using the datasets provided by these works (Table 5.11). Both of these works employed SVM in addition to other classifiers like HMM and Factorisation Machines.

In the case of the travel domain dataset, our features perform better than the features from Wicaksono et al. given that some of their features were specific to the domain i.e.

discussion threads. In the case of tweets, our features perform poorly in comparison to the features from Dong et al. This may be attributed to the features which are dependent on parsing and part of speech tagging, since automatically tagged dependency and part of speech tags maybe more erroneous for tweets in comparison to other domains. Secondly, Dong et al. used class weighting as a measure to overcome class imbalance but do not specify the class weights used by them, however, we use a class weight of 1:5 for non-suggestion and suggestion classes respectively.

Overall, following observations are made from the results:

1. **Domain dependency in training:** In three out of four datasets, open domain and cross domain training improved the f-score over the in-domain training of classifiers when the proposed features were used. The highest F scores were achieved in the open domain training of classifiers. An exception was the software domain, where there was a significant drop in performance on mixing the training datasets, specially in the cross domain setting.

2. **Domain complexity:** The travel domain appears to be the most difficult one for suggestion mining. This also aligns well with the relatively lower inter-annotator agreement observed for this domain (Chapter 4). We observe a number of examples in the travel domain where suggestion and non-suggestion sentences carry similar syntactic properties, and context plays an important role in distinguishing between the two classes of sentences. For example: 'You can get a ticket that covers 6 of the National Gallery sites for only about US$10, for example (including Sternberg Palace and St Agnes Convent).', is not a suggestion but a factual sentence using modal verb and 'you' which are a typically present in suggestion sentences. This becomes evident from the preceding sentence, i.e. 'Old Masters are not my particular favorite art period, anyway, there are plenty of those in the National Gallery.' A contrary scenario is the use of sentences like, 'It's easy to see the CT via the train.' to give a suggestion, where typical suggestion cue are

| **Features** | **Classifier** | **Travel Discussions (original)** | | | **Microsoft Tweets** | | |
|---|---|---|---|---|---|---|---|
| | | P-Avg | R-Avg | F-Avg | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ |
| Wicaksono and Myaeng [73, 74] | CRF | 0.768 | 0.746 | 0.757 | NA | NA | NA |
| | SVM | 0.714 | 0.711 | 0.712 | NA | NA | NA |
| Dong et al. [18] | FM | NA | NA | NA | 0.710 | 0.678 | 0.694 |
| | SVM | NA | NA | NA | 0.622 | 0.644 | **0.633** |
| Uni, P1-P8 | SVM | 0.74 | 0.742 | **0.739** | 0.63 | 0.5 | 0.557 |
| Uni, P1-P8, R1-R4 | SVM | 0.74 | 0.742 | **0.739** | 0.618 | 0.55 | 0.582 |
| NA | Rules | 0.438 | 0.313 | 0.365 | 0.22 | 0.693 | 0.335 |

TABLE 5.11: Comparison of results from related work whose datasets are available. Evaluation is performed using a 5-fold cross validation on the provided datasets.

absent, or 'I would go in fall' which actually means 'If I were you, I would go in fall'. We observe that the writers in travel discussion forums and hotel reviews tend to use figurative language more than the other domains, eg, 'You do know that November isn't the best time to visit the AC, right?'

3. **Top features:** Table 5.12 lists the top ranked features when different training datasets were used. Similarly, Figure 5.7 shows a comparison in performance of the lexical rules and syntactic rules, where syntactic rules tend to be more effective. No content words appear in the top 10 features, except the suggestion keywords. Imperative and subjunctive mood patterns, as well as some POS tag ngrams consistently appear as the top 10 features. Named Entity features proposed by us, and the related work features $R_2$ (proper noun and modal verb as the arguments of an nsubj dependency), $R_4$ (conj, csubj, nsubj dependencies) never appeared in the top 100 features. Table 5.13 provides a summary reference of feature classes and their description.

| **Datasets** | **Features** |
|---|---|
| **Top 10 features** | |
| Hotel | you, suggestion keywords ($P_1$), if you, VB, if, ImperativePattern-VB-NN- ($P_3$), SubjunctivePatternVB-NN- ($P_4$), VBD, ImperativePatternVB-VB-NN- ($P_3$) |
| Electronics | VB, ImperativePatternVB-NN-($P_3$), SubjunctivePatternVB-NN- ($P_4$),you, recommend, suggestion keywords ($P_1$), subjunctivePatternNN-VB- ($P_4$), VB DT, if you |
| Travel | you, ImperativePatternVB-NN- ($P_3$), VB, if you, ImperativePatternVB-VB- ($P_3$), SubjunctivePatternVB-NN- ($P_4$), ImperativePatternVB-VB-NN- ($P_3$), SubjunctivePatternNN-VB- ($P_4$), if, PRP MD |
| Software | ImperativePatternVB-VB- ($P_3$), VB, ImperativePatternVB-VB-NN- ($P_3$), Template 6 ($R_1$), SubjunctivePatternVB-NN- ($P_4$), ImperativePatternVB-NN- ($P_3$), Imperative mood ($R_3$), be, SentiWordNet score normalised sum ($P_7$), MD VB |
| All (open domain) | VB, SubjunctivePatternVB-NN- ($P_4$), ImperativePatternVB-NN- ($P_3$), ImperativePatternVB-VB- ($P_3$), ImperativePatternVB-VB-NN ($P_3$), MD VB, MD, you, VBD, SentiWordNet score normalised sum ($P_7$) |
| **Features which did not appear in the top 100 features** | |
| Hotel | SentiWordNet score sum ($P_6$), named entity ($P_8$), nsubj-proper noun-modal verb ($R_2$), conj/csubj/nsubj ($R_4$) |
| Electronics | SentiWordNet score sum ($P_6$), named entity ($P_8$), nsubj-proper noun-modal verb ($R_2$), conj/csubj/nsubj ($R_4$) |
| Travel | named entity ($P_8$), nsubj-proper noun-modal verb ($R_2$), conj/csubj/nsubj ($R_4$) |
| Software | nsubj-proper noun-modal verb ($R_2$), conj/csubj/nsubj ($R_4$), named entity ($P_8$) |
| All | 9 out of 10 template features ($R_1$), nsubj-proper noun-modal verb ($R_2$), conj/csubj/nsubj ($R_4$) |

TABLE 5.12: Insights into the ranking of features. A reference summary table for feature classes is provided below

| **Feature class** | **Description** |
|---|---|
| $R_1$ | Automatically extracted suggestion templates (Table 5.1) e.g. *need, require, please, i would like, require to* etc. |
| $R_2$ | Proper noun and modal verb |
| $R_3$ | Presence of imperative mood |
| $R_4$ | linguistic dependencies 'conj', 'csubj', and 'nsubj' |
| $P_1$ | Suggestion keywords. expanded from seed words |
| $P_2$ | PoS unigrams, bigrams and trigrams |
| $P_3$, $P_4$ | Imperative and Subjunctive mood sequential patterns of pos tags |
| $P_5$ | nsubj dependency and its attributes |
| $P_6$, $P_7$ | Sentiwordnet score sum absolute and normalized respectively |
| $P_8$ | presence and type of named entity |

TABLE 5.13: Reference summary table for feature types

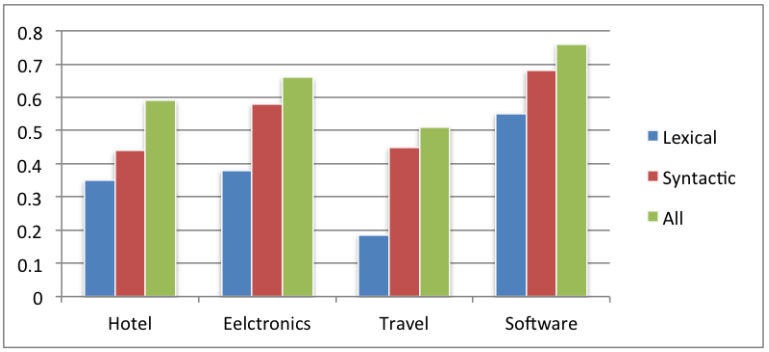



FIGURE 5.7: Comparison of the performance of lexical and syntactic rules across datasets.

## 5.7 Summary

This chapter presented a detailed evaluation and analysis of manually proposed features for suggestion mining. The new features proposed by us improved the performance over the baseline and previously proposed features. To the best of our knowledge, this is the first ever study of open domain and cross domain performance of features and classifiers for suggestion mining. A general trend observed in the features for both statistical and rule based classifiers is that the syntactic features were driving the performance of classifiers. Also, increasing the size of training dataset for a domain by mixing datasets from other domains (open domain training) resulted in much better performance than to use the small sized training dataset from the same domain as the evaluation dataset, with an exception of software domain. Travel discussion domain remains to be most challenging for the classifier.

For most of the domains, classification performance was fairly high with the use of manually selected features. However, there is a significant amount of manual observation and feature tailoring involved in the case of explicitly defined features. In the

next chapters, we investigate the effectiveness of automatically learned features for this problem, and compare the performance with the manual feature engineering.

# Chapter 6

# Representation Learning for Suggestion Mining

## 6.1 Introduction

In the previous chapter we investigated into using different hand picked features for statistical classifiers. We observed that out of all features, syntactic features carried more information than the lexical features in domain specific, open domain, and cross domain training scenario. While some of the lexical features were effective because they carried suggestion keywords (suggestion, recommend, advice etc.), or commonly used non-content words in suggestion sentences (you, if, would, could etc.), the n-gram features representing the content were limited by the size of the training datasets in order to capture the semantic associations with the two classes of sentences. Consider the suggestion, 'For a lovely breakfast, turn left out of the front entrance - on the next corner is a cafe with fresh baked breads and cooked meals.' This sentence was not predicted as a suggestion in the best performing model for the hotel test data (open domain training with all the features). This suggestion does not have the common lexical features like modal verbs, suggestion keywords, etc., but it recommends food from a place near the hotel, which carries similar semantics as to many other suggestions in hotel reviews across different hotels. Our suggested semantic features, i.e., automatically identified named entities, also did not show up in the top 100 effective features.

In this chapter, we study the performance of automatically learned features (representation learning) for suggestion mining, which tend to capture both syntax and semantics of the text in an implicit manner. To the best of our knowledge, ours is the first attempt to employ and evaluate representation learning for suggestion mining. This chapter

also contributes towards the investigation into the second research question, i.e., *What are good features for suggestion mining*, where we investigate if automatically learned features perform better than the manually identified features in the previous chapter. Primarily, dense (50 - 300 dimensions) and continuously valued word vectors learned using large unlabeled corpora are the features being evaluated in this chapter. Two types of classifiers are used with these word vectors, SVM and Long Short Term Memory (LSTM) neural networks. Therefore, the second objective of this chapter is to investigate which of the two classifiers work best with the automatically learned features. Like the previous chapter, all the experiments are performed in domain specific training, open domain training, and cross domain training. Word vectors learned using two different methods are evaluated. Further depth is added to the experiments by evaluating how the performance is co-related to the number of dimensions in the word vectors. An earlier version of this study has been published in [51].

## 6.2 Word Vectors

Many hand picked features for sentence classification problems are based around words, containing lexical, syntactic, semantic, or sentiment related information about words in the text. As seen in the previous chapter also, unigram features still dominate as the baseline features in a number of statistical text classification problems. In the previous chapter, each unigram feature represented one dimension in the feature vector of the sentence, whose value was based on the frequency of this unigram in the given training data. Such feature vectors are typically very large, with the number of dimensions as that of the entire vocabulary of the training dataset. By using feature selection methods, some of the seemingly less important unigrams can be altogether removed from the feature set. A different approach is to represent each word itself as a low dimensional vector using methods like dimensionality reduction, where each dimension stands for an explicitly defined concept or an implicitly learned concept. Therefore, each word vector holds the information of its degree of association with different concepts. It has long been an established area of research (distributional semantics), and have recently gained momentum due to the demonstrated success of neural network based approaches to learn these word vectors [22], [7]. In the current research scenario, a popular practice is to produce vector representations for each word of a given language using language corpora comprising of billions of unlabeled sentences (like Wikipedia), and employ these vectors in a number of tasks. In recent years, a number of methods have been proposed to learn these vectors, which includes neural network based algorithms. The term *word embeddings* is often used for dense and continuous word vectors whose dimensions do not represent any explicit concept. In the rest of the thesis, we may use a more generic

term *word vector* for *word embeddings*.

Different methods to learn these word vectors have been proven to effectively capture syntactic and semantic information in words [56], and also sentiment information in some of the works [67]. For neural network based text classification methods, word vectors can either be simultaneously learned for the full training vocabulary during the training phase of the classifier, or some pre-trained word vectors can be directly used to initiate the word vectors for the training vocabulary at the start of the training. In the second case, the vectors can be updated during the training or can be kept fixed to their initial value.

### 6.2.1 Pre-trained word vectors

In this chapter we experiment with the already available pre-trained word vectors, which were trained on very large corpora of English language. These vectors capture generic semantic and syntactic aspects of words and have been proven to be performing well on a number of benchmark datasets and tasks, like, word similarity and relatedness, synonym detection, named entity recognition, sentiment analysis. Following are the two pre-trained word vectors employed in this chapter:

**GloVe:** Baroni et al. [7] compared the word vectors obtained through the conventional count (word co-occurrence matrix) and the recently developed predict (neural network) methods on word semantics related tasks, and concluded that predict methods perform better overall. The predict vectors were obtained from the CBOW variant of the word2vec algorithm [41]. Shortly after, Pennington et al. [56] showed that their GLoVE model, which was developed using word co-occurrence matrix, outperformed the prediction based word2vec method. They also argued that the count and prediction based classes of methods are not far apart, since they are both dependent on the underlying co-occurrence statistics of the corpus, but the count based methods are much more efficient. Based on these findings, we study the employment of word vectors obtained using the GLoVE algorithms for suggestion mining. Details of the GLoVE algorithm are discussed in Chapter 2. The pre-trained GLoVE vectors are available in 4 different dimension sizes, which we refer to as $\text{GLoVE}_{300}$, $\text{GLoVE}_{200}$, $\text{GLoVE}_{100}$, $\text{GLoVE}_{50}$, which corresponds to Glove vectors of 300, 200, 100, and 50 dimensions respectively. These vectors are learned on English Wikipedia and English Gigaword corpus.

**Deps:** Based on the effectiveness of syntactic features demonstrated in the previous

chapter, we also experiment with dependency based word embeddings, which are referred to as Deps. [34]. The learning algorithm for these embeddings remains same as that of word2vec, except that the contexts of a word are determined on the basis of its dependency relations with other words, instead of the window based context used to learn most of the word vectors. In order to learn these embeddings, the training data is required to be pre-labeled with linguistic dependency relations. Therefore, Deps have been shown to perform better in determining the functional similarity between words as compared to word2vec. More details on dependency embeddings have been provided in Chapter 2.

Table 7.10 provides some statistics of both kinds of pre-trained word vectors. Table

| **Embedding** | **Training data** | **Vocabulary** | **Dimensions** |
|---|---|---|---|
| GLoVE | Wikipedia 2014, English gigaword corpus, 6 billion tokens | 400,000 | 300, 200, 100, 50 |
| Deps | English Wikipedia 2013, 2 billion tokens | 175,000 | 300 |

TABLE 6.1: Statistics for pre-trained word vectors employed in this chapter

| **Target word** | **GLoVE$_{50}$** | **GLoVE$_{100}$** | **GLoVE$_{200}$** | **GLoVE$_{300}$** | **Deps** |
|---|---|---|---|---|---|
| suggest | suggesting, suggests, explain, indicate, fact | indicate, suggesting, suggests, suggested, indeed | suggesting, suggests, indicate, suggested, might | suggesting, suggests, indicate, suggested, might | recommend, proposes, reccomend, recomend, indicate |
| breakfast | dinner, buffet, lunch, meals, dinners | dinner, lunch, buffet, meals, meal | dinner, lunch, meals, buffet, breakfasts | dinner, lunch, breakfasts, meals, meal | lunch, dinner, brunch, meal, drivetime |
| amsterdam | rotterdam, frankfurt, paris, vienna, zurich | rotterdam, frankfurt, paris, stockholm, hamburg | rotterdam, frankfurt, netherlands, schiphol, utrecht | rotterdam, schipol, utrecht, netherlands, frankfurt | rotterdam, utrecht, ghent, antwerp, dordrecht |
| if | not, could, does, might, would | because, not, does, so, but | not, does, because, n't, should | not, does, should, n't, could | eventhough, because, although, unless, whether |
| you | 'll, ?, 'd, me | 'll, n't, know, i, do | 'll, ?, n't, know, me | 'll, n't, ?,know, i | we, commenti, 'you', jeeny, they |

TABLE 6.2: Comparison of the top 5 similar words for a given word using word vectors from different embeddings

6.2 lists top five similar words to a target word obtained by calculating word similarity based on the weights from GLoVE and Deps pre-trained vectors. It shows that Deps do not tend to consider the morphological variations as most similar words to a target word, while GLoVE does. Deps embeddings mostly brings up words with similar part of speech tags, specially in the case of non-content words like *you* and *if*. Deps also tends

to exclude punctuation marks since the linguistic dependency relations exist between words and not tokens, while GLoVE includes all types of tokens in it vocabulary.

## 6.3 Classifiers

We employ the Support Vector Machine (SVM) classification algorithm as in the previous chapter, in order to compare its performance with both manually picked features and automatically learned features. In addition, we also employ Long Short Term Memory (LSTM) network classifiers which belong to the class of neural networks, which are considered to be best suited to use with dense and continuous word representations. Chapter 2 provides details on the functionality of SVM and LSTM classifiers.

SVM relies on the input being in the form of a feature vector for the full sentence, and learns a classification function using these dimensions as parameters of the function. While LSTM initiates or accepts pre-learned word vectors one at a time through its input layer, and further processes these initial vectors through one or more hidden layers of the network, ultimately converting them into a more task specific feature vector at the output layer. The output vector processed through the hidden layers of a neural network can contain higher, equal, or lower number of dimensions than the input vectors. In short, SVM classifiers are not directly involved in learning the feature representations, while LSTM are.

Since it is a sentence classification task, word vectors are required to be composed into sentence vectors before being fed to the SVM classifier. The word vectors should be explicitly combined into a sentence vector before they are fed to the SVM classifiers, while LSTM directly takes in the word vectors in a sequential manner, as they appear in the sentence. Therefore, the composition of word vectors into sentence vectors is a part of the classification framework in the case of LSTM.

### 6.3.1 Neural Network Architecture

As we already saw in Chapter 2, LSTMs are considered to be well suited for prediction tasks related to sequences, which in our case is a sentence, i.e., a sequence of words. This is because they process one item of the sequence at a time, and retain information from the previously processed items, while at the same time deciding which and how much of this information should be remembered till the end of processing the sequence.

**Network architecture:** We evaluated a smaller set of hyper-parameter values as compared to all possible combinations, by picking up some popular values advised by the existing studies [9]. These configurations were evaluated on the domain specific train-test settings, with a 200 dimensional GLoVE vectors. The evaluated hyper-parameter values include the following, with the highlighted ones being the values which were finally chosen:

- Activation functions: **tanh**, relu, softsign
- Regularizer: Dropout (0.2), **l2 regularisation (0.02)**
- Optimisation: SGD, **Adam**.
- Number of hidden layers: 1, **2**, 3
- Type of hidden layers: **LSTM hidden layers**, fully connected dense hidden layers
- Number of epochs: **5**, 8, 10
- Batch size: **35**, 50, 100

We chose the configuration which performed best for a majority of domains, i.e. at-least 3 out of the four domains. The final architecture comprises of 2 LSTM hidden layers with the number of units in each layer depending on the number of units in the input layer (number of dimensions of the pre-trained word vectors). We use 100 and 50 hidden units with 300 and 200 dimensions of input layer, and 50 and 20 units with 100 and 50 dimensional input layer, with an aim to obtain lower dimensional dense representations as we move up in the network. The input layer comprises of LSTM units, and a softmax activation is applied to the outermost dense layer of size 2. Tanh activation is used at each layer, and L2 regulariser with a value of 0.02 is used with each layer. It was observed that the F1 score either remained the same or decreased for all the domains when more than 5 training epochs were used, so we restrict the number of epochs to 5.

### 6.3.2 Support Vector Machines

The SVM classifier implemented in this chapter remains same as the previous chapter with regard to most of the parameters. The number of features used changes from 1000 to the number of dimensions of the pre-trained word vectors, which ranges from 300 to 50.

**Sentence vectors for SVM:** The previously employed SVMs were fed with the feature representation of sentences where each dimension of the feature vector represented

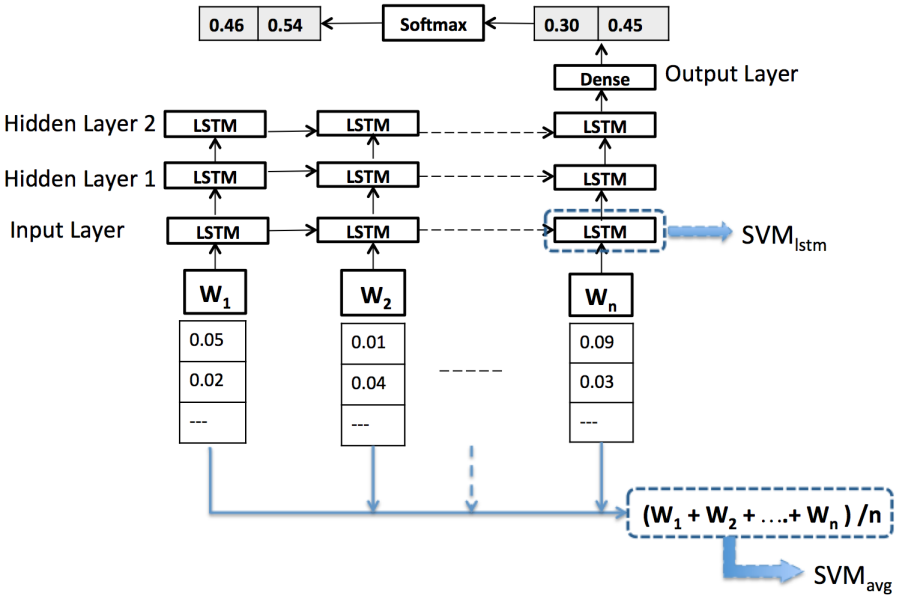



FIGURE 6.1: The two methods used to obtain sentence vectors for SVM using the pre-trained word embeddings.

a single feature. Whereas, in this chapter we first build a feature vector for the sentences by performing two kinds of operations on the already available word vectors. The dimensions of this kind of feature vector may not represent explicit or definable attributes of the sentence, as opposed to manual feature engineering.

1. **Averaging:** Sentence vector is obtained by taking an average of vectors for the constituent words. This method has been used as a baseline for obtaining sentence representations in a number of works, and have been shown as a simple yet strong baseline for semantic textual similarity [30]. Considering a sentence with n words, $w_1$,$w_2$,...,$w_n$, and the corresponding word vectors $W_1$,$W_2$,...,$W_n$, the sentence vector S in this case will be:

$$S = \frac{1}{n}\sum_{i=1}^{n} W_i$$

We denote the SVM classifiers using averaged word vectors as SVM$_{avg}$. In this method, a number of dimensions in the sentence vector remain the same as the number of dimensions of the word vector. This method considers the sentence as a bag of words and does not preserve the sequential nature of sentences, and therefore tends to lose the compositional meaning of words. In the case of average word vectors as above, these two sentences will have the same vectorial representation: 'I recommend the italian cafe, next door.' and 'I recommend the italian door, next cafe'. However, for our problem

there are very low chances of existence of such pairs of suggestion and non-suggestion sentences which differ only in their word order.

2. **Representations from a trained LSTM network:** This method uses the sentence representations obtained as an output of the first (input) layer of a LSTM network trained for suggestion mining (Figure 6.1). The sentence vectors can be of equal or smaller number of dimensions than the pre-trained word vectors, depending on the neural network architecture. In our case, this is the size of first hidden layer which comprises of lower number of units than the input layer, which is 100 (for 300 and 200 dimensions word vectors) or 50 (with 100 and 50 dimensional word vectors). In order to produce train and test sentence representations for this version of SVM, we use the LSTM classifier from the counterpart experiments which use LSTM as the main classifier instead of SVM. We denote the SVM classifiers which use these representations as $\text{SVM}_{lstm}$.

Figures 6.2 and 6.3 show a two dimensional plot of sentence vectors of the hotel domain test dataset which were obtained from the two methods described above. Red colour represents the positive, while black represents the negative class. The vectors obtained using the LSTM method appear to be better separable in this case.

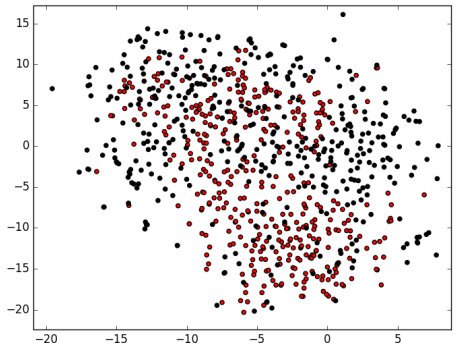



FIGURE 6.2: $\text{SVM}_{avg}$

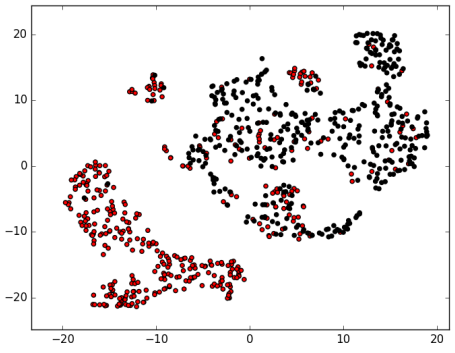



FIGURE 6.3: $\text{SVM}_{lstm}$

FIGURE 6.4: A two dimensional mapping of sentence vectors corresponding to the hotel test dataset, where sentence vectors are obtained by averaging the constituent word vectors.

## 6.4 Training and Evaluation

**Training datasets:** In this chapter, we again perform the classifier and feature evaluation using three different settings for training, i.e., domain specific, open domain, and cross domain. Table 6.3 revisits the details of train and test datasets, which remain the

| Dataset | Class ratio | Dataset | Class ratio |
|---|---|---|---|
| **Training datasets** | | **Test datasets** | |
| Hotel train | 448/7086 | Hotel test | 404/404 |
| Electronics train | 324/3458 | Electronics test | 101/101 |
| Travel train | 1314/3869 | Travel test | 229/229 |
| Software train | 1428/4296 | Software test | 296/296 |
| **Related work datasets** | | | |
| Microsoft Tweets | 238/2762 | NA | NA |
| Travel-original | 2192/3007 | NA | NA |

TABLE 6.3: Datasets used in this chapter

same as in chapter 5.

**Comparison with the related work:** We also compare the best performing experiments in this chapter on datasets obtained from the related work, and compare the results with that of the previous works. A 5-fold cross validation is used for evaluation, as used in the previous works.

**No. of dimensions:** The pre-trained GLoVE vectors come with the dimension sizes 300, 200, 100, and 50, with the 300 dimensional vectors performing best in their introductory evaluation on word analogy tasks [56]. The co-relation between size of vectors and performance can vary for different tasks [40]. Since dense word vectors have never been employed for suggestion mining, the co-relation of classification performance with the number of dimensions is unknown. Therefore, we perform all the experiments which use GLoVE embeddings in 4 runs, with 4 different size of dimensions. The pre-trained Dependency embeddings on the other hand come with the 300 dimension version which was the dimensionality chosen for evaluation in their introductory study [34].

## 6.5 Results and Analysis

**Domain specific training:** Figures 6.5, 6.6, and 6.7 show the variation in F1 score with the use of different number of word vector dimensions for $\text{SVM}_{avg}$, $\text{SVM}_{lstm}$, and LSTM classifiers respectively, across all the datasets, in a domain specific training scenario. It was observed that 200 dimensions work best for GLoVE in most of the runs, with few exceptions. However, different datasets exhibit different amount of change in performance with the change of dimensions. Overall, the change in performance is monotonous with the $\text{SVM}_{avg}$ classifiers. In the case of $\text{SVM}_{lstm}$, and LSTM classifiers, some datasets exhibit non-monotonicity in performance on changing the dimensions. This is different from the trends shown in the earlier evaluation of word embeddings on sentence classification tasks like sentiment analysis [40]. Overall, software domain

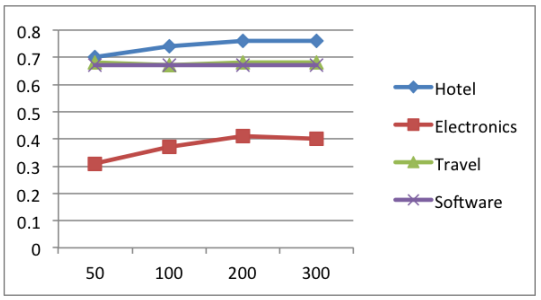



FIGURE 6.5: Variations in F1 score across the test datasets when different dimensions of GloVE are used with $SVM_{avg}$ in a domain specific training setting.

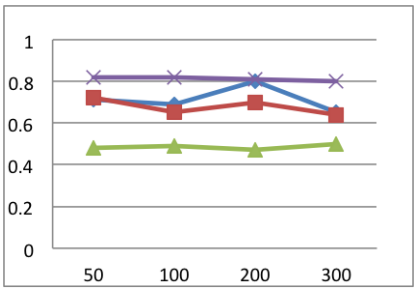



FIGURE 6.6: Variations in F1 score across the test datasets when different dimensions of GloVE are used with $SVM_{lstm}$ in a domain specific training setting.

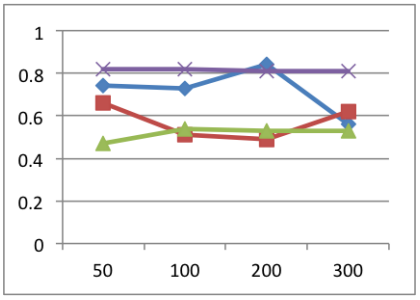



FIGURE 6.7: Variations in F1 score across the test datasets when different dimensions of GloVE are used with LSTM in a domain specific training setting.

| **Vectors** | **P**$_{sugg}$ | | | **R**$_{sugg}$ | | | **F**$_{sugg}$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | SVM$_{avg}$ | SVM$_{lstm}$ | LSTM | SVM$_{avg}$ | SVM$_{lstm}$ | LSTM | SVM$_{avg}$ | SVM$_{lstm}$ | LSTM |
| **Hotel** | | | | | | | | | |
| GlOVE$_{200}$ | 0.81 | 0.99 | 0.76 | 0.71 | 0.67 | 0.95 | 0.76 | 0.80 | **0.84** |
| DEPS | 0.70 | 0.93 | 0.99 | 0.79 | 0.35 | 0.33 | 0.74 | 0.51 | 0.50 |
| **Electronics** | | | | | | | | | |
| GlOVE$_{200}$ | 0.84 | 0.90 | 0.33 | 0.27 | 0.56 | 0.94 | 0.41 | 0.70 | 0.49 |
| DEPS | 0.70 | 0.90 | 0.99 | 0.79 | 0.45 | 0.33 | **0.74** | 0.60 | 0.50 |
| **Travel** | | | | | | | | | |
| GlOVE$_{200}$ | 0.52 | 0.62 | 0.68 | 0.99 | 0.38 | 0.43 | **0.68** | 0.47 | 0.53 |
| DEPS | 0.57 | 0.70 | 0.67 | 0.54 | 0.39 | 0.48 | 0.56 | 0.50 | 0.56 |
| **Software** | | | | | | | | | |
| GlOVE$_{200}$ | 0.50 | 0.87 | 0.83 | 0.99 | 0.76 | 0.81 | 0.67 | 0.81 | **0.82** |
| DEPS | 0.57 | 0.88 | 0.86 | 0.54 | 0.71 | 0.78 | 0.56 | 0.79 | **0.82** |

TABLE 6.4: Performance of different classifiers when trained and evaluated on datasets belonging to the same domain.

remains quite unaffected by the change in dimensions with all three classifiers.

Table 6.4 reports the performance of different classifiers for domain specific training. SVM$_{lstm}$ yields lowest performance against the other two classifiers across all domains. GLoVE vectors perform better than Deps, except in the case of electronics domain. It may indicate that GLoVE vectors perform well with larger number of training instances, since we have the smallest training dataset for the electronics domain. Mixed trends are observed with the precision and recall values, except that precision is always higher

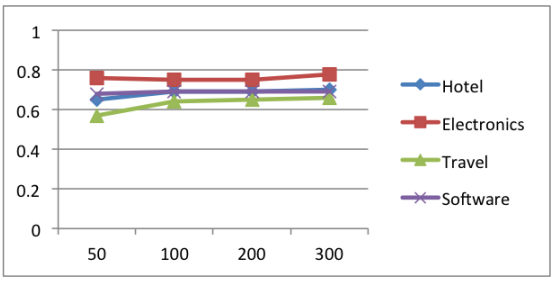



FIGURE 6.8: Variations in F1 score across the test datasets when different dimensions of GloVE are used with $SVM_{avg}$ in an open domain training setting.

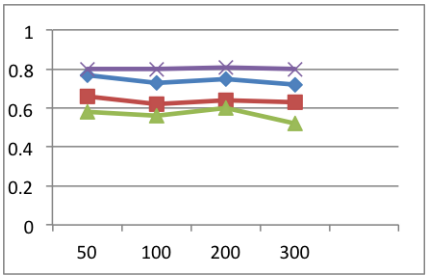



FIGURE 6.9: Variations in F1 score across the test datasets when different dimensions of GloVE are used with $SVM_{lstm}$ in an open domain training setting.

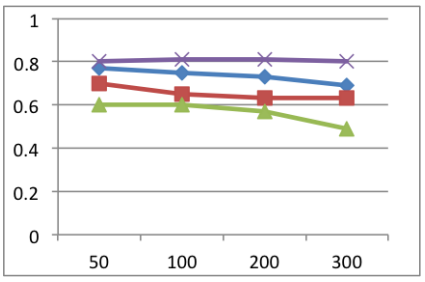



FIGURE 6.10: Variations in F1 score across the test datasets when different dimensions of GloVE are used with LSTM in a open domain training setting.

than recall for the $SVM_{lstm}$ classifier, with both kind of vectors.

**Open domain training:** Figures 6.8, 6.9, and 6.10 show the variation in F1 score with the use of different number of word embedding dimensions for $SVM_{avg}$, $SVM_{lstm}$, and LSTM classifiers respectively, in an open domain training scenario. 300 appears to be best performing dimension for $SVM_{avg}$, and 50 for $SVM_{lstm}$ and LSTM. The change in performance with the number of dimensions do not appear to be as dramatic as observed in the domain specific training, which could be attributed to the same training data (combination of all training datasets) used in all the runs. This may also indicate that the influence of dimension size remains low with the larger training datasets. However, the highest F 1 score values achieved for different domains are mostly lower than domain specific training. The software domain again remains largely unaffected by the change in dimensions. The best performance is shared by all three classifiers in equal number of cases. Precision remains higher than recall for $SVM_{lstm}$ and LSTM classifiers.

**Invalid results with Dependency embeddings:** In tables 6.5 and 6.6 which summarise the results for open domain and cross domain runs, we can see that when dependency embeddings are used with the $SVM_{avg}$ classifier, all the test instances are predicted as suggestions (a recall of 1.00), which results in a F score of 0.67. Given that

| Vectors | $\mathbf{P}_{sugg}$ | | | $\mathbf{R}_{sugg}$ | | | $\mathbf{F}_{sugg}$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $SVM_{avg}$ | $SVM_{lstm}$ | LSTM | $SVM_{avg}$ | $SVM_{lstm}$ | LSTM | $SVM_{avg}$ | $SVM_{lstm}$ | LSTM |
| **Hotel** | | | | | | | | | |
| $GlOVE_{300}$ | 0.85 | | | 0.59 | | | 0.70 | | |
| $GlOVE_{50}$ | | 0.96 | 0.98 | | 0.65 | 0.64 | | **0.77** | **0.77** |
| DEPS | 0.50 | 0.76 | 0.63 | 1.00 | 0.47 | 0.52 | 0.67 | 0.58 | 0.57 |
| **Electronics** | | | | | | | | | |
| $GlOVE_{300}$ | 0.70 | | | 0.87 | | | **0.78** | | |
| $GlOVE_{50}$ | | 0.87 | 0.88 | | 0.53 | 0.58 | | 0.66 | 0.70 |
| DEPS | 0.50 | 0.80 | 0.74 | 1.00 | 0.56 | 0.50 | 0.67 | 0.66 | 0.59 |
| **Travel** | | | | | | | | | |
| $GlOVE_{300}$ | 0.65 | | | 0.66 | | | **0.66** | | |
| $GlOVE_{50}$ | | 0.73 | 0.74 | | 0.49 | 0.51 | | 0.58 | 0.60 |
| DEPS | 0.50 | 0.62 | 0.51 | 1.00 | 0.42 | 0.38 | 0.67 | 0.50 | 0.43 |
| **Software** | | | | | | | | | |
| $GlOVE_{300}$ | 0.55 | | | 0.94 | | | 0.69 | | |
| $GlOVE_{50}$ | | 0.81 | 0.81 | | 0.80 | 0.80 | | **0.80** | **0.80** |
| DEPS | 0.50 | 0.57 | 0.50 | 1.00 | 0.30 | 0.32 | 0.67 | 0.39 | 0.39 |

TABLE 6.5: Performance of different classifiers when trained on a combination of training datasets from different domains which also includes the evaluation domain

this is not the case in domain specific training, it can be assumed that performing an averaging operation over word embeddings using dependency embeddings capture semantic features for the sentence well, while fail to be good representations for syntactic features because open domain and cross domain model performance tends to heavily rely on the syntactic features. We therefore do not include these entries when making a comparison for the best F score, even though the F-score could be higher than the counterparts. The results are marked with an underline in the results tables. In another study by [33], where dependency based word embeddings are used for sentence classification tasks, it was seen that the results were lower with averaged out embeddings with SVM classifier, in comparison to LSTM and CNN classifiers.

**Cross domain training:** Figures 6.11, 6.12, 6.13 show the variation in F1 score with the use of different number of word vector dimensions for $SVM_{avg}$, $SVM_{lstm}$, and LSTM classifiers respectively, in a cross domain training scenario. Similar to the open domain training, the best performance was distributed among the 300 and 50 dimensional embeddings. $SVM_{avg}$ classifier works best for 3 out of 4 domains in this case. Like the previous two cases, $SVM_{avg}$ exhibits a monotonic F score vs dimensions curve, while a few small peaks were observed in the rest of the two classifiers. The highest F 1 score values achieved for different domains is lowest than the previous two settings. Precision again remains higher than recall for $SVM_{lstm}$ and LSTM classifiers.

**GLoVE vs. Deps vectors:** Overall, GLoVE vectors outperform the dependency embeddings. One reason for this could be that the vocabulary size of dependency embeddings is 175,000 while that of GLoVE is 400,000. Graph 6.14 shows the comparison of vocabulary overlap of different datasets with both kind of embeddings, which shows that dependency embeddings have a smaller vocabulary overlap for all the datasets in

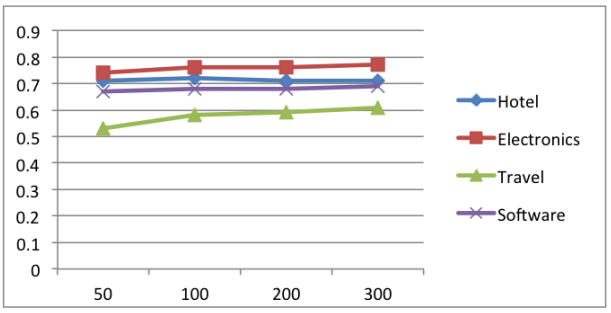



FIGURE 6.11: Variations in F1 score across the test datasets when different dimensions of GloVE are used with $SVM_{avg}$ in cross domain training.

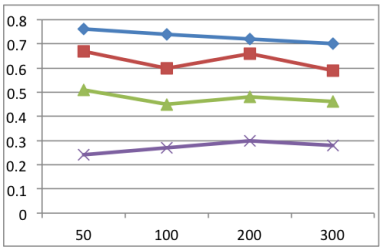



FIGURE 6.12: Variations in F1 score across the test datasets when different dimensions of GloVE are used with $SVM_{lstm}$ in cross domain training.

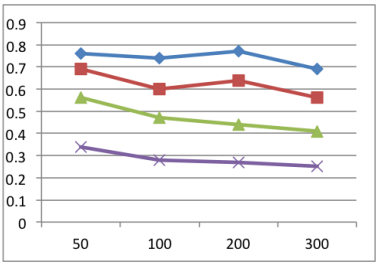



FIGURE 6.13: Variations in F1 score across the test datasets when different dimensions of GloVE are used with LSTM in a domain specific training setting.

| **Vectors** | **P**$_{sugg}$ | | | **R**$_{sugg}$ | | | **F**$_{sugg}$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | SVM$_{avg}$ | SVM$_{lstm}$ | LSTM | SVM$_{avg}$ | SVM$_{lstm}$ | LSTM | SVM$_{avg}$ | SVM$_{lstm}$ | LSTM |
| **Hotel** | | | | | | | | | |
| GLoVE$_{300}$ | 0.69 | | | 0.74 | | | 0.71 | | |
| GLoVE$_{50}$ | | 0.80 | 0.78 | | 0.73 | 0.75 | | **0.76** | **0.76** |
| DEPS | 0.50 | 0.75 | 0.79 | 1.00 | 0.61 | 0.63 | 0.67 | 0.68 | 0.70 |
| **Electronics** | | | | | | | | | |
| GLoVE$_{300}$ | 0.66 | | | 0.91 | | | **0.77** | | |
| GLoVE$_{50}$ | | 0.80 | 0.73 | | 0.58 | 0.66 | | 0.67 | 0.69 |
| DEPS | 0.50 | 0.77 | 0.77 | 1.00 | 0.52 | 0.57 | 0.67 | 0.62 | 0.66 |
| **Travel** | | | | | | | | | |
| GLoVE$_{300}$ | 0.67 | | | 0.56 | | | **0.61** | | |
| GLoVE$_{50}$ | | 0.77 | 0.77 | | 0.39 | 0.44 | | 0.51 | 0.56 |
| DEPS | 0.50 | 0.83 | 0.79 | 1.00 | 0.35 | 0.43 | 0.67 | 0.49 | 0.56 |
| **Software** | | | | | | | | | |
| GLoVE$_{300}$ | 0.56 | | | 0.90 | | | **0.69** | | |
| GLoVE$_{50}$ | | 0.59 | 0.61 | | 0.19 | 0.24 | | 0.28 | 0.34 |
| DEPS | 0.50 | 0.57 | 0.59 | 1.00 | 0.17 | 0.21 | 0.67 | 0.26 | 0.31 |

TABLE 6.6: Cross-domain evaluation: Performance of different classifiers when trained on a combination of datasets belonging to different domains, which excludes the evaluation domain.

comparison to GLoVE. However, some of the additional vocabulary in GLoVE could be tokens (symbols etc.) and not necessarily words.

**Comparison with the hand crafted features:** Table 6.7 summarizes the results and compares them with the results obtained in the previous chapter (manually identified features). Manual features remain slightly above word vectors if we only compare

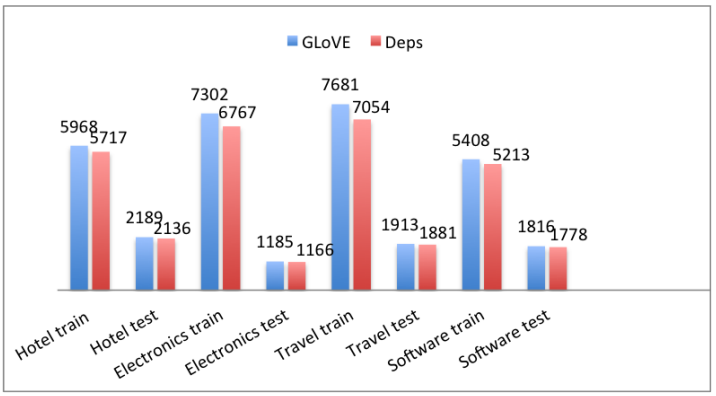



FIGURE 6.14: Comparison of vocabulary overlap of GLoVE and Dependency embeddings with different datasets

the number of highest F scores achieved. However, word vectors always perform much better for the travel domain, given that it was observed to be the most challenging domain in the previous chapters.

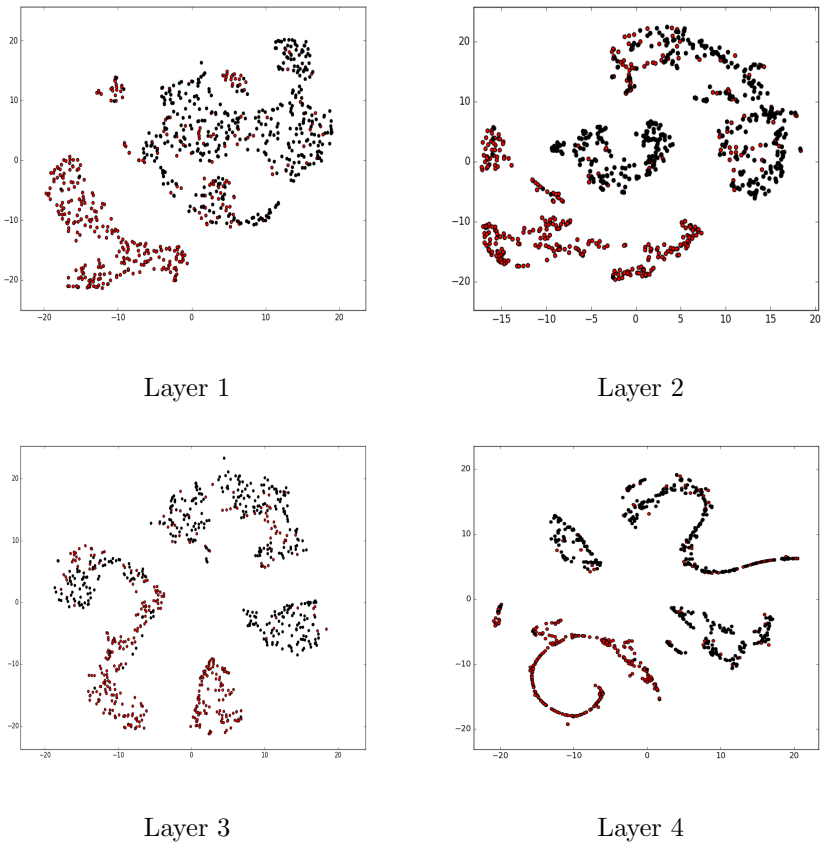
**Sentence vectors:** Even though the sentence representations obtained from LSTM

| **Domain** | **SVM**$_{manual}$ | **Using word embeddings** | | |
|---|---|---|---|---|
| | | **SVM**$_{avg}$ | **SVM**$_{lstm}$ | **LSTM** |
| **Domain specific** | | | | |
| Hotel | 0.815 | 0.76 | 0.80 | **0.84** |
| Electronics | **0.766** | 0.74 | 0.70 | 0.50 |
| Travel | 0.519 | **0.68** | 0.50 | 0.56 |
| Software | **0.822** | 0.67 | 0.81 | **0.82** |
| **Open domain** | | | | |
| Hotel | **0.891** | 0.70 | 0.77 | 0.77 |
| Electronics | **0.894** | 0.78 | 0.66 | 0.70 |
| Travel | 0.57 | **0.66** | 0.58 | 0.60 |
| Software | 0.796 | 0.69 | **0.80** | **0.80** |
| **Cross domain** | | | | |
| Hotel | **0.84** | 0.71 | 0.76 | 0.76 |
| Electronics | **0.848** | 0.77 | 0.67 | 0.69 |
| Travel | 0.515 | **0.61** | 0.51 | 0.56 |
| Software | **0.767** | 0.69 | 0.28 | 0.34 |

TABLE 6.7: Comparison of best results obtained by manual features vs best results obtained by word embeddings in all three settings

appear to be fairly discriminative with regard to the two classes, the averaging method works better with SVM. Figure 6.15 plots the representation of sentences from the hotel test dataset obtained after each layer of the trained network. The class separation appears to improve with the number of layers, resulting in a very complex function at the output later.

**Dimensionality vs performance:** The evaluations from previous studies on word



Layer 1 Layer 2

Layer 3 Layer 4

FIGURE 6.15: Separation of positive and negative class in hotel test dataset when moving up the layers of the LSTM classifier.

vectors show that the performance of embedding dimensions is correlated with the nature of task. These studies use neural network classifiers with the word vectors. Pennington et al. [56] demonstrated that syntactic tasks minimally benefit from increasing the dimensions of GLoVE vectors. In our case, different patterns for performance vs. dimensions were observed within the same task, for different domains (Figure 6.7). One yet to test theory could be that the role of syntax and semantics in suggestion mining varies from domain to domain.

**Comparison with related work:** We achieve better results by using word representations in comparison to the manually defined features in the previous chapter. In the travel domain, our $SVM_{avg}$ classifier outperforms all the classifiers used by Wicaksono and Myaeng. For tweets, the Factorisation Machine classifier still performs better than our best classifier ($SVM_{avg}$).

**Further Experiments:** The experiments described above targeted the comparison of manually picked features and automatically learned features i.e. word embeddings. One possible extension to these set of experiments would be to use a combination of

| **Work** | **Classifier** | **Travel Discussions (original)** | | | **Microsoft Tweets** | | |
|---|---|---|---|---|---|---|---|
| | | P-Avg | R-Avg | F-Avg | $P_{sugg}$ | $R_{sugg}$ | $F_{sugg}$ |
| Wicaksono and Myaeng [73, 74] | CRF | 0.768 | 0.746 | 0.757 | NA | NA | NA |
| | SVM | 0.714 | 0.711 | 0.712 | NA | NA | NA |
| Dong et al. [18] | FM | NA | NA | NA | 0.710 | 0.678 | **0.694** |
| | SVM | NA | NA | NA | 0.622 | 0.644 | 0.633 |
| Our Experiments | $SVM_{avg}$ | 0.78 | 0.80 | **0.789** | 0.64 | 0.69 | **0.664** |
| | $SVM_{lstm}$ | 0.63 | 0.75 | 0.68 | 0.591 | 0.514 | 0.55 |
| | LSTM | 0.79 | 0.73 | 0.758 | 0.56 | 0.61 | 0.583 |

TABLE 6.8: Comparison of results from related work whose datasets are available. Evaluation is performed using a 5-fold cross validation on the provided datasets.

word embeddings and manually picked features as feature vectors. In the case of SVM classifiers, this can be achieved by concatenating the feature vectors of manually picked features and word sentence embeddings obtained by the two methods described above, i.e. *average* and *lstm*. However, in the case of LSTM classifiers, adding manually picked features isn't a usual practice because the principles behind using representation learning methods is to eliminate the need for manual feature engineering. However, experimenting with this set-up for the sake of comparison of results. Since LSTMs use feature vectors for each unit of the sequence which is word in this case, concatenating word level features to the embeddings vectors would be one straightforward method for this experiment. Eg, of word features would be word sentiment score, POS tag, etc. Not all the manually selected features are word level features, for example presence of suggestion phrases and templates in the sentence and therefore some modifications in the architecture may be required to incorporate sentence level features. One such modification could be to concatenate the vector representing sentence features to the output of layer preceding the dense layer.

## 6.6 Summary

In the experiments reported in this chapter, we used automatically learned word vectors in combination with the SVM and LSTM classification algorithms, while at the same time performed a new kind of extrinsic evaluation for the state of the art word vectors GLOVe and Deps. The experiments were extensive given that the experiment setup covered four different domains, different training settings, different types of classifiers, different types of vectors, and different sizes of dimensions used with the embeddings.

On the point of the primary question, i.e., manual vs. automatic features, manual features proved to be slightly ahead of the automatically learned features, with the given

set of datasets and the SVM learning algorithm. An exception being the travel discussion forum, where word embeddings always performed better than manual features. As opposed to the ongoing trend of statistical text classification, SVM classifiers proved to be better than LSTM classifier with both manual and learned features. It may be argued that exhaustive hyper parameter optimisation was not performed with LSTM to perform a deep comparison of LSTM and SVM in this case. However, a comparison of popular and default hyper parameter values was still performed, where it was observed that the best parameter values tend to vary with the dataset domain. Therefore, using domain specific parameter values may improve some of the current results for LSTM.

Moving to the use of pre-trained word embeddings, it was interesting to observe that SVM performed poorly with the sentence representations extracted from LSTM, in comparison to the representations obtained from the simpler method of averaging, which has also been shown as a strong baseline in other tasks. Pre-trained dependency embeddings perform poorly in comparison to GLoVE given the smaller vocabulary of the former, although dependency embeddings are proven to capture syntactic regularities, and syntactic manual features have been shown to be the performance drivers in the experiments described in the previous chapter. This indicates that GLoVE vectors are capable of capturing the amount of syntactic information required for the task, with an added benefit of larger vocabulary. The 200 dimensional GLoVE vectors remained the clear choice for domain specific training using all three types of classifiers. But this changed in the open domain and cross domain training setup, where 300 dimensions suited the $\text{SVM}_{avg}$ classifier, while 50 dimensions suited better with $\text{SVM}_{lstm}$ and LSTM classifiers.

Overall, most of the simpler methods proved to be better with the current training datasets. This could be attributed to the relatively smaller size of datasets, as compared to the bigger datasets which are generally used with neural network classifiers and automatic learning of feature representations. In the next chapter we investigate into building much larger silver standard datasets, and again employing representation learning with the same.

# Chapter 7

# Distant Supervision for Suggestion Mining

In the previous chapters we saw that the suggestion class instances are sparse in most of the training datasets, and so we adapted methods like class weights and oversampling to address the same. We also observed that the SVM classifiers were at par with the LSTM classifiers, which on the other hand are a popular choice of neural network classifiers for sequence classification. Also, for three out of four domains the manually identified features outperformed automatically learned features. This goes against the current trend of representation learning methods for supervised learning, which tend to outperform the methods which use manually determined features. Neural network based classifiers are known to automatically learn optimal data representations (representation learning) for minimizing the classification error. However, the manually labeled training datasets used previously were relatively small and unbalanced to leverage representation learning at a higher extent.

In this chapter we propose and evaluate some methods to address data sparsity in suggestion mining, while still avoiding manual labeling of additional data. We approach this by performing distant supervision by means of a silver standard training dataset comprising of about 1.3 million sentences, and employing representation learning methods with the same. This dataset is constructed from sentences extracted from the articles on wikiHow and Wikipedia, serving as positive and negative instances respectively, and is referred to as the *Wiki Suggestion* dataset. We propose two approaches to use this dataset, where both use a LSTM classifier, but vary in their method to employ the silver standard dataset in the training process. The first approach directly trains a classifier on the wiki dataset, while the second approach only learns word representations from this dataset. Different variants are proposed for each of the two approaches. One of

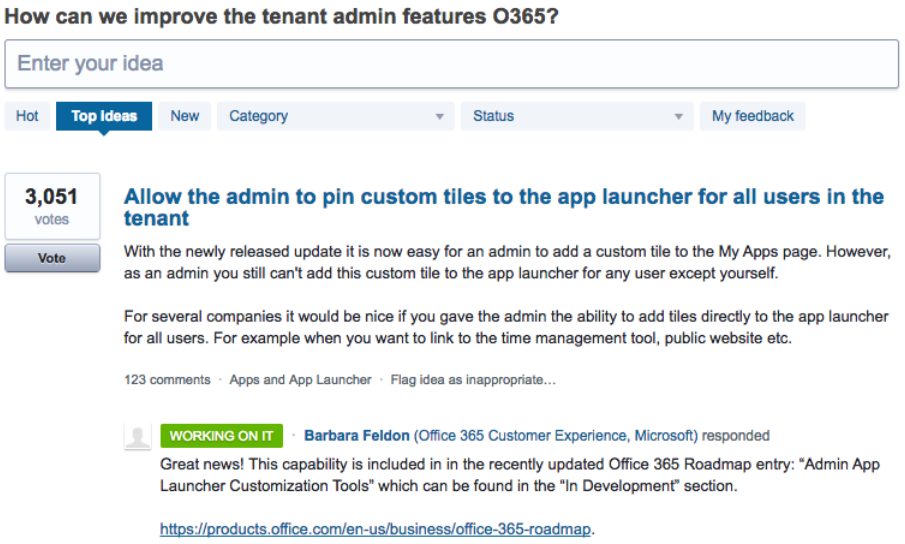



FIGURE 7.1: Web based suggestion forums

the variants of the second approach is to learn vectors for Part of Speech (POS) tags instead of words. We also propose approaches to combine the standard word vectors with the new POS vectors in order to leverage the information captured by both. An early version of this chapter has been published in [50].

## 7.1 Suggestion Rich Data on Web

Due to the sparse nature of suggestion sentences in text from most of the domains, a large number of sentences turn out to be non-suggestions and thus manual annotation remains expensive. In the light of open domain training performing better or comparable to the domain specific training (Chapter 6), an intuition is to look for existing sources of text on the web, irrespective of the domain, where suggestions are already explicitly marked.

One potential source is web based suggestion forums for different products and services. An example of such suggestion forums is, a dedicated forum to convey the ideas for improvements in Microsoft Office 365, which is hosted by the uservoice platform (Figure 7.1). The uservoice platform provides customer feedback management service to brands. A first impression about this suggestion forum is that it will mostly contain positive class instances for a suggestion dataset and can be directly used without needing any further human annotations. However, we observe that it is not the case, since text from these posts also contain many elaborative and conversational sentences in addition to the explicitly expressed suggestions, where non-suggestions comprise of more than 50% of the sentences in the posts.

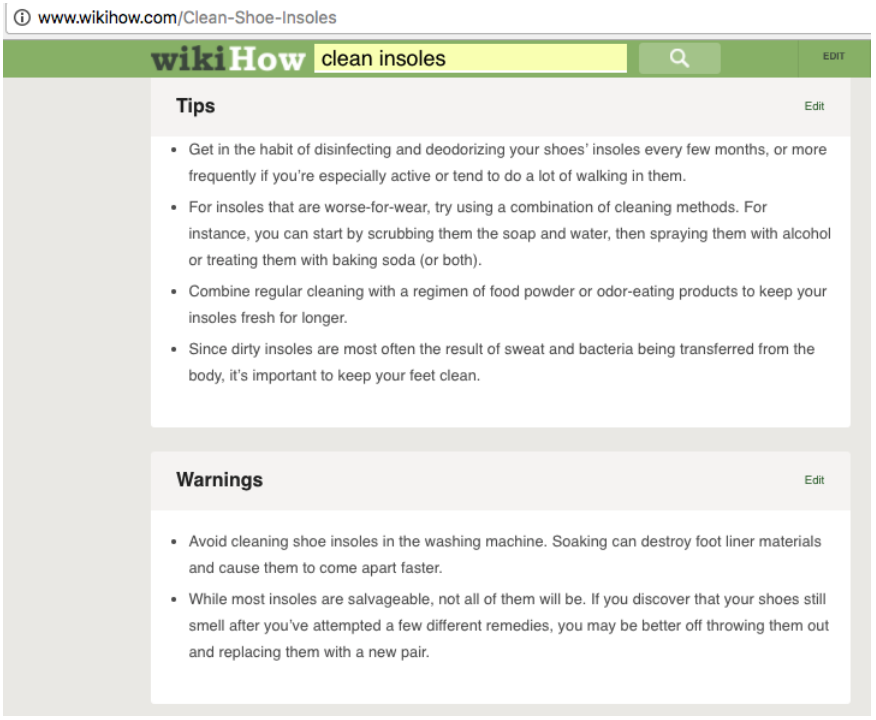



FIGURE 7.2: Tips and Warning Sections of the Wikihow article on *How to Clean Shoe Insoles*

### 7.1.1 Wiki Suggestion Dataset

Collaborative content websites like Wikipedia have proven to be a very useful resource for various NLP research problems due to their wide coverage of knowledge and semi structured content. wikiHow is an online wiki-style community consisting of an extensive database of how-to guides, which spans across a large variety of topics. wikiHow articles range over factual instructions like *Clean Shoe Insoles*, to opinionated instructions like *Resolve Conflict Effectively*. Due to a wide coverage, many of wikiHow's topics may have significant overlap with the candidate domains in suggestion mining, like an article on 'Time-Your-Trip-to-Berlin' [1], may have similar content to that of suggestions in hotel review and travel forum domain. Each article always comprises of a *Steps* section which lists the main steps of doing a certain thing (topic of the page). In addition, there are optional sections like *Tips*, *Warnings*, and *Community Q and A*. The tips and warnings section contains short and to-the-point list of suggestions and advice related to the topic of the article, which are very much like the positive instances of our suggestion mining training datasets. For the negative sample, we randomly choose equal number of sentences from Wikipedia, since it mainly contains descriptive and factual text, and therefore is extremely less likely to contain expressions of suggestions.

[1] http://www.wikihow.com/Time-Your-Trip-to-Berlin

| Dataset | Source/ Domain | Sentences | Vocabulary | Tokens |
|---|---|---|---|---|
| Wiki suggestion | Wikihow | 675,851 | 156,768 | 10,739,320 |
| | Wikipedia | 675,851 | 548,505 | 20,863,958 |

TABLE 7.1: Details of the wiki suggestion dataset

There are two main reasons to consider this dataset as a silver standard. Firstly, we do not perform a manual check on each sentence, and the labels may not fully adhere to the annotation guidelines used for the gold standard datasets. Secondly, the negative sample from Wikipedia is much cleaner as compared to the ambiguous negative instances in the gold standard datasets, for example, implicit non-suggestions which possess linguistic properties similar to that of explicit suggestions.

### 7.1.2 Pre-processing

We obtain the wikiHow dataset from a previous study by Pareti et al. [53], under which the original data was downloaded from wikiHow during July 2014. Not all the articles contain a tips and warnings section. We extracted all the items under the available tips and warnings sections. These items can sometimes be longer than one sentence. We observe that the main advice is mostly provided in the first sentence, and the following sentences contain further details, which may not carry an explicit expression of suggestion. For example, 'Do not store Ugg boots when they are damp. This encourages mildew growth'. Therefore, we perform automatic sentence splitting of each item under the tips and warnings section and only retain the first sentence as a suggestion instance. The sentence splitting is performed using NLTK's sentence tokenizer [36]. Before performing the sentence split, we remove URLs, any further lists enclosed within an item of the main list of tips, and any content within brackets. This results in about 675,851 sentences, which was matched against the same number of sentences from Wikipedia, resulting in the final wiki suggestion dataset to contain about 1.3 million sentences. The used Wikipedia dump is also from 2014 [2]. Table 7.1 provides the statistics for the final dataset.

## 7.2 Using Wiki Suggestion Dataset

In the previous chapter, we performed experiments using pre-trained word embeddings and manually labeled datasets with LSTM classifiers. We further explore the usability of representation learning in suggestion mining with the much larger wiki dataset, using two different approaches.

[2] https://meta.wikimedia.org/wiki/Data_dumps

### 7.2.1 Approach 1: Wiki Suggestion Dataset for Training the Classifiers

In this approach, we induce distant supervision through the wiki dataset by employing it like a gold standard dataset, i.e., for training the classifier. Similar to the previous chapter, we use the vocabulary and initial weights of the word vectors from the pre-trained GLoVE vectors. The weights from GLoVE are updated each time we train a classifier; therefore, the words which are not present in the training corpus retain the weights from GLoVE. This approach is evaluated with the following variations in the training data:

1. **Wiki dataset only:** The full wiki suggestion dataset is used for training.
2. **Wiki and manually labeled dataset:** Training is performed in two passes. The classifier is first trained on the entire wiki dataset. The wiki trained model is then further trained on the manually tagged dataset relevant for the experiment.
3. **Semantic subsample of wiki dataset:** A subset of wiki dataset which is semantically related to the evaluation domain is used for training. The subset is obtained in the following steps:
   (a) Obtain a set of all the unique words in the test dataset.
   (b) Obtain a vector of these words by performing an addition of the corresponding word vectors.
   (c) Obtain a vector for each sentence in the wiki dataset by using the same method as above.
   (d) Calculate the similarity score of the test set vector with each of the wiki sentence vector.
   (e) Choose the top 1% similar sentences from both the classes of the wiki dataset. This results in a balanced subset of about 13,500 sentences.
4. **Wiki subsample and manually labeled dataset:** Same as the variation no. 2, except that the semantic subsample of wiki dataset is employed in place of the full wiki dataset.

**Evaluation:** We only perform domain specific and open domain training with this approach, and cross domain training is ruled out with this set of experiments. This is because wiki dataset is an open domain dataset, which is likely to contain text related to the evaluation domain.
Based on our evaluation from the previous chapter, 200 dimensional GLoVE vectors are employed for domain specific training and 50 dimensions for open domain training.

| Domain | WikiHow | Wikipedia |
|---|---|---|
| Hotel | Even though most of the Universal Orlando Parks have a few areas where kids can explore and run out all their excitement, this may sometimes not be enough. | Several important people of the city of Clermont had asked me to let them know when I would make the ascent. |
| Electronics | Create a back-up of the Adobe Epic folder before deleting it, or move the folder to a separate location | When I'm writing a song, I'm so in that zone that it's really sort of a trance I go into, she explained. |
| Travel | If you reacted a certain way because they did a certain thing, you'll want to bring it up tactfully. | So I thought,'ok, I'll carry on doing this for a bit and the next thing you know that's how I make my living these days. |
| Software | At this time, Keek only allows users to delete their accounts from the Keek.com website, however, this feature will soon be available for Keek apps on iOS, Android, Blackberry 10, and Windows 8 devices. | The HD's user interface, he noted, was the first such Microsoft product to rely on text rather than icons, and it would form the basis for Windows Phone, Windows 8, Xbox and all of the company's web-based services. |

TABLE 7.2: Examples of sentences from the two classes in the Wiki dataset with the highest similarity score with different test datasets.

**Baseline:** The baseline for this approach is to only use the manually labeled dataset.

## Results

Tables 7.3 and 7.6 show the results of using wiki dataset as a training dataset with the proposed variations. The baseline approach remains the best in the case of domain specific training, except for the electronics domain. In the case of open domain training, using the full wiki dataset without combining any manually tagged dataset proves to perform better in three out of four domains. Using Wiki dataset without combining it with any manually labeled dataset outperforms domain specific training for the electronics and travel domain.

| Train data | Word vector | Hotel Test | | | Electronics Test | | |
|---|---|---|---|---|---|---|---|
| | | P | R | F1 | P | R | F1 |
| **Domain specific** | | | | | | | |
| Domain data (baseline) | $Glove_{200}$ | 0.76 | 0.95 | **0.84** | 0.33 | 0.94 | 0.49 |
| Wiki subsample | $Glove_{200}$ | 0.95 | 0.26 | 0.41 | 0.80 | 0.36 | 0.49 |
| Wiki subsample + manually labeled | $Glove_{200}$ | 0.98 | 0.52 | 0.68 | 0.96 | 0.44 | **0.60** |
| **Open domain** | | | | | | | |
| All domains | $Glove_{50}$ | 0.96 | 0.64 | 0.77 | 0.88 | 0.58 | 0.70 |
| Wiki | $Glove_{50}$ | 0.75 | 0.89 | **0.82** | 0.66 | 0.87 | **0.75** |
| Wiki + all | $Glove_{50}$ | 0.95 | 0.55 | 0.70 | 0.83 | 0.39 | 0.53 |

TABLE 7.3: Results of directly using Wiki dataset for training of the classifier

| **Train data** | **Word vector** | **Travel Test** | | | **Software Test** | | |
|---|---|---|---|---|---|---|---|
| | | P | R | F1 | P | R | F1 |
| **Domain specific** | | | | | | | |
| Domain data (baseline) | $Glove_{200}$ | 0.68 | 0.43 | **0.53** | 0.83 | 0.81 | **0.82** |
| Wiki subsample | $Glove_{200}$ | 0.80 | 0.37 | 0.50 | 0.47 | 0.60 | 0.53 |
| Wiki subsample + manually labeled | $Glove_{200}$ | 0.69 | 0.38 | 0.49 | 0.86 | 0.76 | 0.81 |
| **Open domain** | | | | | | | |
| All domains (baseline) | $Glove_{50}$ | 0.74 | 0.51 | 0.60 | 0.81 | 0.80 | 0.80 |
| Wiki | $Glove_{50}$ | 0.62 | 0.75 | **0.68** | 0.52 | 0.70 | 0.59 |
| Wiki + all | $Glove_{50}$ | 0.68 | 0.44 | 0.53 | 0.87 | 76 | **0.81** |

TABLE 7.4: Results of directly using Wiki dataset for training of the classifier

### 7.2.2 Approach 2: Wiki Suggestion Dataset for Learning Word Representations

In this approach, we use the wiki dataset to incorporate supervision in learning suggestion specific word embeddings. We aim to encode task specific information in the word vectors itself, which may enable them to compose with other words in a sentence in a way that the two classes of sentences become better separable.

We first train word embeddings using the wiki suggestion dataset, replace the pre-trained GLoVE vectors in the previous approach with these vectors, and train the classifier using a manually labeled dataset. We also introduce a novel concept of Part of Speech tag embeddings against the standard notion of word embeddings. The POS embeddings are also learned from the Wiki dataset. This follows from our observation on the effectiveness of syntactic features for this task. Following are the different variations we employed to learn the word embeddings.

1. **Wiki Suggestion Embeddings (WiSE)**: We train a LSTM classifier using the full wiki suggestion dataset, without using any pre-trained word vectors for initializing the training process. Word weights are then extracted from the input layer at the end of the training. The network architecture remains the same as that of the previously used LSTM classifiers. We refer to these embeddings as the Wiki Suggestion Embeddings (WiSE). These embeddings are different from the standard co-occurrence based embeddings (GloVE, Word2vec etc.) in the sense that these are learned using a supervised learning objective. We train two variations of the WiSE embeddings:

    - $WiSE_w$: These are the vectors for words. The vocabulary comprises of words which are present in the Wiki suggestion dataset, and is therefore smaller

than that of the GLoVE vectors. These embeddings are trained for 200 and 50 dimensions.

- WiSE$_p$: These are the vectors representing POS tags. These embeddings are very light weight, with 50 dimensions and a vocabulary of 34 tags when trained on wiki suggestion dataset. These are learned using the same method as above, only that the words in the Wiki dataset are replaced with their respective POS tags. NLTK's [10] POS tagger was used to replace words in the Wiki suggestion dataset with their automatically identified POS tags. Usually it is seen that the vector size representing a word in a language is smaller than the size of its vocabulary. Although the POS vocabulary count in the wiki dataset resulted in 34 tags, we chose a vector size (50) which is larger than the vocabulary for two reasons. First is to accommodate vocabulary resulting from incorrect tokenizing and PoS tagging, eg., *)"*, *:"*. Secondly, the smallest vector size for pre-trained word embeddings is 50, and therefore for the sake of uniformity we chose to keep the same number of dimensions for PoS embeddings.

**Baseline:** The baseline for these embeddings are the pre-trained GloVE vectors of the same dimension.

**Evaluation:** We perform domain specific, open domain, and cross domain training with this approach. Based on our evaluation from the previous chapter, 200 dimensional GLoVE vectors are employed for domain specific training and 50 dimensions for open domain and cross domain training.

### Results

Tables 7.5 and 7.6 show the results of using WiSE vectors in the domain specific, open domain, and cross domain settings. The WiSE$_p$ vectors gave the best performance in more number of experiments than any pf the other vectors, whereas WiSE$_w$ vectors never outperformed any of the other vectors. Specially for the electronics and travel domain, WiSE$_p$ performed best for all three training scenarios.

## 7.3 Learning with Two Types of Embeddings

In the results reported above, we observe that the best performance is mostly shared between the GLoVE vectors (word vectors) and the WiSE$_p$ vectors (POS vectors). Figures

| **Embedding** | **Hotel Test** | | | **Electronics Test** | | |
|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 |
| **Domain specific training** | | | | | | |
| $Glove_{200}$ (Baseline) | 0.76 | 0.95 | **0.84** | 0.33 | 0.94 | 0.49 |
| $WiSE_{w,50}$ | 1.00 | 0.51 | 0.68 | 0.87 | 0.46 | 0.60 |
| $WiSE_{w,200}$ | 0.99 | 0.43 | 0.60 | 0.97 | 0.31 | 0.47 |
| $WiSE_{p}$ | 0.82 | 0.81 | 0.81 | 0.71 | 0.77 | **0.74** |
| **Open domain training** | | | | | | |
| $Glove_{50}$(Baseline) | 0.98 | 0.64 | 0.77 | 0.88 | 0.58 | 0.70 |
| $WiSE_{w,50}$ | 0.96 | 0.63 | 0.76 | 0.83 | 0.54 | 0.66 |
| $WiSE_{w,200}$ | 0.95 | 0.60 | 0.74 | 0.81 | 0.51 | 0.63 |
| $WiSE_{p}$ | 0.87 | 0.76 | **0.81** | 0.78 | 0.81 | **0.80** |
| **Cross domain training** | | | | | | |
| $Glove_{50}$ (Baseline) | 0.78 | 0.75 | **0.76** | 0.73 | 0.66 | 0.69 |
| $WiSE_{w,50}$ | 0.76 | 0.70 | 0.73 | 0.77 | 0.64 | 0.70 |
| $WiSE_{w,200}$ | 0.78 | 0.70 | 0.74 | 0.75 | 0.53 | 0.62 |
| $WiSE_{p}$ | 0.86 | 0.64 | 0.73 | 0.70 | 0.86 | **0.77** |

TABLE 7.5: Classification results of using different types of word vectors

| **Embedding** | **Travel Test** | | | **Software Test** | | |
|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 |
| **Domain specific training** | | | | | | |
| $Glove_{200}$(Baseline) | 0.68 | 0.43 | 0.53 | 0.83 | 0.81 | **0.82** |
| $WiSE_{w,50}$ | 0.73 | 0.41 | 0.53 | 0.88 | 0.75 | 0.81 |
| $WiSE_{w,200}$ | 0.66 | 0.54 | 0.59 | 0.50 | 1.00 | 0.67 |
| $WiSE_{p}$ | 0.65 | 0.79 | **0.72** | 0.71 | 0.69 | 0.70 |
| **Open domain training** | | | | | | |
| $Glove_{50}$ (Baseline) | 0.74 | 0.51 | 0.60 | 0.81 | 0.80 | **0.80** |
| $WiSE_{w,50}$ | 0.71 | 0.49 | 0.58 | 0.83 | 0.78 | **0.80** |
| $WiSE_{w,200}$ | 0.73 | 0.45 | 0.56 | 0.82 | 0.76 | 0.79 |
| $WiSE_{p}$ | 0.70 | 0.86 | **0.77** | 0.62 | 0.91 | 0.74 |
| **Cross domain training** | | | | | | |
| $Glove_{50}$(Baseline) | 0.77 | 0.44 | 0.56 | 0.61 | 0.24 | 0.34 |
| $WiSE_{w,50}$ | 0.83 | 0.30 | 0.44 | 0.65 | 0.34 | 0.44 |
| $WiSE_{w,200}$ | 0.77 | 0.37 | 0.50 | 0.56 | 0.21 | 0.30 |
| $WiSE_{p}$ | 0.68 | 0.87 | **0.77** | 0.60 | 0.94 | **0.73** |

TABLE 7.6: Classification results of using different types of word vectors

7.3 and 7.4 are the plots of hotel test data instances using GLoVE and $WiSE_p$ vectors respectively, where sentence vectors are obtained from the first LSTM layer. It is evident that the positive and negative classes have been separated upto some extent in both the cases, but using different kind of functions. Therefore, both types of embeddings may possess complementary properties for this task. This motivates us to use both of these embeddings to learn the classification models. Below we propose two methods for the same:

**Concatenation of Word and POS embeddings:** In this case, the network architecture for the LSTM classifier remains unchanged. We simply concatenate the GLoVE and $WiSE_p$ vectors for a given word and then feed the resulting vector to the learning algorithm, like in a normal case of using only one type of word vector. Figure

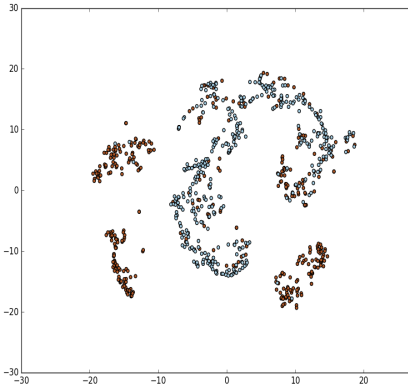


FIGURE 7.3: Class separation in hotel test dataset when the model is trained using $\text{GLoVE}_{200}$ vectors.

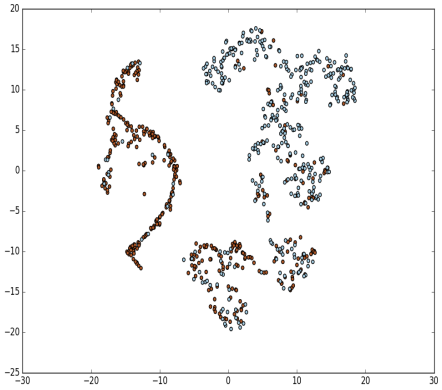


FIGURE 7.4: Class separation in hotel test dataset when the model is trained using $\text{WiSE}_p$ vectors.

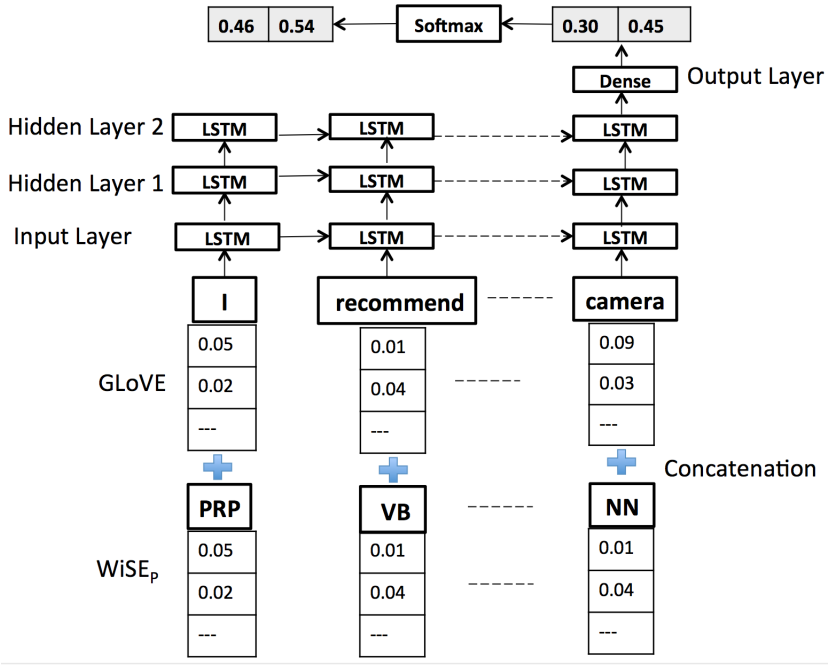



FIGURE 7.5: Simple concatenation of embeddings

7.5 illustrates this method. The best performing dimensions of GlOVE are used based on the observations from the previous experiments.

**Learning with two input vectors:** In this method, we input the two pre-trained vectors for each word separately and combine them midway the neural network architecture. Figure 7.6 illustrates the network architecture. This architecture is different from the architecture used so far, as it replaces the two LSTM hidden layers with a fully connected dense layer. We choose this architecture in order to reduce the architectural

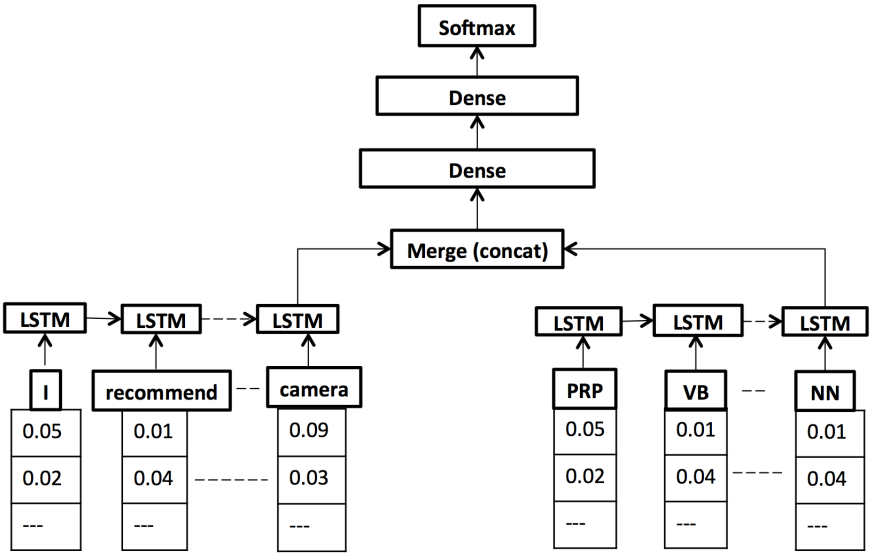



FIGURE 7.6: Learning to use two embeddings as input to the network

complexities caused by combining two sub-networks in time steps (LSTM). We envisage that this method will lead the neural network to be trained with the constraint of using two different inputs vectors and combining them in the merge layer, which will lead to the two vectors gaining optimal weights before they are combined.

### 7.3.1 Results

Tables 7.7 and 7.8 show the comparison of performance of the two methods employed to use the two kind of word vectors with a single classifier. The method where two input vectors are incorporated in the model appears to perform better in most of the cases, and in all the cases when cross domain training is employed.

| **Method** | **Hotel Test** | | | **Electronics Test** | | |
|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 |
| **Domain specific training** | | | | | | |
| Concat | 0.97 | 0.58 | 0.73 | 0.90 | 0.44 | 0.59 |
| Learn | 0.87 | 0.85 | **0.86** | 0.82 | 0.59 | **0.69** |
| **Open domain training** | | | | | | |
| Concat | 0.95 | 0.53 | 0.68 | 0.81 | 0.50 | 0.61 |
| Learn | 0.84 | 0.82 | **0.83** | 0.78 | 0.57 | **0.66** |
| **Cross domain training** | | | | | | |
| Concat | 0.82 | 0.40 | 0.54 | 0.76 | 0.45 | 0.56 |
| Learn | 0.84 | 0.79 | **0.81** | 0.75 | 0.78 | **0.76** |

TABLE 7.7: Results obtained from the two methods of combining the GLoVE and $\text{WiSE}_p$ embeddings

| Method | Travel Test | | | Software Test | | |
|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 |
| **Domain specific training** | | | | | | |
| Concatenation | 0.59 | 0.31 | 0.41 | 0.85 | 0.77 | **0.81** |
| Two input model | 0.60 | 0.73 | **0.66** | 0.55 | 0.80 | 0.65 |
| **Open domain training** | | | | | | |
| Concatenation | 0.72 | 0.41 | **0.52** | 0.85 | 0.77 | **0.81** |
| Two input model | 0.80 | 0.26 | 0.40 | 0.57 | 0.96 | 0.71 |
| **Cross domain training** | | | | | | |
| Concatenation | 0.78 | 0.36 | 0.49 | 0.59 | 0.30 | 0.40 |
| Two input model | 0.71 | 0.56 | **0.62** | 0.67 | 0.62 | **0.64** |

TABLE 7.8: Results obtained from the two methods of combining the GLoVE and WiSE$_p$ embeddings

## 7.4 Analysis of Results

Table 7.9 summarizes the results obtained from all the experiments reported in the previous sections, which achieved best F1 score in at-least one run. In the first approach, using only semantically related wiki subsample never achieved best F1 score. In the second approach, WiSE word vectors never outperformed any other vector in a cross domain setting. These two results are therefore not included in the summary table.

It can be observed that the induction of distant supervision through the wiki dataset always outperformed the baseline in all the three training scenarios. Exception being the domain specific training for the software domain, where the results achieved by distant supervision are lower than the baseline with a very small margin. Most interesting results are the high performance of WiSE$_p$ embeddings in all the three settings. These POS embeddings outperform GLoVE in a majority of experiments, provided its much smaller vocabulary (Table 7.10). Figure 7.7 shows a two dimensional plot of the WiSE$_p$ embeddings. We can see that the tags belonging to the same part of speech are not grouped together, as opposed to the word embeddings where semantically similar words are often located close to each other. Figure 7.9 shows the gradual separation of classes for the hotel test dataset when moving up the layers of a trained network using WiSE$_p$ embeddings. Given the very small number of features (34), the curve fits very well. These results again demonstrate the importance of syntax and grammar in the task of suggestion mining.

**Approach 1 vs Approach 2:** Approach 1, i.e., using the full Wiki dataset outperforms manually labeled dataset in an open domain training scenario, except for the software domain. However, approach 2, i.e. using WiSE$_p$ in combination with the manually labeled datasets yields better classification performance for the open domain training. Therefore, the models learned using Wiki dataset only, can be worth trying

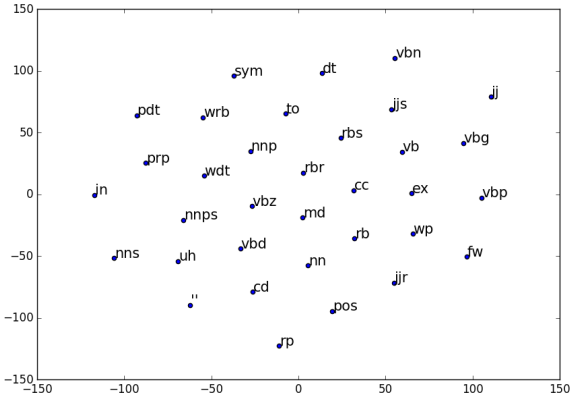



FIGURE 7.7: A two dimensional projection of WiSE$_p$ embeddings

| **Training data** | **Word Vector** | **Hotel** | **Electronics** | **Travel** | **Software** |
|---|---|---|---|---|---|
| **Domain specific training** | | | | | |
| Manual | Glove$_{200}$ | 0.84 | 0.49 | 0.53 | **0.82** |
| Related Wiki subsample, manual | Glove$_{200}$ | 0.68 | 0.60 | 0.49 | 0.81 |
| Manual | WiSE$_p$ | 0.81 | **0.74** | **0.72** | 0.70 |
| Manual | GLoVE$_{200}$,WiSE$_p$ (concat) | 0.73 | 0.59 | 0.41 | 0.81 |
| Manual | GLoVE$_{200}$,WiSE$_p$ (learn) | **0.86** | 0.69 | 0.66 | 0.65 |
| **Open domain training** | | | | | |
| All manual | Glove$_{50}$ | 0.77 | 0.70 | 0.60 | 0.80 |
| Wiki | Glove$_{50}$ | 0.82 | 0.75 | 0.68 | 0.59 |
| Wiki, All manual | Glove$_{50}$ | 0.70 | 0.53 | 0.53 | **0.81** |
| All manual | WiSE$_p$ | 0.81 | **0.80** | **0.77** | 0.74 |
| All manual | GLoVE$_{200}$, WiSE$_p$ (concat) | 0.68 | 0.61 | 0.52 | **0.81** |
| Manual | GLoVE$_{200}$, WiSE$_p$ (learn) | **0.83** | 0.66 | 0.40 | 0.71 |
| **Cross domain training** | | | | | |
| Manual | GLoVE$_{50}$ | 0.76 | 0.69 | 0.56 | 0.34 |
| Manual | WiSE$_p$ | 0.73 | **0.77** | **0.77** | **0.73** |
| Manual | GLoVE$_{200}$, WiSE$_p$ (learn) | **0.81** | 0.76 | 0.62 | 0.64 |

TABLE 7.9: A comparison of F1 scores achieved in all the experiments in this chapter. First row for each type of training is the baseline method which do not involve any distant supervision. *Manual* stands for manually labeled datasets.

| **Embedding** | **Dimensions** | **Vocabulary** |
|---|---|---|
| Glove (Baseline) | 100 | 400,000 |
| WiSE$_w$ | 100 | 339,240 |
| Part of Speech Embeddings | | |
| Glove (Baseline) | 50 | 6B |
| Glove-Wiki$_p$ | 50 | 34 |
| WiSE$_p$ | 50 | 34 |
| Glove + WiSE$_p$ Concat | 150 | 6B * 34 |

TABLE 7.10: Statistics of different embeddings used

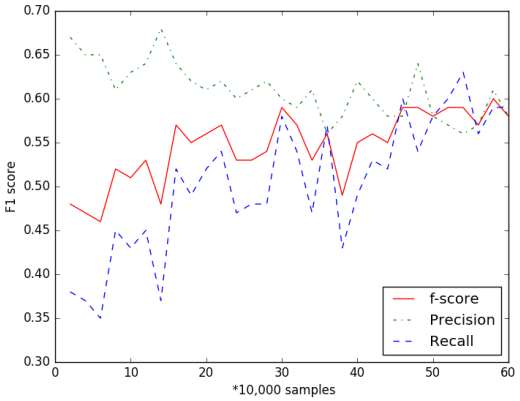



FIGURE 7.8: Training sample size of wiki dataset vs. precision, recall and F1 score on the test dataset for the travel domain

for new domains whose training dataset is not available. In the open domain training, travel domain achieves best improvement when Wiki dataset is used over the manually labeled dataset. Figure 7.8 shows the variations in F score for the travel test dataset, on increasing the training sample size of the Wiki dataset. F 1 score improves with large additions to the training datasets.

## 7.5 Summary

In this chapter we proposed some methods to employ distant supervision in suggestion mining with the aid of representation learning. A large silver standard and open domain training dataset comprising of 1.3 million sentences is developed using the sentences from wikiHow and Wikipedia. The two approaches to use this new wiki suggestion dataset perform better than the baseline. Therefore, the Wiki suggestion dataset proves to be an important resource for suggestion mining. In future, as the wikihow dataset grows, we expect an improvement in the performance of the proposed approaches. However, it is also evident that in order to achieve higher classification accuracies, manually labeled data would still be required.

In addition to the improvement in performance of the baseline LSTM classifiers, our experiments led to two interesting outcomes. The first is the demonstrated effectiveness of POS tag embeddings, which were learned using a supervised method on a silver standard training dataset. This results in an extremely light weight yet powerful classification model as compared to the other neural network models discussed in this thesis. To the best of our knowledge, this is a first of its kind study of the performance of POS tag embeddings on a sentence classification task. It provides a basis for evaluating this

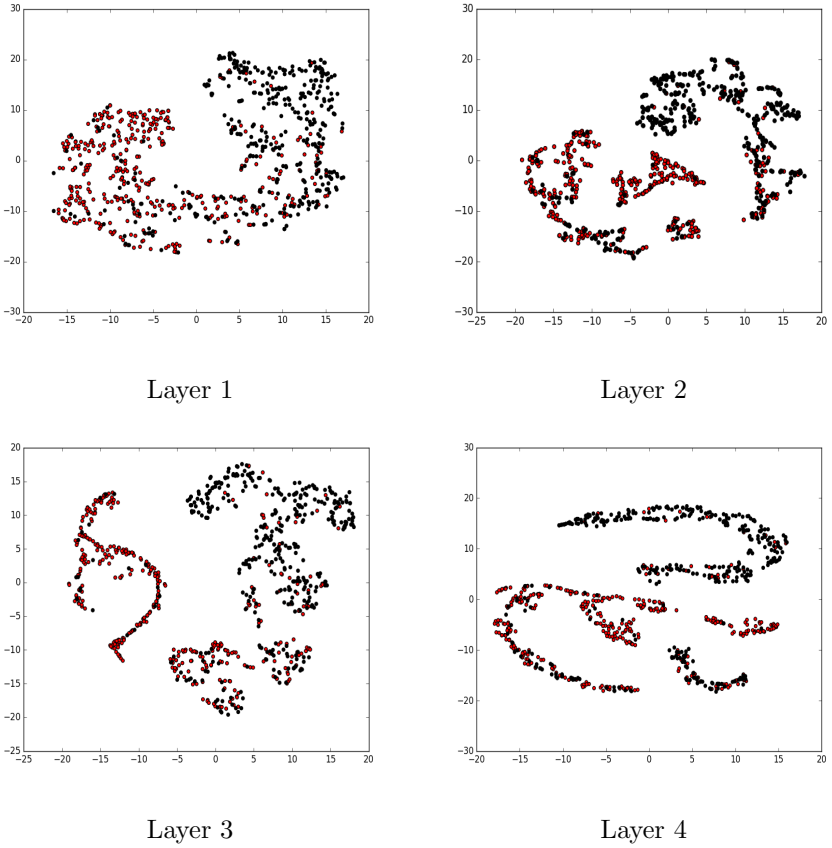



FIGURE 7.9: Separation of positive and negative classes in hotel test dataset, when moving up the layers of a trained LSTM classifier which uses WiSE$_p$ vectors

method on other sentence level tasks. The second is the achievement of high classification accuracy in open domain and cross training scenarios, which opens the possibility of directly applying the existing open domain and cross domain models to new domains.

The method of employing two types of inputs in the neural network proved to be better than concatenating them beforehand. But this method still does not perform better than the single vector method for most of the domains, and therefore failed to effectively complement the information gained from the two vectors. Exploring and evaluating more approaches to use both word and POS embeddings can be a future extension of the study conducted in this chapter.

# Chapter 8

# Conclusion

In this thesis we investigated the extraction of suggestion expressing sentences from opinionated text, like reviews, discussion forums etc., which can be referred to as *suggestion mining*. Several datasets were also developed and released for the same. With this chapter, we conclude the thesis by providing a summary of the contributions delivered in chapters 4, 5, 6, and 7. We also reflect on the limitations of the current study, possible extensions to the experiments, and directions for the future work in suggestion mining. Lastly, we describe two openly available suggestion mining services, which were developed by us for demonstration purposes.

## 8.1 Summary of Contributions

The investigations in this thesis were guided by three research questions, *What are suggestions in the context of suggestion mining?*, *What are good features for supervised learning in suggestion mining?*, and *How can distant supervision be employed in suggestion mining?* Below we discuss our contributions towards answering each of these questions.

### 8.1.1 Defining Suggestions

In this part, we first studied qualitative and quantitative properties of suggestions and the challenges associated with the manual labeling of sentences. We then extended the formal definition of suggestion mining by providing a categorization of sentences present in the source text, and by defining the scope of suggestion and non-suggestion classes in terms of these categories. The categories were, *explicit suggestions, implicit suggestions, explicit non-suggestions, and implicit non-suggestions*. A context based definition of

each of the categories was also proposed, with some examples of context. The scope of suggestion class for this thesis was defined as *explicit suggestions*, while non-suggestion class comprised of sentences from the rest of the three categories.

Based on this theory, we devised a two phase annotation method to prepare benchmark datasets for suggestion mining. This method reduced the subjectivity associated with the manual labeling of sentences, with minimally defined annotation guidelines, which if very detailed tend to be ignored by the annotators. Training and evaluation datasets were prepared for four domains: hotel reviews, electronics reviews, travel discussion forum, and software suggestion forum. In addition, two datasets (travel forum and tweets) from the previous works were re-annotated with our method in order to employ them for the current experiments. Different domains differed in the way suggestions were expressed in them, the ratio of classes, and the level of inter-annotator agreement.

**Limitations and future extensions**

Although we investigated four domains, this study can be expanded to several other application domains, and the task definition can be revisited accordingly. One of the additional domains which we are interested in is Youtube, where suggestions can be extracted from the comments section. We observe that several Youtube channel owners upload videos on demand, where they have to manually read all the comments to identify the most frequent request. This thesis also limits the scope of study to explicit suggestions, and extending the scope to implicit suggestions can introduce further challenges and use cases to the task. Implicit suggestion mining can also cross paths with the field of natural language inference [37], where suggestions need to be inferred from their implicit forms. For examples, 'There is this lovely English cafe next to the hotel, which serves fresh scones and cakes every morning.', can be considered as an implicit suggestion, which can be used to infer an explicit suggestion, 'I recommend the English cafe next to the hotel.'

One limitation of the current task being framed as a sentence classification task is that automatic sentence classification tends to be error prone, specially with the noisy text. Also, often context and suggestions are either split into two consecutive sentences or are only present within a single clause of a long sentence. For example, 'So far i have tried three different apps and I cannot access my pre windows phone 8.1 photos.', 'This needs fixing as a number 1 priority' are two consecutive sentence from the software domain. An alternate framing of the task could be the identification of start and end position of a suggestion in the full candidate document.

### 8.1.2 Features

We performed the first in-depth study and comparison of the performance of manually defined features, as well as automatically learned features for suggestion mining. SVM and LSTM neural network classifiers were employed for the same. All the features were evaluated for their performance in three types of training scenarios, domain specific, open domain, and cross domain training. The automatically learned features included pre-computed word vectors which can be learned using different methods and can possess different number of dimensions. We compared the performance of two popular types (GLoVE and Dependency), as well as 4 different size of dimensions (300, 200, 100, 50) for word vectors.

Our findings show that the manually determined features are at par with the automatically learned features for three out of four domains, given the currently available manually tagged datasets. Apart from the n-gram features, majority of the top ten manual features were proposed by us. When word vectors were employed, SVM classifiers performed better than LSTM classifiers in the majority of cases. Interestingly, automatic feature learning worked best for the travel domain which was seen to be the most difficult domain to manually distinguish suggestions from non-suggestions. The classifiers from related works which used SVM with manually determined features, were outperformed when SVM was used with word vectors.

We also proposed our own word vectors WiSE, which further improved the results for representation learning for all the domains. WiSE have been summarized in the next section. Table 8.1 summarizes the overall best performing classifiers and features for different domains, across all types of training. Hotel, electronics, and travel domain evaluation datasets yielded best results when training datasets from all the domains were combined. Therefore, the manually defined features were able to perform well on combining training datasets from different domains.

Another useful set of results are from the cross domain training, in which a classifier is trained on manually labeled datasets which exclude the evaluation domain. Manual features again gave best results for this setting, except for the travel domain.

**Limitations and future extensions**

A limitation of the current work is the hand tuning (trial and error) of hyperparameter values, which can be replaced with more automated, exhaustive, and advanced methods like

The observation that manual features work better with the relatively small manually

| Domain | Training data | Classifier | Features | F1 score |
|---|---|---|---|---|
| Hotel | Open domain, manually labeled | SVM | Manual | 0.891 |
| Electronics | Open domain, manually labeled | SVM | Manual | 0.894 |
| Travel | Open domain, manually labeled | LSTM | $WiSE_p$ | 0.77 |
| Software | Domain specific, manually labeled | SVM, LSTM | Manual (SVM), $GLoVE_{200}$ (LSTM) | 0.82 |

TABLE 8.1: Best performing experiment for different domains

| Domain | Classifier | Features | F1 score |
|---|---|---|---|
| Hotel | SVM | Manual | 0.84 |
| Electronics | SVM | Manual | 0.848 |
| Travel | LSTM | $WiSE_p$ | 0.77 |
| Software | SVM | Manual | 0.767 |

TABLE 8.2: Best performing methods for each domain in the case of cross domain training

labeled training datasets for suggestion mining can be further validated by using a different kind of neural network classifier. An immediate choice would be Convolutional Neural Networks, which do not attempt to learn long term dependencies in sentences (LSTM), but are known to effectively capture local co-relations of spatial or temporal structures. Therefore, an intuition is that CNN might capture well the good n-gram features (spatial structures) at different positions in a sentence.

### 8.1.3 Distant Supervision

In this part of our research, we evaluate methods where non-manually labeled but larger datasets can be employed to train better classifiers for suggestion mining. Another objective was to better employ representation learning by increasing the size of training dataset. We released a silver standard dataset of 1.3 million sentences which were freely collected from wikiHow and Wikipedia, and used it with the LSTM classifiers.

We proposed two approaches to employ the wiki suggestion dataset. First was to use the dataset directly for training the classifier, in addition or without the available manually labeled data. The other was to learn word embeddings in a supervised manner using the wiki dataset (WiSE vectors), and employ them in training the classifiers with the manually labeled datasets. We also proposed to learn POS version of WiSE vectors ($WiSE_p$), where vectors were learned for POS tags instead of words. The $WiSE_p$ vectors used with LSTM significantly improved the classification performance achieved by baseline LSTM classifiers in a majority of cases, across the three training scenarios, for all the domains. Combining $WiSE_p$ vectors with GLOVE vectors gave best results for the hotel domain across all three training scenarios for the LSTM classifiers. Using $WiSE_p$

with cross domain and open domain training of LSTM classifiers outperformed all other approaches for the travel domain (Tables 8.1, 8.2).

**Limitation and future extensions**

In future, we are interested to devise more methods for coupling WiSE$_p$ embeddings with the state of the art word vectors through different neural network architectures. We would also like to evaluate the approach of learning POS tag embeddings using silver standard datasets on other sentence classification problems.

## 8.2 Directions for new research problems

We discussed potential further experiments regarding each of the research questions previously. On a higher level, there are two additional research questions we are interested in pursuing with regards to suggestion mining.

**Multilinguality**

Given the decent results obtained by leveraging data from wikiHow and using small gold standard datasets, we are motivated to perform multilingual counterpart of the distant supervision experiments (Chapter 7), using multilingual wikiHow (Figure 8.1) and Wikipedia to obtain a silver standard, open domain, and multilingual dataset for the same.

**Suggestion summarisation**

Once suggestions are extracted from a given source text, there is a possibility that same suggestion is repeated a number of times when the source data is very large. For example, *Make sure to buy a tripod and separate flash unit at the same time*, and, *I would recommend buying a tripod as an accessory for professional quality pictures*, are essentially same suggestions extracted from the reviews of a certain camera. Therefore, it would be a useful to investigate into the identification of differently paraphrased suggestions, and summarizing them by either choosing a representative sentence or generating one. Existing state of the art methods on sentence similarity, text summarization and generation can be investigated for the same.

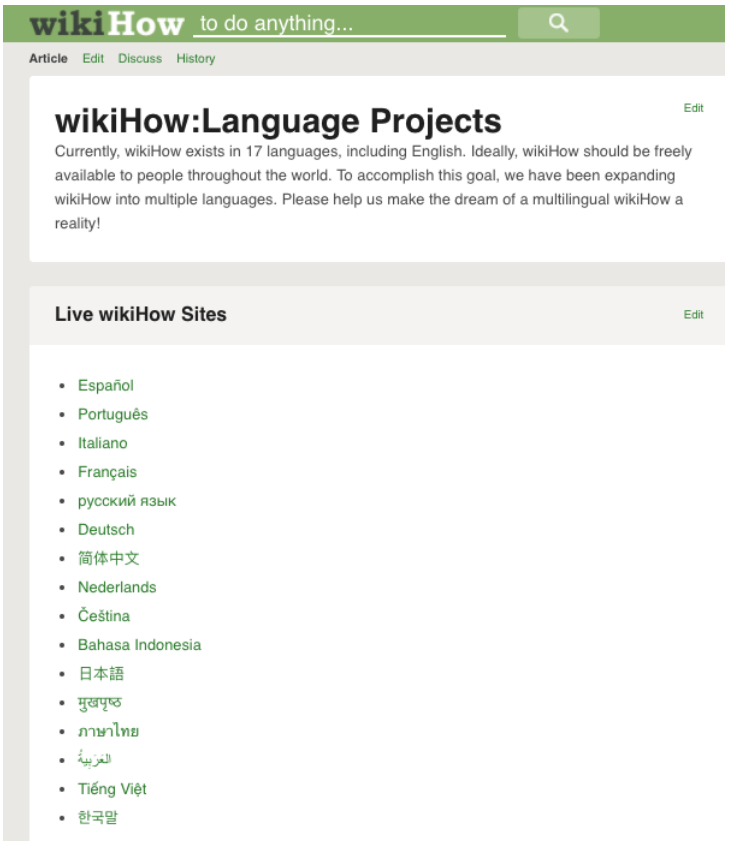



FIGURE 8.1: Availability of non-English versions of WikiHow

## 8.3 Demonstration and Services

### 8.3.1 Research Demonstrator: MiSO

We developed a demonstrator MiSO [1] for suggestion mining, which allows the user to provide a candidate text and compare between the predictions of a rule based classifier, support vector machines using manual features, and LSTM classifier. The rule based classifier is the one discussed in Chapter 5, which uses an aggregation of all the domain independent rules from the previous studies on suggestion mining. The statistical classifiers are trained on a combination of all manually labeled datasets. SVM uses all the manually identified features described in Chapter 5, while LSTM uses 50 dimensional GLoVE embeddings. As a part of the demonstrator, we also provide a rest API which accepts the source text in English and returns the sentences which are predicted as suggestions, using the LSTM classifier.

Currently, MiSO is developed for standard text (reviews, blogs etc.), but we plan to

[1] http://server1.nlp.insight-centre.org/miso/

extend the demonstrator with a dedicated service for tweets, which would involve additional pre-processing steps, and preparing twitter specific training dataset.

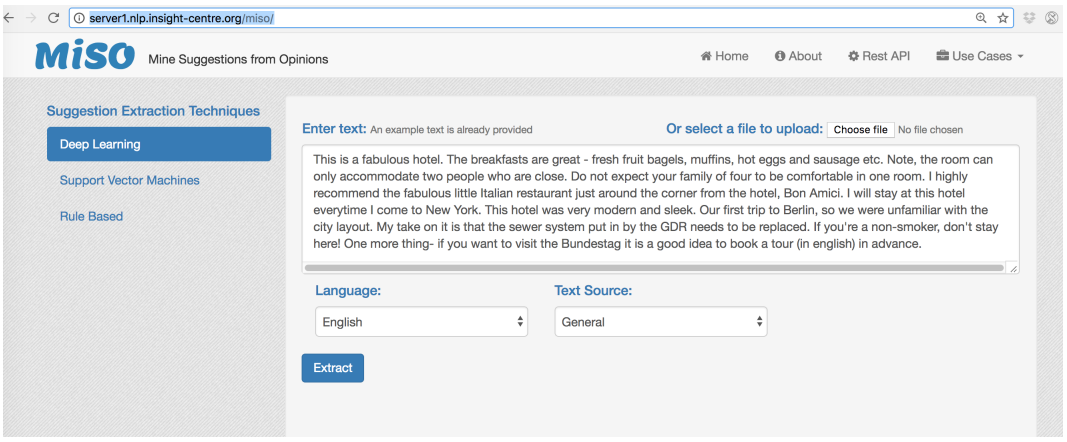



FIGURE 8.2: Home page of our demonstrator MiSO

### 8.3.2 MixedEmotions Platform

Mixed Emotions was a European Union project [2], which delivered open source services for the development of innovative multilingual multi-modal big data analytics applications. The projects culminated into the MixedEmotions platform [3], which is a big data toolbox for multilingual and multimodal emotion extraction and analysis, which can identify emotions from text, audio and video. In addition to emotion analysis, many other capabilities, such as sentiment analysis, social network analysis, suggestion mining etc. were also integrated in the platform [12]. The different modules are delivered by means of one or more of the following: source code, docker images, and rest API (with a user interface) (see Figure 8.3, 8.4). The suggestion mining module is delivered via all three modes. The rest API for suggestion mining provided with the Mixed Emotions platform accepts one sentence at a time and predicts whether it is a suggestion or not (Figure 8.4).

[2] https://mixedemotions-project.eu/
[3] http://mixedemotions.insight-centre.org/

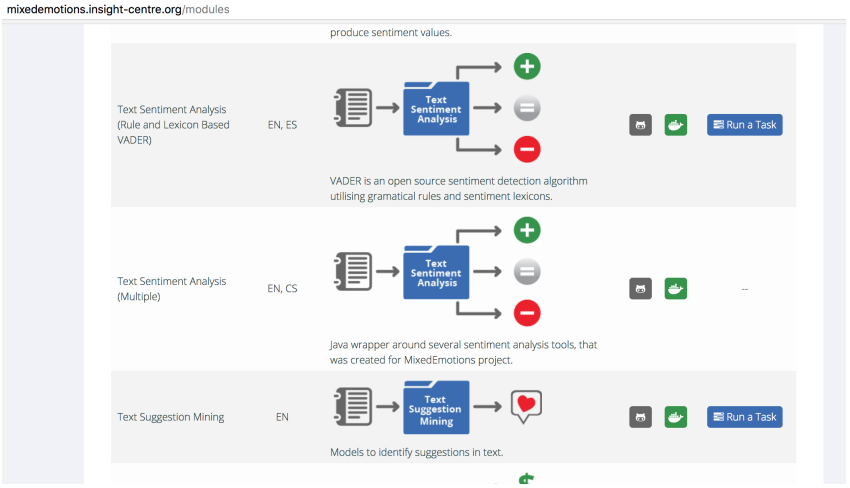



FIGURE 8.3: Some services provided in Mixed Emotions platform

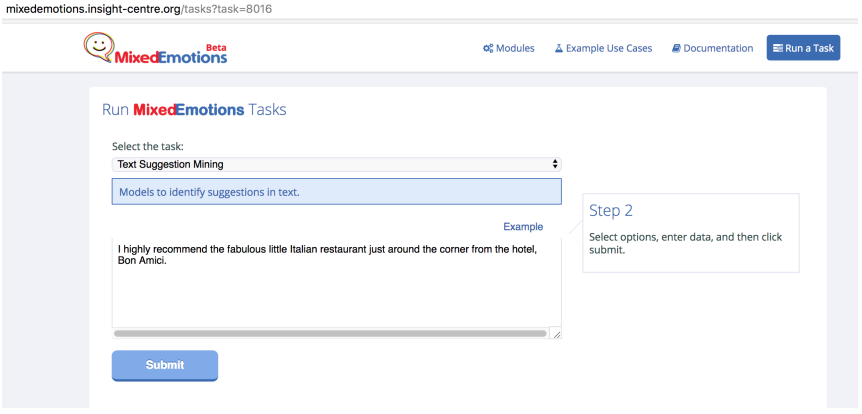



FIGURE 8.4: The suggestion mining service on MixedEmotions platform

# Bibliography


[1] Ron Artstein. Inter-annotator agreement. In *Handbook of linguistic annotation*, pages 297–313. Springer, 2017.

[2] Nicholas Asher, Farah Benamara, and Yannick Mathieu. Appraisal of Opinion Expressions in Discourse. *Lingvisticæ Investigationes*, pages 279–292, 2009.

[3] John Langshaw Austin. *How to do things with words*. Oxford university press, 1962.

[4] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *in Proc. of LREC*. Citeseer, 2010.

[5] K. Bach and R. Harnish. *Linguistic Communication and Speech Acts*. MIT Press, 1979.

[6] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98, pages 86–90. Association for Computational Linguistics, 1998.

[7] Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247. Association for Computational Linguistics, June 2014.

[8] Farah Benamara, Baptiste Chardon, Yannick Mathieu, Vladimir Popescu, and Nicholas Asher. How do negation and modality impact on opinions? In *Proceedings of the Workshop on Extra-Propositional Aspects of Meaning in Computational Linguistics*, ExProM '12, pages 10–18. Association for Computational Linguistics, 2012.

[9] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. *arXiv preprint arXiv:1206.5533*, 2012.

[10] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, Inc., 1 edition, 2009.

[11] Caroline Brun and Caroline Hagege. Suggestion mining: Detecting suggestions for improvement in users' comments. *Research in Computing Science*, 2013.

[12] Paul Buitelaar, Ian D. Wood, Sapna Negi, Mihael Arcan, John P. McCrae, Andrejs Abele, Cecile Robin, Vladimir Andryushechkin, Housam Ziad, Hesam Sagha, Maximilian Schmitt, Björn W. Schuller, J. Fernando Sanchez-Rada, Carlos Angel Iglesias, Carlos Navarro, Andreas Giefer, Nicolaus Heise, Vincenzo Masucci, Francesco A. Danza, Ciro Caterino, Pavel Smrx017E, Michal Hradix0161, Filip Povolnx00FD, Marek Klimex0161, Pavel Matx011Bjka, and Giovanni Tummarello. Mixedemotions: An open-source toolbox for multimodal emotion analysis. *IEEE Transactions on Multimedia*, 20:2454–2465, 2018.

[13] Ana Cardoso-Cachopo and Arlindo L Oliveira. An empirical comparison of text categorization methods. In *International Symposium on String Processing and Information Retrieval*, pages 183–196. Springer, 2003.

[14] Noam Chomsky. *Syntactic Structures*. Mouton and Co., The Hague, 1957.

[15] Corinna Cortes and Vladimir Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.

[16] D. Crystal. *A Dictionary of Linguistics and Phonetics*. The Language Library. Wiley, 2011.

[17] Marie-Catherine De Marneffe and Christopher D Manning. The stanford typed dependencies representation. In *Coling 2008: proceedings of the workshop on cross-framework and cross-domain parser evaluation*, pages 1–8. Association for Computational Linguistics, 2008.

[18] Li Dong, Furu Wei, Yajuan Duan, Xiaohua Liu, Ming Zhou, and Ke Xu. The automated acquisition of suggestions from tweets. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*. AAAI Press, 2013.

[19] Vernon Howard Dudman. Indicative and subjunctive. *Analysis*, pages 113–122, 1988.

[20] Andrea Esuli and Fabrizio Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of the 5th Conference on Language Resources and Evaluation (LREC'06*, pages 417–422, 2006.

[21] Christiane Fellbaum. *WordNet: an electronic lexical database*. MIT Press, 1998.

[22] Lorenzo Ferrone and Fabio Massimo Zanzotto. Symbolic, distributed and distributional representations for natural language processing in the era of deep learning: a survey. *arXiv preprint arXiv:1702.00764*, 2017.

[23] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 363–370. Association for Computational Linguistics, 2005.

[24] Xiaowei Guan. A study on the formalization of english subjunctive mood. Academy Publisher, 2012.

[25] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–1780, 1997.

[26] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 168–177. ACM, 2004.

[27] Claudia Iacob and Rachel Harrison. Retrieving and analyzing mobile apps feature requests from online reviews. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, MSR '13, pages 41–44. IEEE Press, 2013.

[28] Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. *Machine learning: ECML-98*, pages 137–142, 1998.

[29] Dan Jurafsky and James H Martin. *Speech and language processing*, volume 3. Pearson London, 2014.

[30] Tom Kenter, Alexey Borisov, and Maarten de Rijke. Siamese cbow: Optimizing word embeddings for sentence representations. *arXiv preprint arXiv:1606.04640*, 2016.

[31] Anthony Khoo, Yuval Marom, and David Albrecht. Experiments with sentence classification. In *Proceedings of the 2006 Australasian language technology workshop*, pages 18–25, 2006.

[32] Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 423–430, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[33] Alexandros Komninos and Suresh Manandhar. Dependency based embeddings for sentence classification tasks. In *Proceedings of the 2016 Conference of the North*

*American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1490–1500. Association for Computational Linguistics, 2016.

[34] Omer Levy and Yoav Goldberg. Dependency-based word embeddings. In *ACL*, 2014.

[35] Bing Liu. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167, 2012.

[36] Edward Loper and Steven Bird. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, pages 63–70. Association for Computational Linguistics, 2002.

[37] Bill Maccartney. *Natural Language Inference*. PhD thesis, 2009.

[38] Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[39] Alicia Martínez Flor. A theoretical review of the speech act of suggesting: Towards a taxonomy for its use in flt. *Revista alicantina de estudios ingleses, No. 18 (Nov. 2005); pp. 167-187*, 2005.

[40] Oren Melamud, David McClosky, Siddharth Patwardhan, and Mohit Bansal. The role of context types and dimensionality in learning word embeddings. In *Proceedings of NAACL-HLT*, pages 1030–1040, 2016.

[41] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[42] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of NAACL-HLT*, pages 746–751, 2013.

[43] Samaneh Moghaddam. Beyond sentiment analysis: mining defects and improvements from customer feedback. In *European Conference on Information Retrieval*, pages 400–410. Springer, 2015.

[44] Saif M Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. Semeval-2016 task 6: Detecting stance in tweets. pages 31–41, 2016.

[45] Roser Morante and Caroline Sporleder. Modality and negation: An introduction to the special issue. *Computational Linguistics*, 38(2):223–260, 2012.

[46] Ramanathan Narayanan, Bing Liu, and Alok Choudhary. Sentiment analysis of conditional sentences. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, EMNLP '09, pages 180–189. Association for Computational Linguistics, 2009.

[47] Sapna Negi. Suggestion mining from opinionated text. In Alberto Pozzi, Elisabetta Fersini, Enza Messina, and Bing Liu, editors, *Sentiment Analysis in Social Networks*, chapter 8. Elsevier, 2016.

[48] Sapna Negi and Paul Buitelaar. Towards the extraction of customer-to-customer suggestions from reviews. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2159–2167, Lisbon, Portugal, 2015. Association for Computational Linguistics.

[49] Sapna Negi and Paul Buitelaar. Curse or boon? presence of subjunctive mood in opinionated text. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 101–106, London, UK, 2015. Association for Computational Linguistics.

[50] Sapna Negi and Paul Buitelaar. Inducing distant supervision in suggestion mining through part-of-speech embeddings. *arXiv preprint arXiv:1709.07403*, 2017.

[51] Sapna Negi, Kartik Asooja, Shubham Mehrotra, and Paul Buitelaar. A study of suggestions in opinionated texts and their automatic detection. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 170–178. Association for Computational Linguistics, 2016.

[52] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

[53] Paolo Pareti, Benoit Testu, Ryutaro Ichise, Ewan Klein, and Adam Barker. Integrating know-how into the linked data cloud. In Krzysztof Janowicz, Stefan Schlobach, Patrick Lambrix, and Eero Hyvönen, editors, *Knowledge Engineering and Knowledge Management*, volume 8876 of *Lecture Notes in Computer Science*, pages 385–396. Springer International Publishing, 2014.

[54] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.

[55] Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, Jianyong Wang, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Mei-Chun Hsu. Mining sequential patterns by

pattern-growth: The prefixspan approach. *IEEE Transactions on knowledge and data engineering*, 16(11):1424–1440, 2004.

[56] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[57] Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammed AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Veronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Núria Bel, Salud Maria Jiménez-Zafra, and Gülşen Eryiğit. Semeval-2016 task 5 : aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, pages 19–30. Association for Computational Linguistics, 2016.

[58] Paul Portner. *Modality*. Oxford University Press, 2009.

[59] Federico Alberto Pozzi, Elisabetta Fersini, Enza Messina, and Bing Liu. *Sentiment analysis in social networks*. Morgan Kaufmann, 2016.

[60] J Ramanand, Krishna Bhavsar, and Niranjan Pedanekar. Wishful thinking - finding suggestions and 'buy' wishes from product reviews. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 54–61. Association for Computational Linguistics, 2010.

[61] Steffen Rendle. Factorization machines. In *Proceedings of the 2010 IEEE International Conference on Data Mining*, ICDM '10, pages 995–1000, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-0-7695-4256-0.

[62] Ellen Riloff and Janyce Wiebe. Learning extraction patterns for subjective expressions. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 105–112. Association for Computational Linguistics, 2003.

[63] Sara Rosenthal, Preslav Nakov, Svetlana Kiritchenko, Saif Mohammad, Alan Ritter, and Veselin Stoyanov. Semeval-2015 task 10: Sentiment analysis in twitter. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, pages 451–463, Denver, Colorado, 2015. Association for Computational Linguistics.

[64] John Searle. What is a speech act? In Max Black, editor, *Philosophy in America*, pages 221–239. Ithaca: Cornell University Press, 1965.

[65] John R. Searle. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press, 1969.

[66] John R. Searle. A classification of illocutionary acts. *Language in Society*, 5(1): 1–23, 1976.

[67] Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1555–1565. Association for Computational Linguistics, 2014.

[68] Chen Tao, Xu Ruifeng, He Yulan, and Wang Xuan. Improving sentiment analysis via sentence type classification using bilstm-crf and cnn. *Expert Systems with Applications*, 72:221 – 230, 2017.

[69] Ann Taylor, Mitchell Marcus, and Beatrice Santorini. The penn treebank: an overview. In *Treebanks*, pages 5–22. Springer, 2003.

[70] Amar Viswanathan, Prasanna Venkatesh, Bintu G Vasudevan, Rajesh Balakrishnan, and Lokendra Shastri. Suggestion mining from customer reviews. In *AMCIS*, 2011.

[71] Henning Wachsmuth, Martin Trenkmann, Benno Stein, Gregor Engels, and Tsvetomira Palakarska. A review corpus for argumentation analysis. In *Proceedings of the 15th International Conference on Computational Linguistics and Intelligent Text Processing*, volume 8404 of *LNCS*, pages 115–127, Kathmandu, Nepal, 2014. Springer.

[72] Alfan Farizki Wicaksono and Sung-Hyon Myaeng. Mining advices from weblogs. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 2347–2350. ACM, 2012.

[73] Alfan Farizki Wicaksono and Sung-Hyon Myaeng. Automatic extraction of advice-revealing sentences for advice mining from online forums. In *Proceedings of the Seventh International Conference on Knowledge Capture*, K-CAP '13, pages 97–104. ACM, 2013.

[74] Alfan Farizki Wicaksono and Sung-Hyon Myaeng. Toward advice mining: Conditional random fields for extracting advice-revealing text units. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*, CIKM '13, pages 2039–2048. ACM, 2013.

[75] Janyce Wiebe and Ellen Riloff. Creating subjective and objective sentence classifiers from unannotated texts. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 486–497. Springer, 2005.

[76] Xifeng Yan, Jiawei Han, and Ramin Afshar. Clospan: Mining closed sequential patterns in large datasets. In *Proceedings of the 2003 SIAM international conference on data mining*, pages 166–177, 2003.

[77] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 412–420. Morgan Kaufmann Publishers Inc., 1997.