

INDIAN INSTITUTE OF TECHNOLOGY, HYDERABAD

# Faceted Navigation for Reviews

by

Neelesh Dewangan

A thesis submitted in partial fulfillment for the  
degree of Master of Technology

in

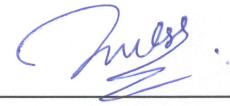
Computer Science Engineering

under guidance of  
Dr. Manish Singh

June 2016

## *Declaration*

I, Neelesh Dewangan, declare that this thesis titled, “**Faceted Navigation for Reviews**” and the work presented in it are written by my own. I confirm that this work was done wholly or mainly while in candidature for a thesis research. Any part of this thesis has not previously been submitted for any degree or any other qualification at this University or any other institution, this has been clearly stated. I have consulted the published work of others, this is always clearly attributed. 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.



---

Signature

NEELES H DEWANGAN

Student Name

CS14MTECH11009

Roll Number

# Approval

This Thesis entitled “**Faceted Navigation for Reviews**” by Neelesh Dewangan is approved for the degree of Master of Technology from IIT Hyderabad

*H. B. Dinu*

---

Examiner  
Dept. of CSE  
IITH

*C. Krishna Mohan*

---

Examiner  
Dept. of CSE  
IITH

*AD Singh*

---

(Dr. Manish Singh) Adviser  
Dept. of CSE  
IITH

*U. Raghava*  
*U. RAMAKRISHNA*

---

Chairman  
Dept. of CSE  
IITH

# *Abstract*

User often go through the product reviews to get an insight into product quality and its various features. As reviews are unstructured and voluminous, user faces many difficulties to find the relevant information when he reads the several reviews. And due to certain feature preferences, it becomes more cumbersome for a user to read the whole review which could be hundreds in lines. In this dissertation, we propose an interactive system to resolve these problems where user can extract the relevant reviews by few interaction with the user interface. Also in our work, we improved the user experience by providing more meaningful and relevant results by associating relationship between different features to our model. At last, we constructed the feature ontology tree to overcome existing information overload problem and provided a faceted navigation to explore the reviews in more efficient manner.

## *Acknowledgements*

I express my deepest gratitude to all the people who have supported me and encouraged me during the course of this project. I am very grateful to my thesis advisor Dr. Manish Singh for giving me the opportunity of carrying out this research under their guidance. I am very deeply indebted to him for his support, advice and encouragement without which this research could not have proceeded smoothly. He had encouraged me to think in a scientific and methodical manner.

I would also like to thank my colleague konjengbam anand for his continuous support on helping in understanding the domain and discussing the relevant topic on text processing and retrieval. Finally, I thank to my family members and my friends for their continuous motivation and encouragement.

# Contents

<b>Declaration</b>	<b>i</b>
<b>Approval</b>	<b>ii</b>
<b>Abstract</b>	<b>iii</b>
<b>Acknowledgements</b>	<b>iv</b>
<b>List of Figures</b>	<b>vii</b>
<b>List of Tables</b>	<b>viii</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 Problem Definition</b>	<b>4</b>
2.1 Overview . . . . .	4
2.2 Aspect Based Review Extraction . . . . .	5
2.3 Semantic Aspect Based Review Extraction . . . . .	5
2.4 Feature Ontology . . . . .	6
2.5 Faceted Navigation over reviews . . . . .	7
<b>3 Related Work</b>	<b>8</b>
3.1 Faceted Navigation . . . . .	8
3.2 Opinion Search . . . . .	8
<b>4 Proposed System Architecture</b>	<b>10</b>
4.1 Candidate Feature Set Generation . . . . .	10
4.2 Feature Ontology Tree Construction . . . . .	11
4.3 Mining Feature review mapping . . . . .	14
<b>5 Experimental Setup and Results</b>	<b>15</b>
5.1 Dataset . . . . .	15
5.2 Implementation and System Setup . . . . .	15
5.3 Results . . . . .	16
5.3.1 Percentage of relevance to Original features in ConceptNet . . . . .	16
5.3.2 Feature Extraction Evaluation . . . . .	17
5.3.3 Topic Drift vs Relation Class measure . . . . .	19
<b>6 Snapshot</b>	<b>20</b>

---

6.1	User Interface . . . . .	20
6.2	Faceted Navigation Panel . . . . .	21
6.3	Review Panel . . . . .	22
<b>7</b>	<b>Conclusion</b>	<b>23</b>
	<b>Bibliography</b>	<b>24</b>

# List of Figures

4.1	Faceted Navigation System Architecture for Reviews . . . . .	11
5.1	Precision Recall Curve - Camera Product . . . . .	17
5.2	Precision Recall Curve - Phone Product . . . . .	18
5.3	Precision Recall Curve - Jukebox Product . . . . .	18
5.4	Precision Recall Curve - DVD Player Product . . . . .	18
6.1	Complete User Interface . . . . .	20
6.2	Faceted Navigation Panel for Camera Product . . . . .	21
6.3	Extracted semantically meaningful reviews for “Lens” feature . . . . .	22



# List of Tables

4.1	Relation Class and Priority Order(“>” and “=” shows inner class priority order) . . . . .	12
4.2	Edge types and Relations List . . . . .	12
5.1	Existence of product review concepts in ConceptNet . . . . .	17
5.2	Precision Recall Score for each domain with respect to Tf-Idf threshold . . . . .	17
5.3	Irrelevant Features with respect to Relation classes(with noun phrase extension) . . . . .	19
5.4	Irrelevant Features with respect to Relation classes(without noun phrase extension) . . . . .	19

# Chapter 1

## Introduction

In today's highly competitive e-market, making online purchase is not an easy task. With the availability of wide range of products and services, choosing a product is very difficult for any user. To ease up the task, existing e-commerce websites (e.g. flipkart, amazon, snapdeal etc.) provide various ways of exploring and narrowing down the products to help users in making better decision. Usually these methods include **category exploration**, **product search** and **feature(or facet) selection**. These methods lead to small set of results where user can choose the right product. But even if the user has find the right product for him, he has to make sure that quality and features are good enough as per their specification. This is where they rely on the reviews of the product.

Usually, reviews play an important role in decision making of product purchase. Reviews generally describe the quality of products with respect to certain key terms. These key terms are referred as **feature** or **aspect** of the product. For example, in a camera product review, these features could be lens, zoom or picture quality. Analysing these features and reviews help sellers to extract and identify the potential customers behaviour as well as opinion on any product.

While making online shopping, for a single product, the feature preference varies from user to user. Users generally focus on some subset of features when they make the decision and to verify the quality of those features, they go through several reviews to find out the relevant information about those features. Finding this relevant information out of the pool of unstructured reviews is very difficult due to following limitations in existing system:

- Existence of large number of unstructured text reviews for a single product due to which going through all the reviews is difficult.

- Filtering reviews based on set features is not developed in any existing state-of-art-systems.

To overcome limitation 1, many researcher proposes various text summarization techniques [1–3] to summarize the opinions of different features into a rating score measure. These ratings give information about quality of the feature in terms of positive or negative score. The rating system have its own pitfalls i.e. it fails to answer the question: “**what is good or bad about this specific feature**”. It just gives information about “**how much good or bad is this feature**”.

To address this issue, an efficient system is needed that can extract and map the features to their respective relevant review sentences(**review snippets**). In our work, we refer this task as **Aspect Based Review Extraction**. Usually this task will extract the relevant review sentences in which interested feature is present. For example, here is the camera review:

“Battery life of camera is good. Lens and zoom the camera is not that great. Also picture quality is not that good enough.”

If the interested feature is “lens” then aspect based feature extraction will give you result as “ Lens and zoom of the camera is not that great.”. This static system have one limitation also: It fails to capture **semantic relationship** between features. Semantic relationship determines how two different objects are related to each other. It provides meaning to overall information. Semantic relationship between features can be important to the user as it enhance the understanding about the product while giving the rich information about features and its related sub-features. In above example, lens, zoom and picture quality are semantically related. In our work, we have included the semantic relationship between features to improve the overall model which is our novel work. E.g. for above example, interested feature “lens” will provide the result as:

“Lens and zoom the camera is not that great. Also picture quality is not that good enough.”

For interested feature “lens”, above result represents a knowledge that “as lens is not great picture quality is not good too”. We referred this review extraction process as **Semantic Aspect Based Review Extraction**.

Our other novel work is that we have built a faceted navigation system for reviews by leveraging the feature semantic relationship. Faceted navigation systems[4] provides lot of benefits. One of the benefits that it provide is that it deals with information overflow problem using exploratory search. Information overflow problem can exist in our model as the number of features can be exist in hundreds. Hence all the feature

can't be shown to user at once. To show few features at any instance to the user, we folded down the features by leveraging the feature semantic relationship into a feature ontology tree. This tree will be exposed to user as a faceted navigation panel. User can extract the semantically relevant reviews by selecting the interested feature. Also user can narrow down the extracted result by exploring the relevant interested path in feature hierarchy.

## Chapter 2

# Problem Definition

In this chapter, first we will give some overview to product reviews, features and existing state-of-art-system. Later we will define each problem definition in different subsection.

### 2.1 Overview

In existing e-commerce websites(e.g. flipkart, amazon), product reviews are maintained to get products feedback. These reviews are usually submitted by the users in unstructured text format. These reviews are generally centred towards various **product features** and discuss about quality of those aspects. Product features are frequent over review data as most of the reviews mentions these key terms frequently. These features are usually represented by **noun terms** in unstructured text. For example, for a mobile product, features could be processor, RAM, touchscreen etc.

When user go through the reviews, they usually search for these product features. As the reviews, for a product, are voluminous and abundant, reading all the reviews is cumbersome task. Existing e-commerce websites provide overall rating to product. Overall rating measures quality of complete product rather than the quality of individual features. Hence if the user wants to check quality of individual feature user need to go through several reviews. While reviewing the several reviews, user often faces these two problems:

- Existing e-commerce websites do ranking of the reviews. Limitation of top ranking is that it sometimes do not provide information about non popular features.

- Sometime reviews are described in way too much detail. Going through the complete review is not relevant to user.

Due to both problems user has to read several reviews to have adequate knowledge about certain feature. In our work, we will provide a way to find all the relevant information about certain features. Also in our model, all the semantic relation knowledge are included to improve the meaning of overall result.

## 2.2 Aspect Based Review Extraction

Assume review set  $R$  have thousands of reviews for certain product. Each review  $R_i$  consists set of review snippets  $S = \{r_1, r_2, r_3, \dots\}$  having various features  $f$  from feature set  $F = \{f_1, f_2, f_3, \dots\}$ . Our goal is to create  $\langle \text{feature}, \text{review snippet} \rangle$  mappings for each feature i.e.  $\langle f_x, r_y \rangle$  which will help in order to extract the relevant review snippet  $r_y$  by selecting interested feature  $f_x$  by user.

This process can be easily demonstrated as follow: For a mobile product here are the two reviews:

“Touch screen of the mobile is great, but the battery life is very short.”

“This phone’s battery is not good. I have to charge the phone twice a day.”

Our system will generate  $\langle \text{feature}, \text{review snippet} \rangle$  mappings as:

$\langle \text{touch screen}, [ \text{“Touch screen of the mobile is great”} ] \rangle$

$\langle \text{battery}, [ \text{“but the battery life is very short”}, \text{“This phone’s battery is not good.”} ] \rangle$

By choosing this feature the relevant feature “touch screen” extracted relevant review will be:

“Touch screen of the mobile is great”

## 2.3 Semantic Aspect Based Review Extraction

Associating semantic relationship with aspect based review extraction to enhance semantic meaning of the extracted review snippets is our primary goal. Assume  $\langle \text{feature}, \text{review snippet} \rangle$  mapping as  $\langle f_i, \bar{S}_i \rangle$  extracted from previous result. If the interested feature is  $f_i$  and it is semantically related to features in feature set  $F_s = \{f_j, f_{j+1}, f_{j+2}, \dots, m \text{ elements} \}$  whose respective relevant snippets are

$S = \{S_j, S_{j+1}, S_{j+2}, \dots, m \text{ elements}\}$  then semantic aspect based review will generate  $\langle \text{feature}, \text{review snippet} \rangle$  mapping for interested feature  $f_i$  as:

$$\langle f_i, \bar{S}_x \rangle = \langle f_i, \bigcup_{k=j}^{j+m-1} S_k \cup \bar{S}_i \rangle \text{ where } S_k \in \langle f_k, S_k \rangle \text{ and } S_k \in S \text{ and } f_k \in F_s$$

Example of semantic relations can be viewed as “camera flash is related to picture”. Here in the example camera flash and picture are related to each other by relation “Related to”. In previous problem, the result miss out semantic relation and provide only static reviews. This might lead to **insufficient review extraction problem**. For example, here is a review of mobile:

“Shutter speed of camera is not good. Hence image taken for moving object is blurred.”

For above example, for interested feature “image” problem 1 will result as “Hence image taken for moving object is blurred.”. This result lacks in one information that why image are not good. By associating these semantic relations with problem 1 can easily resolve the insufficient review extraction problem. As “Shutter speed” and “image” are semantically related, our model will result the whole review for above example.

As automatic extraction of these relationships are difficult, we need an existing dataset for our model to extract these relationship. For our model, we have used conceptNet[5] for semantic relationship extraction which we will discuss in later section.

## 2.4 Feature Ontology

Information overload is a common issue that user face with most of the information retrieval system. In our system, user is presented with set of features  $F = \{f_1, f_2, f_3, \dots\}$  extracted from previous step. Feature extraction could result in hundreds of features which might lead to information overload when features set  $F$  is exposed to user.

To efficiently control features exposure at any instance, a *feature ontology tree* is constructed. Feature ontology tree  $T = (V, E)$  is a feature relationship hierarchy where each node  $v_i$  represents a *feature* of product and directed edge  $e_i = (v_i, v_j)$  represents *parent – child relationship* or *feature – subfeature relationship*. Root of the tree is the “*domain*” of the product.

Feature ontology tree leverages feature-feature relationship to fold down multiple features into single feature as a child. Example of this feature-feature relationship can be viewed as:

“focus is part of lens”

From above example, we can deduce that focus and lens are related as child and parent. Many user are unaware of these relationship between features and organising in hierarchy improves the product understanding. Other advantage of organising features in hierarchy is that it can be used for controlling the feature overflow information while exposing features to user. Construction of this feature ontology tree was published in research work[6]. Our algorithm is variation of their work which we will discuss in algorithm section.

## 2.5 Faceted Navigation over reviews

Faceted navigation are generally used to control information overflow while exploring abundant data. Faceted navigation[4] uses cataloguing technique to improve data categorization using facets and provide exploration search over these data. Our work can easily be mapped to faceted navigation solution as review snippets are categorized using features. In our work, feature ontology tree is used as a faceted navigation panel where each facet represents a feature of the product. User can easily browse through reviews for find the relevant information by choosing the features from tree iteratively. Extracted reviews snippets are either relevant to the feature selected or relevant to their child. For example, for a camera product, choosing lens will give reviews about lens and its child focus, picture etc. Initially domain of product and all the reviews are exposed to user. User can narrow down the result using iterative selection of feature over tree until he is satisfied with presented result.



## Chapter 3

# Related Work

Our work is relevant to two broad categories: *faceted navigation*, *opinion search*. We will discuss each of these categories in separate section.

### 3.1 Faceted Navigation

Faceted navigation has been a popular research topic in past decade. Due to various advantages of faceted navigation, most of the researcher came up with faceted navigation on various applications. Authors in [7] proposed faceted based interface for mobile interface called *FaThumb* to browse large amount of information in mobile. *FaThumb* uses an hybrid model using both keyword search and hierarchical facet metadata navigation to prune out irrelevant data resulting satisfactory results. Faceted navigation for wikipedia searches were proposed in work[8]. In their work, they arranged various attributes templates called infobox templates into a ontology and provided a search interface over RDF triplet knowledge base to extract the relevant information. In these two faceted navigation, faceted hierarchy is manually built. Building manual faceted ontology for reviews is a challenging task as it comes in abundance and unstructured format. In our work, we will show how these hierarchy is created automatically.

### 3.2 Opinion Search

Other relevant research falls under opinion search[9]. In opinion search, based on user's query, relevant opinionated documents are extracted from document corpus. Relevant documents are usually recognized using certain set of keywords which represents the features of the documents. In paper[10], authors used noun and noun phrase extraction

technique[11] to extract these features from user’s query and find relevant documents involving these feature and their synonyms using similarity search. Opinionated sentences from documents are later extracted using SVM supervised learning.

In similar area, in *opinion mining*, various feature extraction techniques are discussed. In Hu’s work[1] using *association rule mining* technique, frequent features are extracted from the product review. Researchers in work[12] came up with Opine system which extracts product features by associating *point-wise mutual information scores* to frequent noun phrases using unsupervised learning. Also other researches[2, 13] are related to sentence-level feature extraction. These approaches depends on text patterns to extract the product features.

Our work leverages conceptNet[5] to build feature ontology tree. ConceptNet database can be viewed as semantic network graph database which represent the real world knowledge and common sense relations between various objects. Each node in conceptNet is a concept which represents a real world object and edges represents semantic relationship between various concepts. ConceptNet can be used in various artificial intelligent and text analytics applications to extract the semantic relationship and domain sensitive information. Example of conceptNet relationship is:

“lens is part of camera”

“camera is capable of taking picture.”

In first example, “part of” is the relationship between concepts lens and camera. Similarly for second example, “capable of” is the relationship between camera and picture. Our work is adaptation of Mukhargee’s work[6]. In their work, they proposed feature ontology tree construction algorithm to build feature hierarchy. They have used this tree to efficiently feature and sentiments to user. Our goal is different from them i.e. to solve information overload problem while presenting features to user and enhancing understanding of product attributes and their relationships. Our work is different in two aspects. First is we have proposed **edge type** concept to built a **directed feature ontology tree**. In [6] information of *parent-child* relation is missing due to undirected tree. Other difference is that we chose **BFS exploration** for our tree expansion model and provided inner class priority order to enhance feature ontology tree construction. BFS exploration provide shortest path to relevant concepts which is beneficial for efficient and consistent tree construction. These aspects will be further explained in next chapter.

## Chapter 4

# Proposed System Architecture

Figure 4.1 shows our faceted navigation system architecture for reviews. Our proposed solution system consist three modules: Candidate feature generation, feature ontology tree construction, mining feature review mappings. Input to the system are product reviews collected from various datasets. Each of these reviews are exists in unstructured text format. Hence, raw review inputs are first cleaned and tagged into linguistic parts(nouns, verb etc) using **brill part-of-speech tagger**[14]. Later in our first module, nouns are extracted and pruned out from tagged data to generate candidate features. Candidate features determines potential features of the product extracted from review corpus. These features are usually represented by noun terms in unstructured text and can be easily extracted from tagged data. In our third module, we hierarchize the candidate feature as a feature ontology tree. Nodes of these ontology tree represents features of the product. At last, we generate feature and review snippet mappings and store it to our database. Feature ontology is presented to user as a faceted navigation panel and user interactively narrows down the reviews and relevant snippet by exploring the facets. Each modules are explained in details as separate subsections.

### 4.1 Candidate Feature Set Generation

In order to construct feature ontology tree, candidate features are extracted from the review corpus. In our work to achieve these we used two different approaches: **association rule mining**[1] and **tf-idf pruning**. These approaches helps in pruning out the irrelevant nouns and generates a set of candidate features. Association rule mining extract the frequent nouns patterns occurred in review corpus. These frequent nouns are again pruned out using compact pruning technique to generate features. This

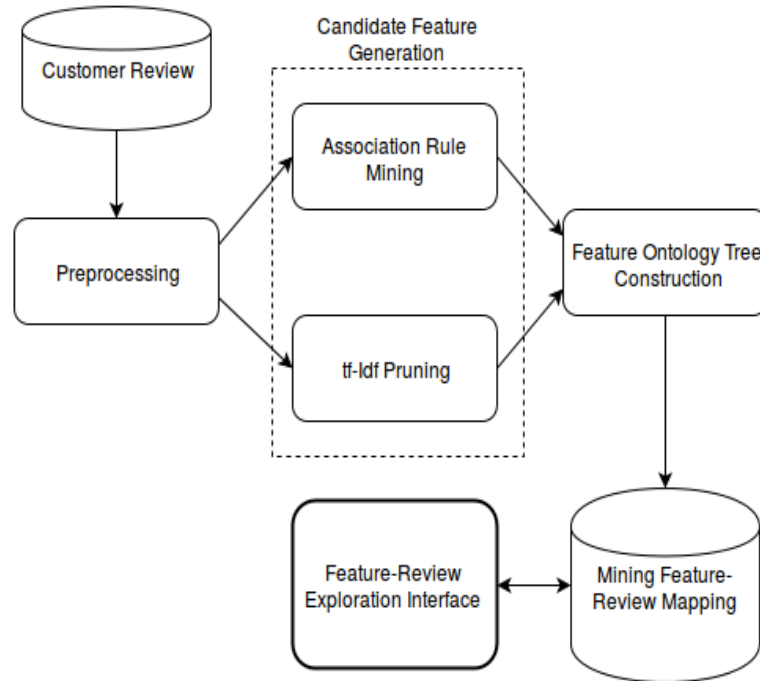


FIGURE 4.1: Faceted Navigation System Architecture for Reviews

technique is only able to extract frequent features and left out the infrequent features which occurs very less in review corpus.

*Tf-Idf* statistically measures term importance over the set of documents. This term importance are measure by two units: *term frequency* and *inverse document frequency*. Term frequency determines how frequent a term is occurring in a document whereas IDF defines how rare a term is to set of documents. Candidate feature or noun can be easily extracted from review corpus by applying certain *tf-idf threshold*. Advantage of applying *tf-idf pruning* is it includes the infrequent feature in the candidate feature set. By these two approaches, we will create two sets of candidate features and apply other modules separately on both sets. Later we will evaluate the final results in evaluation section.

## 4.2 Feature Ontology Tree Construction

Feature ontology gives information about how features are correlated with each other. We can resolve feature information overload problem using this feature ontology tree by presenting top level features at a time. User can explore the other subfeatures by their relevant top level features. Root of the ontology tree is the domain name of the product. The domain can be easily extracted from the candidate features by extracting the most frequent feature from the candidate features. Nodes in the tree are

the features of the product and an edge between features  $f_1 \rightarrow f_2$  determines *parent to child* relationship.

To build feature ontology tree we leveraged **ConceptNet** database to extract semantic relationship between feature to other feature. ConceptNet provides various many semantic relationship. Out of which, we selected 12 relations to built our ontology tree. Also ConceptNet has one to many relationship between various domains. Hence there is big chance of **topic drift** while expanding ontology tree. For example, “*Camera* is relatedTo *lens*”, “*lens* is madeOf *glass*”, “*glass* is relatedTo *window*”. Here *camera* and *window* are related as per conceptNet but *window* doesn’t belong to camera domain. Hence to control *topic drift*, authors in [6] proposed categorization of the relations into three classes(table 4.1): **Heirarchical relations**, **Synonym relations**, **Functional relations** and also proposed evaluation order  $H > S > F$ .

Relation Class	Relations	Priority
Heirarchical Relations	HasA, PartOf, MadeOf, LocatedNear	1
Synonym Relations	Synonym = DefinedAs > DerivedFrom > IsA > RelatedTo	2
Functional Relations	UsedFor, CapableOf, HasProperty	3

TABLE 4.1: Relation Class and Priority Order(“>” and “=” shows inner class priority order)

Algorithm proposed in work[6] missed out one information: how will we decide that a  $feature_{new}$  will be a *parent* or a *child* of the  $feature_{old}$ . For example, in accordance to algorithm mentioned in [6] , “*lens* is partOf *camera*” will result in undirected edge (*lens,camera*) in ontology tree. Hence due to undirected edge it is unclear that which one of them is a *parent* or a *child*. Correct *parent-child* relationship is the necessity of our model. To resolve this, we proposed the concept of **edge type** in our model.

Edge Type	Relations
Top to Bottom Relation(TBR)	HasA, LocatedNear, MadeOf UsedFor, CapableOf, HasProperty
Bottom Up Relation(BUR)	PartOf, DerivedFrom
Bidirectional relations(BDR)	Synonym, DefinedAs, RelatedTo, IsA

TABLE 4.2: Edge types and Relations List

For our purpose, *edge type* will provide the information of the new feature  $f$  as being a **parent** or a **child**. *Edge type* is decided based on semantic relationship. We classified relations into three edge types: **Top to bottom relation**(TBR), **Bottom**

to **up relation**(BUR) and **Bidirectional relations**(BDR). Relations belongs to these edge types is described in table 4.2.

---

**Algorithm 1:** Construction of feature ontology tree

---

**Input** : Candidate Feature Set  $D = \{ \langle N, f \rangle \}$  where  
 $N$ =candidate feature set,  
 $f$ =frequency count of each candidate feature  
Relation List  $H, S, F$

**Output:** Feature Ontology Tree  $G$

```

1 Graph  $G = (V, E)$  where  $V$ =Vertex Set and  $E$ =Edge Set
2 Initialize  $V=\phi$  and  $E=\phi$ , Initialize Queue= $Q$ 
3 domain  $d_1$  =max frequency candidate feature
4  $Q.enqueue(d_1)$ 
5 visited[ $d_1$ ]=true
6 for each relation set  $R$  in  $\{H, S, F\}$  do
7   while  $Q$  is not empty do
8     Concept  $c = Q.dequeue()$ 
9      $V_1 =$  Extract all nodes connected to concept  $c$  in concept-net iff  $r_{new} \in R$ 
10    if  $v_i \in V_1$  and  $v_i \in N$  and  $v_i \notin$  visited then
11      if  $v_i \notin G$  then
12        Add vertex  $v_i$  to  $V$ 
13        if  $r_{new} \in TBR$  or  $r_{new} \in BDR$  then
14          Add edge  $(c, v_i)$  to  $E$ 
15        else
16          Add edge  $(v_i, c)$  to  $E$ 
17      else
18        Get parent  $p$  and old relation  $r_{old}$  between parent  $p$  and concept  $c$ 
19        if  $priority_{r_{new}} > priority_{r_{old}}$  then
20          Update parent of  $v_i$  to new parent  $c$ 
21       $Q.enqueue(v_i)$ 
22      visited[ $v_i$ ]=true
23 Merge the nodes in  $G$  iff relation  $\in \{Synonym, DefinedAs\}$ 

```

---

$TBR$  relations shows *parent* to *child* relationship whereas  $BUR$  exhibits the property of *child* to *parent* relationship.  $BDR$  relations are special case where it exhibits both the property of  $TBR$  relations and  $BUR$  relations. Depending on these edge types *parent* and *child* are decided. For example:

“lens is partOf Camera”

“Camera is capableOf taking pictures”

“focus is related to lens”

In the first example, lens and camera has bottom up relationship hence edge ( lens  $\leftarrow$  camera ) is added to ontology tree. Similarly for next two example edge ( camera  $\rightarrow$

picture ) and ( focus  $\leftrightarrow$  lens ) is added to ontology tree as they have top to bottom and bidirectional relationship respectively. For bidirectional edges, we have two possibility to add edge (e.g.  $f_1 \leftrightarrow f_2$ :  $f_1 \leftarrow f_2$  or  $f_1 \rightarrow f_2$ ). Any of them can become subfeature of other feature. Hence to maintain tree coherency, i.e. edges only direct from top to bottom, we will make feature  $f_1$  as subfeature iff feature  $f_2$  is present in tree  $G$  and vice versa.

In addition to edge types, we have used BFS exploration for our ontology expansion. We extract relevant concepts from conceptnet by providing the *product domain* initially and connect the features edges based on edge type in BFS manner. We repeat the process of each *subfeature* until tree expanded up to predefined **threshold height**  $h$ . Benefits of BFS exploration is that it provides shortest distance to relevant concepts from product domain in minimum depth expansion. In case of many to many relation edge conflict  $H > S > F$  order evaluation is followed. Also, to cluster the all the relevant synonyms at single respective feature, we defined inner priority order within synonym relation class as mentioned in 4.1. In conflicting case, edge in ontology tree is added for higher priority relation and deleting the lower priority relation edge. Finally we will merge all the synonyms to remove the redundant information from the tree. Our complete algorithm is described in algorithm 1.

### 4.3 Mining Feature review mapping

After creating feature ontology tree, we mine all the  $\langle \text{feature}, \text{review snippet} \rangle$  mapping as a **transaction file**. Each transaction file represent a mapping between a single feature to corresponding reviews and it is created for each features individually. In our model, we stored these files into the nosql database ( e.g. couchdb, mongodb ). These transaction files are later queried for exploring the reviews iteratively based on consumer interaction on the interface. Structure of each transaction file and reviews are mentioned as follows:

**Transaction File:**

```
{ id: <Feature ID>, Feature: <Feature Name> , review: <List of reviews > }
```

**Review:**

```
{ revId: <Review Id>, lineId: <LineNumber> , Value: <Corresponding Review Line>
}
```

## Chapter 5

# Experimental Setup and Results

In this chapter, we will discuss about various evaluation parameters and result generated from our model. We divided this chapter into three section: Dataset, System Setup and result. Each of these sections are explained as follows.

### 5.1 Dataset

In our work, we used 4 product review published in Bing and Liu's work[1]. These product reviews 4 different domains ie camera, phone, music jukebox and dvd player. Features of each review lines are already annotated in the product review dataset. We have used these features as ground truth for our precision recall evaluation. Number of reviews and features in each product domain are mentioned in following table:

To extract semantic relationship between features and to build feature ontology, we have used ConceptNet semantic network database[5]. ConceptNet have lot of domains and various concepts are interconnected to each other using various relations.

### 5.2 Implementation and System Setup

To setup each system module, we have divided the system into three main module for implementation purpose: Backend system, Databases and user interface. Backend system is our main model which deals with generation of feature ontology tree and  $\langle feature, review snippet \rangle$  mappings. Input to the this module is unstructured review text as per Bing's dataset format. This module includes preprocessing of data, candidate feature generation, feature ontology tree generation and  $\langle feature, review snippet \rangle$



mapping generation. To implement our backend system, we have used python programming language. After each of these processes, two results are generated: feature ontology tree as Json file,  $\langle feature, review snippet \rangle$  as multiple documents in our NoSQL database.

For our work, we have used CouchDb database as NoSQL database to store  $\langle feature, review snippet \rangle$  mapping. CouchDb is a **document based NoSQL database**. As name suggests, benefit of using couchDb is it stores the data as a document and provide distributed architecture to retrieve the data. Each document in couchDb represent single  $\langle feature, review snippet \rangle$  mapping. We have used javascript to retrieve data from couchDb.

To develop our user interface, we have used javascripts and bootstrap library to build an interactive UI. This UI have two panels: faceted navigation panel, review panel. Using javascript, extracted json file is converted into collapsible feature tree. Faceted navigation panel expose this feature tree to user and expanded based on user interaction. By last user interaction, relevant review snippets are extracted and shown in review panel.

## 5.3 Results

For our work, we have used three kinds evaluation to evaluate our model. Each of these evaluations are discussed as follows:

### 5.3.1 Percentage of relevance to Original features in ConceptNet

Our feature extraction process is highly dependent on ConceptNet. In our model, candidate feature is considered irrelevant if it is not present in ConceptNet. ConceptNet has lot of concepts but there is a chance that any tagged feature might not be present in it. We have checked that if an original feature exist in conceptNet as a concept and calculated percentage of original features mapped to any concept in ConceptNet. For our dataset, we have noticed that most of missing features are noun phrases that doesn't exist in conceptNet. The relevance percentage of original feature that exist in conceptNet of each domain is mentioned in table 5.1.

### 5.3.2 Feature Extraction Evaluation

For our model, We have used precision and recall score to evaluate feature extraction process. We have taken annotated features from dataset as ground truth to calculate

Domain	Nouns Existence	Feature Existence
Camera	0.43	0.67
Phone	0.49	0.73
Jukebox	0.40	0.75
DVD Player	0.46	0.74

TABLE 5.1: Existence of product review concepts in ConceptNet

precision-recall scores. Our feature extraction depend on two threshold parameters: **support-threshold** in Association rule mining, **tf-Idf score** threshold. Each of these threshold are independent to each other. In our work, we are only able to finish the Tf-Idf based model. As we discussed that most of the noun phrases are missing in the conceptNet. To reduce this problem at some extent, we have considered noun phrases as feature whose part of word belongs to any feature and connected this noun phrases as child of those feature. This process significantly improved our precision and recall score in the model. Our result for each domain in mentioned in table 5.2. Precision recall curves are mentioned in figure 5.1, 5.2, 5.3 and 5.4 with respect to different domain.

Domain	Threshold(Tf-Idf)	Precision	Recall
Camera	0.0013	0.69	0.47
Phone	0.0016	0.65	0.49
Jukebox	0.0004	0.49	0.43
DVD Player	0.0013	0.89	0.18

TABLE 5.2: Precision Recall Score for each domain with respect to Tf-Idf threshold

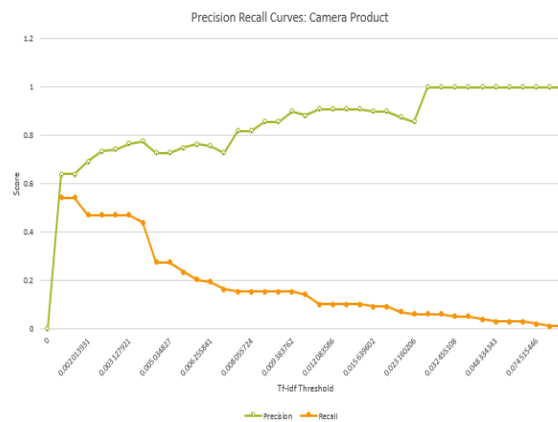


FIGURE 5.1: Precision Recall Curve - Camera Product

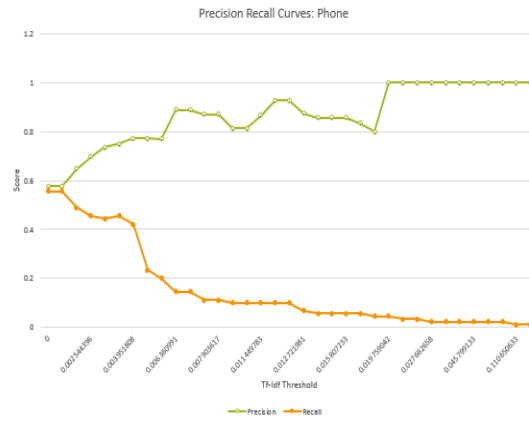


FIGURE 5.2: Precision Recall Curve - Phone Product

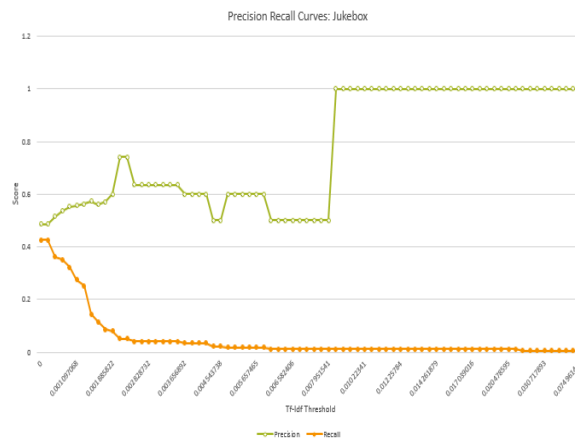


FIGURE 5.3: Precision Recall Curve - Jukebox Product

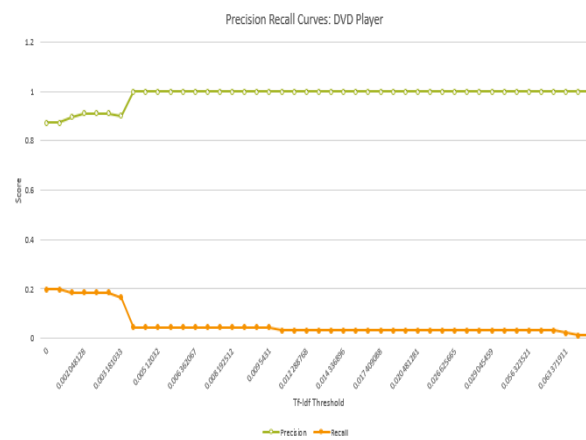


FIGURE 5.4: Precision Recall Curve - DVD Player Product

Domain	Nodes in Graph	Hierarchical	Synonym	Functional
Camera	334	4	114	0
Phone	304	4	116	8
Jukebox	503	7	249	0
DVD Player	112	0	14	0

TABLE 5.3: Irrelevant Features with respect to Relation classes(with noun phrase extension)

Domain	Nodes in Graph	Hierarchical	Synonym	Functional
Camera	116	4	63	0
Phone	134	4	76	7
Jukebox	171	7	118	0
DVD Player	76	4	52	2

TABLE 5.4: Irrelevant Features with respect to Relation classes(without noun phrase extension)

### 5.3.3 Topic Drift vs Relation Class measure

To measure topic drift, we have counted irrelevant features with respect to each relation class. Number of irrelevant features is directly proportional to topic drift in any domain. By counting the irrelevant features, we have noticed that most of the irrelevant feature belongs to Synonym classes. Also most of the irrelevance feature comes from relation 'IsA' and 'RelatedTo'. It is difficult to overcome this issue as most of the relations in feature tree belongs to 'RelatedTo' relation. Following table represents the topic drift for each domain with respect to each relation class(table 5.3 and 5.4):

# Chapter 6

## Snapshot

Here we will show all our User Interface snapshots. We have two main Panel: Faceted Navigation Panel and Review Panel. Each of these snapshots are shown in each subsection.

### 6.1 User Interface

Here is the snapshot of complete user interface 6.1:

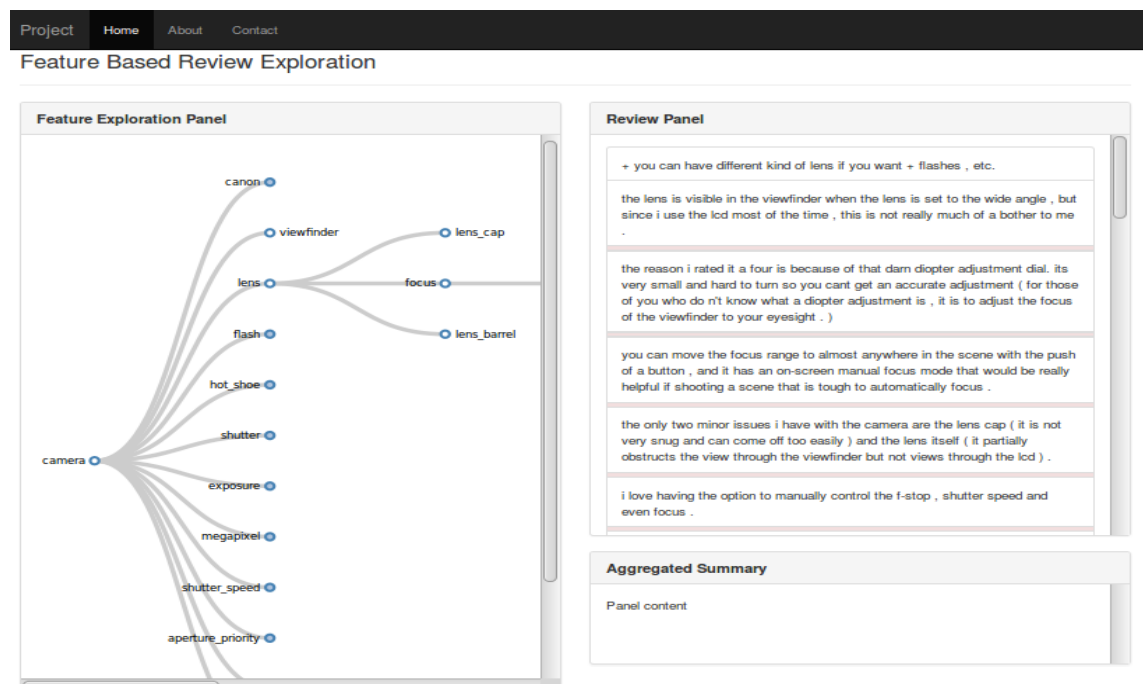


FIGURE 6.1: Complete User Interface

## 6.2 Faceted Navigation Panel

Snapshot 6.2 represents a feature ontology tree generated at tf-idf threshold 0.00519 for “camera” product domain.

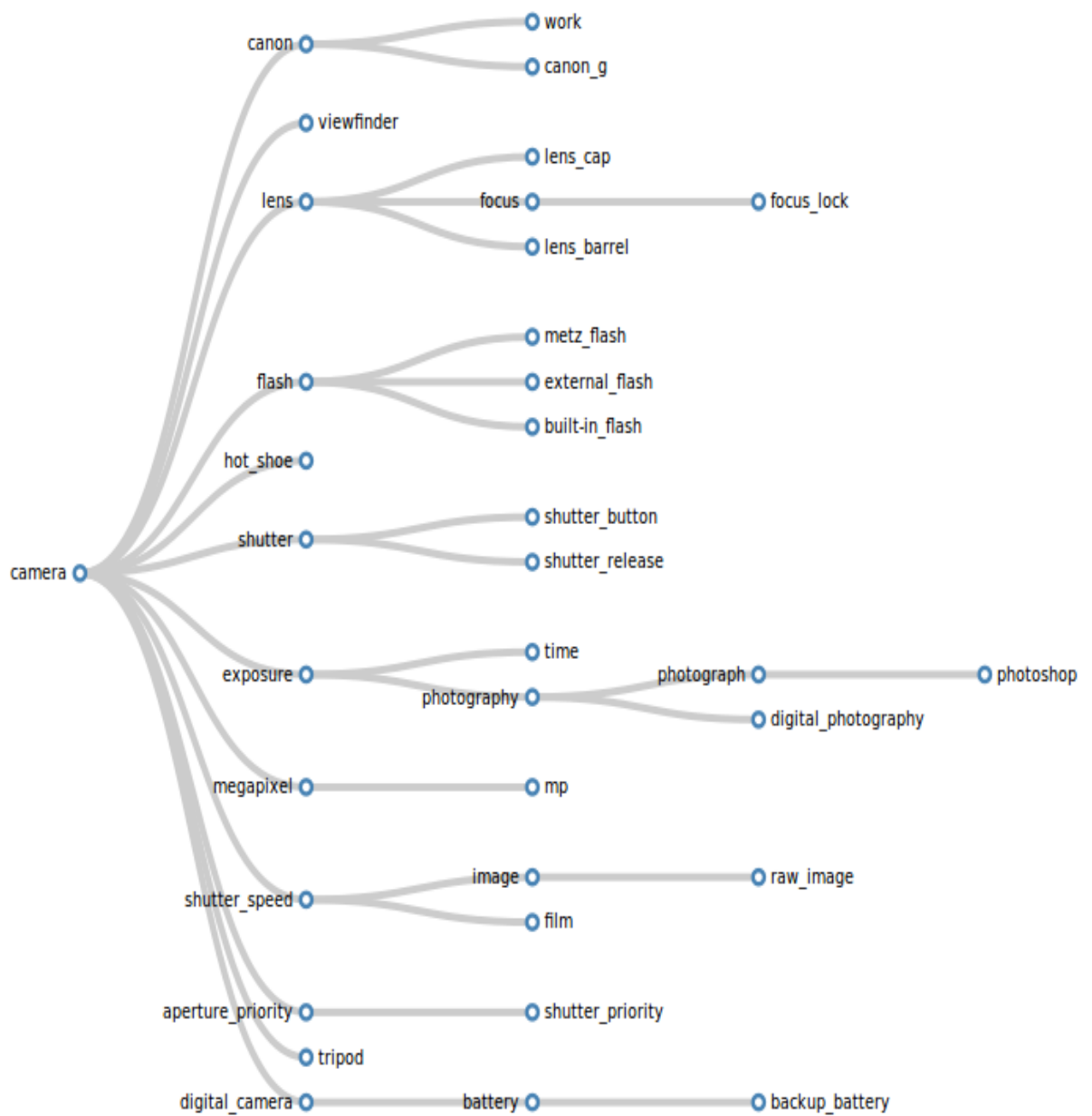


FIGURE 6.2: Faceted Navigation Panel for Camera Product

## 6.3 Review Panel

Snapshot 6.3 shows that the reviews are extracted for interested feature “lens”. In our review it includes all the review snippet which include lens as well as its child(focus, focus lock, lens barrel, lens cap).

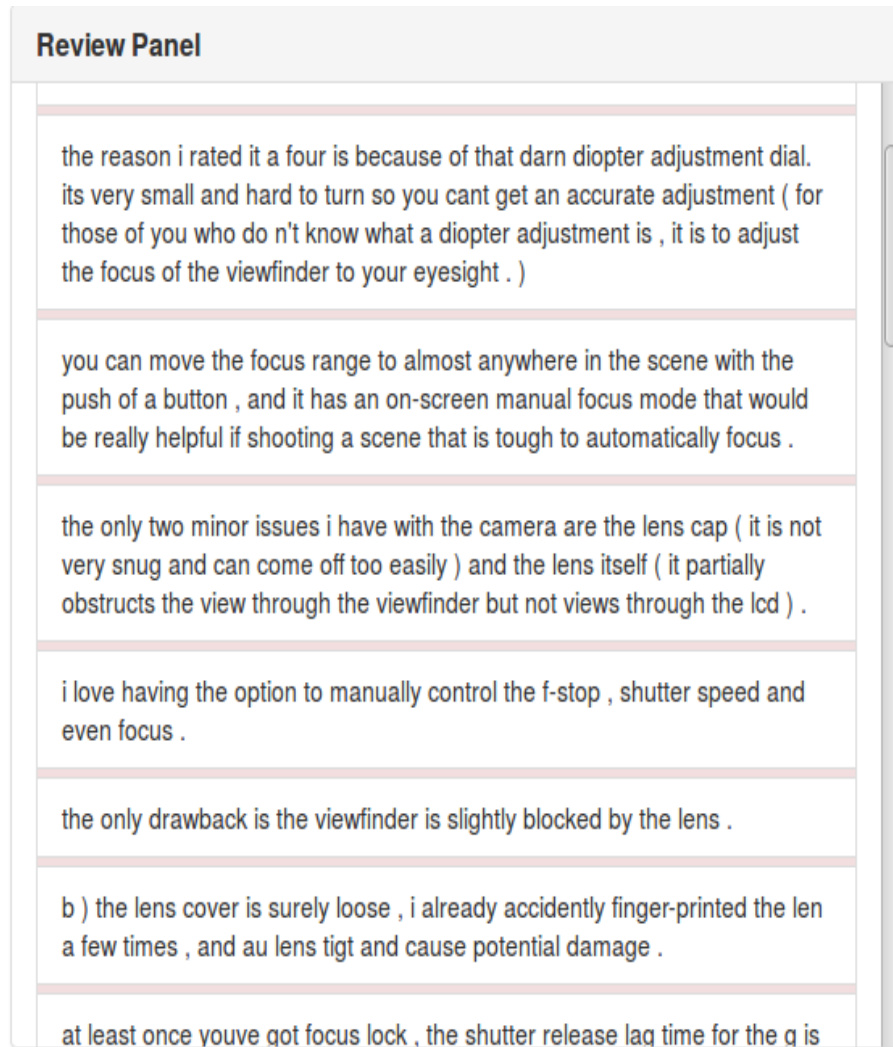


FIGURE 6.3: Extracted semantically meaningful reviews for “Lens” feature

## Chapter 7

# Conclusion

In our work, we have proposed “Faceted navigation” for a product reviews exploration which is our novel work. As user faces lot of issues while making online purchase, our research can help user to improve the experience of online shopping. In this dissertation, we have proposed a system architecture which extracts and build faceted navigation panel automatically which is an challenging task in relevant research area. Also we have proposed novel method to extract semantically meaningful information while extracting the reviews snippet from reviews. We have improved the existing model[6] for building feature ontology tree to improve product understanding by considering parent-child relationship between features. At last we would like to say that our research problem has combined two broad fields(faceted Navigation and text retrieval techniques) together to achieve our goal.



# Bibliography

- [1] 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*, pages 168–177. ACM, 2004.
- [2] Xiaowen Ding, Bing Liu, and Philip S Yu. A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pages 231–240. ACM, 2008.
- [3] Yue Lu, ChengXiang Zhai, and Neel Sundaresan. Rated aspect summarization of short comments. In *Proceedings of the 18th international conference on World wide web*, pages 131–140. ACM, 2009.
- [4] Daniel Tunkelang. *Faceted Search*. Morgan and Claypool Publishers, 2009. ISBN 1598299999, 9781598299991.
- [5] Hugo Liu and Push Singh. Conceptnet-a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004.
- [6] Subhabrata Mukherjee and Sachindra Joshi. Sentiment aggregation using conceptnet ontology. In *IJCNLP*, pages 570–578, 2013.
- [7] Amy K. Karlson, George G. Robertson, Daniel C. Robbins, Mary P. Czerwinski, and Greg R. Smith. Fathumb: A facet-based interface for mobile search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '06*, pages 711–720, New York, NY, USA, 2006. ACM. ISBN 1-59593-372-7. doi: 10.1145/1124772.1124878. URL <http://doi.acm.org/10.1145/1124772.1124878>.
- [8] Rasmus Hahn, Christian Bizer, Christopher Sahnwaldt, Christian Herta, Scott Robinson, Michaela Bürgle, Holger Düwiger, and Ulrich Scheel. Faceted wikipedia search. In *Business Information Systems*, pages 1–11. Springer, 2010.
- [9] Bing Liu. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167, 2012.

- 
- [10] Min Zhang and Xingyao Ye. A generation model to unify topic relevance and lexicon-based sentiment for opinion retrieval. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 411–418, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-164-4. doi: 10.1145/1390334.1390405. URL <http://doi.acm.org/10.1145/1390334.1390405>.
- [11] Wei Zhang, Shuang Liu, Clement Yu, Chaojing Sun, Fang Liu, and Weiyi Meng. Recognition and classification of noun phrases in queries for effective retrieval. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 711–720, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-803-9. doi: 10.1145/1321440.1321540. URL <http://doi.acm.org/10.1145/1321440.1321540>.
- [12] Ana-Maria Popescu, Bao Nguyen, and Oren Etzioni. Opine: Extracting product features and opinions from reviews. In *Proceedings of HLT/EMNLP on interactive demonstrations*, pages 32–33. Association for Computational Linguistics, 2005.
- [13] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Multi-facet rating of product reviews. In *Advances in information retrieval*, pages 461–472. Springer, 2009.
- [14] Eric Brill. A simple rule-based part of speech tagger. In *Proceedings of the workshop on Speech and Natural Language*, pages 112–116. Association for Computational Linguistics, 1992.