A FEATURE-BASED SENTENCE MODEL FOR EVALUATION OF SIMILAR ONLINE PRODUCTS[1]

Dr. Haiping Xu[*]
Computer and Information Science Department,
University of Massachusetts Dartmouth
285 Old Westport Rd, Dartmouth, MA 02747, USA
hxu@umassd.edu

Yuhan Zhang
Computer and Information Science Department,
University of Massachusetts Dartmouth
285 Old Westport Rd, Dartmouth, MA 02747, USA
yzhang5@umassd.edu

Richard DeGroof
Computer and Information Science Department,
University of Massachusetts Dartmouth
285 Old Westport Rd, Dartmouth, MA 02747, USA
rdegroof@umassd.edu

ABSTRACT

To help customers with their buying decisions, many e-commerce websites allow buyers to provide reviews for their purchased products; however, due to a large amount of reviews for many similar online products, consumers often feel it is difficult to determine which products have the most desirable features. In this paper, we propose a supervised learning approach to efficiently and effectively analyzing online product reviews and identifying the strengths and weaknesses of a product by its product features. The proposed approach uses a novel Feature-based Sentence Model (FSM), where a latent layer, called the feature layer, is introduced between review sentences and words. Once a model has been trained with sufficient labeled data points, it can identify the most related product feature, if any, for each review sentence. With the identified product features, we perform sentiment analysis for each sentence, and derive the weighted feature preference vectors for the review. Finally, we combine the results of all review comments for a product into a review summary. We demonstrate in two case studies that our approach works more effectively than existing approaches, and provides consumers a much easier way to find online products with the most desirable product features.

Keywords: E-commerce; Product reviews; Feature-based sentence model; Feature identification; Sentiment analysis

1. Introduction

Increasingly, shoppers are going online instead of purchasing products at physical stores. A major advantage of this trend is that an online merchant can offer a wider range of product type where warehousing facilities can be distributed across a country or even the world. With this increase in product diversity, it inevitably results in an increase in the complexity of product selection as goods may vary in their capabilities and quality. To help with product selection, online retailers often provide a forum where consumers can rate their purchases. The Average Star Rating (ASR) mechanism has been adopted by many e-commerce websites to help customers with their buying decisions; however, such ratings could be misleading as they do not necessarily reflect the actual product quality and strengths [Wei & Xu 2013]. For the same product, some customers may have greater or less expectations that cause them to rate the product differently. For example, a professional photographer might expect significantly more functionality from a digital camera; while a casual user might be perfectly happy with the same product. To make the ratings more meaningful, major e-commerce websites such as Amazon, allow users to provide reviews for the

[1] Paper manuscript submitted on January 29, 2018, revised on July 2, 2018, and accepted on October 26, 2018.
[*] Corresponding author

products or services they bought. These reviews are hand-written collections of text by which users justify their star ratings. Popular sources for consumer reviews are e-commerce websites like Amazon for general products, IMDb for movies, and Yelp for restaurants. A product review typically comments on the quality and various features of the product based on the specifications provided by the manufacturer or the seller. A popular online product listed at e-commerce websites such as Amazon, often receives hundreds or even thousands of reviews, contributing to what researchers term "information overload", or the inability for the consumer to make conscientious choices in the face of overwhelming and conflicting evidence [Ye et al. 2009]. The prior related research has found that for most readers, they rarely view online comments beyond the first 20 reviews [Pavlou & Dimoka 2006; Ye et al. 2009], as manual inspection of the reviews is usually too time-consuming for an interested reader to get the complete story about a product. Consequently, they may miss important information if they do not read the entire set of documents. So, to provide a better view about the products and save customers' time for browsing the reviews, we introduce a model-based approach that can automatically analyze all reviews of a product and output its strengths and weaknesses. Similar to the methodology proposed by Hu *et al.*, we summarize the reviews in terms of product features [Hu et al. 2016]. While their approach incorporated user-supplied terminologies, our model is iterative and incorporates new distributions of words occurring across features. As a result, our model becomes increasingly accurate as new data is incorporated.

Note that while reviews on Amazon are ranked according to customer perceptions of helpfulness [Yin et al. 2014], a positive factor in this assessment is review depth [Mudambi & Schuff 2010]. The reviews ranked higher in helpfulness tend to describe a larger number of features associated with the product. Therefore, such reviews will have more influence in decision making on product quality using our feature-based model. Additionally, research has attempted to automatically classify helpfulness in product reviews. Recognizing that user-supplied values take time to accumulate and that new reviews always begin at the end of the list, Zhang and Tran developed a linear classifier to automatically apply this ranking [Zhang & Tran 2010].

Analysis of online reviews requires a Natural Language Processing (NLP) capability. Latent Semantic Analysis (LSA) is such a technique that analyzes the relationships between a set of documents and the terms they contain [Deerwester et al. 1990]. LSA assumes that words with similar meanings will occur in similar documents. It uses a term-document matrix that records the occurrences of terms in documents. Evolved from LSA, probabilistic LSA or pLSA, was proposed as a statistical technique for analysis of co-occurrence of words and documents. It was first introduced by Hofmann in 1999, which has been used for text-based applications such as indexing and information retrieval [Hofmann 1999]. The pLSA model is a two-layer Bayes classifier for documents that contain multiple topics, and expresses data using three sets of variables, namely documents, topics and words, where topics are the latent (or hidden) variables. Assuming exchangeability of documents and words (i.e., different orderings of documents and words are processed identically as in the "bag-of-words" model), Latent Dirichlet Allocation (LDA) addresses the issues of over-fitting and scalability occurring in the pLSA approach [Blei et al. 2003]. In practice, the pLSA approach must maintain a number of variables, which grows linearly with the size of the documents, resulting in over-fitting. On the other hand, LDA introduces conditioning corpus-level variables that are assumed to be derived from a known distribution. Not only is this methodology more efficient in practice than the others, these variables permit the capture of intra-document statistics.

Although the aforementioned approaches are useful mechanisms for text mining, they are not accurate enough for the purpose of evaluating and comparing similar online products due to the nature of unsupervised learning. In this paper, we propose a supervised learning approach, called Feature-based Sentence Model (FSM). As a formal and novel model for text mining, it can be used to classify review sentences into different product features. Different from pLSA, as well as other unsupervised learning approaches such as LSA and LDA, FSM is a supervised learning technique in that it employs a labeled training data set for product feature identification. Unsupervised learning techniques have the advantage of not requiring labeled training data sets; however, they do not perform as well as supervised techniques, tending to have lower classification accuracy as we demonstrate in this paper. Additionally, their use in classification is not entirely unsupervised. For example, one of the results of inference of the LDA model is a matrix of distributions of latent topics over documents. By clustering, it is expected that similar distributions of latent topics will reflect similarities in documents. However, the identification of a cluster as representing one type of document or another would still require manual interpretation, at a certain degree of supervision. As methodological and practical contributions of this paper, we introduce a feasible supervised learning approach to developing an effective formal model that can accurately calculate feature scores for similar online products. To alleviate the requirement for large labeled data sets, and make our approach more efficient, we start with supervised learning, but also incorporate a semi-supervised learning mechanism, by which we bolster the training set with exemplary classifications that are highly reflective of particular topics. This is similar to the work of Yang *et al.*, whereby unlabeled data may be classified automatically and used to develop a model with less manual labeling [Yang et al. 2012].

By applying Bayes' rule, FSM can effectively identify a product feature, if any, in each sentence. When a feature is identified, we perform sentiment analysis of the sentence to find how the review author likes or dislikes the feature. When all review comments have been processed, we combine the results into a review summary in terms of the strengths and weaknesses of the product features. To illustrate the feasibility and effectiveness of our approach, we retrieve and process review comments for a number of similar products from Amazon. We show that by comparing the strengths and weaknesses of product features among similar products, our approach can greatly save customers' time for making decisions on selecting the most desirable online products.

The rest of the paper is organized as follows. Section 2 summarizes the work related to our approach. Section 3 presents the overall framework for feature-based sentence modeling and review preprocessing. Feature identification using FSM are discussed in Section 4. Section 5 presents sentiment analysis and review summary. To demonstrate the effectiveness of our approach, we present two case studies in Section 6. Section 7 concludes the paper and mentions future work.

2. Related Work

With the rapid growth of online reviews, review mining has attracted a great deal of attention. Pang *et al.* employed machine-learning techniques to label the polarity of IMDb movie reviews [Pang & Lee 2004]. They proposed to first extract the subjective portion of text with a graph min-cut algorithm, and then input them into a sentiment classifier. Rather than applying the straightforward frequency-based bag-of-words feature selection methods, Whitelaw *et al.* defined the concept of "adjectival appraisal groups" headed by an appraising adjective and optionally modified by words like "not" or "very" [Whitelaw et al. 2005]. Each appraisal group was further assigned four types of features: attitude, orientation, graduation, and polarity. Zhang and Varadarajan attempted to determine the author's opinion with different rating scales [Zhang & Varadarajan 2006]. They built regression models with a diverse set of features, and achieved competitive performance for utility scoring. Different from the above approaches, our FSM method is a statistical approach to modeling latent product features in product reviews, which can be directly applied at the sentence level to maximize the accuracy for text mining.

In machine learning and NLP, a topic model is defined as a type of statistical model for discovering the abstract "topics" that occur in a collection of documents. An early topic model for text mining, called Latent Semantic Indexing (LSI), proposed by Papadimitriou *et al.*, is an information retrieval technique based on the spectral analysis of the term-document matrix [Papadimitriou et al. 2000]. The authors showed that under certain conditions, LSI could capture the underlying semantics of the corpus and achieve improved retrieval performance. Blei *et al.* developed LDA [Blei et al. 2003], a commonly used topic model, which is a generalization of pLSA. LDA is a three-level hierarchical Bayesian model that allows documents to have a mixture of topics. LDA seeks to specify the underlying distribution of topics over documents and topics over words. In practice, LDA is often implemented using a Markov Chain Monte Carlo sampling technique known as Gibbs sampling, which is used where the exact estimation of a distribution is intractable due to complexity [Gelfand 2000]. LDA using Gibbs sampling has been applied to the domain of information retrieval by several practitioners. For example, Rigouste *et al.* used Gibbs sampling to cluster text into "themes" [Rigouste et al. 2007]. They showed that a Gibbs sampling algorithm is tractable and compares favorably to the basic Expectation-Maximization (EM) approach. Variations of Gibbs sampling also exist, which accelerate the process and/or increase the accuracy. For example, Porteous *et al.* proposed FastLDA, which assumes the majority of the topic distribution will be composed of a subset of the topics, permitting speed-ups [Porteous et al. 2008]. DeGroof and Xu proposed the variational Gibbs sampling approach in LDA to avoid the tendency of inconsistency for multiple trials on the same data set due to the usage of random numbers in the conventional Gibbs sampling approach [DeGroof & Xu 2017]. In further research, they introduced a parallelized LDA to analyze clustered textual data in online social media [DeGroof & Xu 2018]. Papanikolaou *et al.* introduced a modified collapsed Gibbs sampling methodology similar to collapsed variational Bayesian inference and pLSA in that a topic distribution is maintained for each term [Papanikolaou et al. 2016]. Different from typical implementations of LSA, pLSA and LDA, in FSM, a document (i.e., a review) is split into sentences. Since one sentence in a review generally addresses at most one product feature, our model requires generating only one matrix, namely the *feature-word* matrix. As such, FSM is a more efficient and accurate classifier, which can be more suitable for analysis of online reviews.

Another related approach is the naive Bayes classifier, which has been applied to text mining applications targeting different aspects of minable documents to reveal insight. For example, Wikarsa and Thahir mined emotional content of Twitter posts [Wikarsa & Thahir 2015]. They developed a text mining application using naive Bayes algorithm to classify emotions of Twitter users into one of six emotions, namely happiness, sadness, anger, disgust, fear, and surprise. Hasan *et al.* used a Bayesian classifier to identify sentiment of Amazon reviews as being positive or negative [Hasan et al. 2015]. Their implemented tool supports both English and Bangla text, and can be used to label the polarity of each opinion as weak, steady and strong. Akaichi *et al.* performed the same task on Facebook status updates [Akaichi et al. 2013]. They proposed a method based on Support Vector Machines (SVMs) and naive Bayes, to extract useful information about users' sentiments and behaviors. Unlike these works, we first classify Amazon reviews according to features, and then perform a sentiment analysis to determine and quantify what people really think about the products. Our approach considers a reviewer's opinion about a certain product feature as a piece of evidence to support either the strength or weakness of the product. This is useful as the ASR mechanism available on Amazon is too arbitrary and vague, making it unhelpful when similar online products are compared.

A commonly used supervised learning approach, called Multinomial Logistic Regression (MLR) [Taddy 2013], is used as a basis for comparison with FSM to demonstrate the effectiveness of our approach. Like other regression techniques, MLR assumes a relationship between predictors and feature labels, where common implementations fit a linear or quadratic model. MLR is similar to standard logistic regression approaches, but allows for classification of more than two categories, and has been applied to the problem of text categorization [Taddy 2013][Cao et al. 2011][Lam et al. 2016].

There is also previous work on sentiment analysis. Sentiment analysis, also known as opinion mining, analyzes people's opinions, sentiments, evaluations, appraisals, and attitudes for certain subjects. The term sentiment analysis was first proposed by Nasukawa and Yi in 2003 [Nasukawa & Yi 2003]. In general, sentiment analysis can be investigated at three different levels, namely the document level, the sentence level, and the entity and aspect level. In our approach for review analysis, since we need to determine whether a sentence expresses a positive, negative or neutral opinion, the sentiment analysis is performed at the sentence level. Such analysis is closely related to subjectivity classification [Wiebe et al. 1999]. The reason we do not use aspect and entity level sentiment analysis is that a typical review sentence describes only one feature. Therefore, for analysis of online reviews, aspect- and entity-level sentiment analysis would lead to the same result as sentence-level sentiment analysis.

From another perspective, sentiment analysis can also be divided into two types, namely regular opinions and comparative opinions [Jindal & Liu 2006]. A regular opinion expresses a sentiment on a single aspect or feature, while a comparative opinion describes and compares multiple aspects or features. Due to the nature of review comments, our approach treats reviews as regular opinions for sentiment analysis. Detecting sentiment from text usually contains two basic methodologies, which are symbolic techniques and machine learning techniques [Boiy et al. 2007]. Symbolic techniques assume a corpus is a "bag of words"; therefore, the relationships between the individual words are not considered [Turney 2002]. Kamps *et al.* used the lexical database WordNet to determine the emotional content of a word along different dimensions [Kamps et al. 2004]. WordNet is a large lexical database of words containing nouns, verbs, adjectives and adverbs, which are grouped into sets of cognitive synonyms (synsets) that describe different concepts. Based on WordNet, SentiWordNet has been developed as an extended lexical resource for opinion mining [Esuli & Sebastiani 2006]. To support sentiment analysis, SentiWordNet assigns each synset of WordNet three sentiment scores, namely positivity, negativity and objectivity. In this paper, we adopt the Stanford CoreNLP toolkit for sentiment analysis [Manning et al. 2014]. The Stanford CoreNLP toolkit is a Java-based annotation pipeline framework, which supports most of the common core NLP functions, including Part-Of-Speech (POS) tagger and sentiment analysis.

This work extends our preliminary work on analysis of online reviews using a Sentence Level Topic Model (SLTM), which attempted to classify review sentences into different classes corresponding to different product features [Zhang & Xu 2016]. In our current approach, we improved our previous method by applying a smoothing algorithm, which can deal with special cases in which certain words do not appear for a predefined feature in a training dataset. To demonstrate the advantages of our approach, we further compare our improved approach with well-known text mining approaches MLR and LDA, and show that our approach outperforms these for feature identification of online product reviews.

In Table 1 below, we summarize the strengths and weaknesses of the different text classification methods compared in the case studies (described in Section 6). The table shows that MLR is an efficient approach with good classification accuracy; however, it is a supervised learning method requiring a large labeled dataset, and also assumes a linear relationship between predictors and labels. LDA has the advantage of being an unsupervised learning approach, but it is not very accurate and slow to train. In contrast, our proposed FSM approach is a supervised machine learning method incorporated with a semi-supervised learning mechanism, which leads to good classification accuracy and fast run time. Though the approach assumes probabilistic independence between word tokens found in a sentence, the case studies show that this assumption has little impact on the experimental results.

Table 1: Strengths and Weaknesses of Compared Text Classification Methodologies

Method \ Aspects	Strengths	Weaknesses
MLR	Good classification accuracy; model is trained quickly	Supervised learning; assumes a linear relationship between predictors and labels
LDA	Unsupervised learning	Not as accurate in classification tasks; model can be slow to train
FSM	Good classification accuracy; supported by a semi-supervised learning mechanism; fast run time	Assumes probabilistic independence between word tokens found in a sentence

3. Feature-Based Sentence Modeling for Review Analysis

3.1. A Framework for Feature-Based Sentence Modeling

The framework for feature-based sentence modeling of online product reviews consists of four major components, namely review preprocessing, feature identification, sentiment analysis, and product summary. As shown in Fig. 1, the system is flexible and highly modularized, which supports processing a collection of reviews for a chosen online product. The reviews can be processed either sequentially or in parallel as we consider them being written by independent customers.

To identify product features and analyze sentiment, we first collect the reviews for a class of online products P (e.g. camera). Then we define a set of product features of P. As shown in Fig. 1, the input of the framework is a collection of reviews for product $p \in P$, and the output is a product summary that specifies the weighted feature preference for each predefined product feature. In the review preprocessing module, a review is split into a number of sentences that are parsed and tagged using an NLP tool. The tagged sentences are then sent to the feature identification modules, where we use FSM to calculate the probability of each feature. If a feature is identified in a sentence, the feature identification module outputs the sentence to the sentiment analysis module. Therefore, we output only the words with either tag NN or JJ to the feature identification module for further processing. The sentiment analysis module then calculates a feature score for the sentence as an integer between 0-4, with 0 representing very negative and 4 representing very positive.

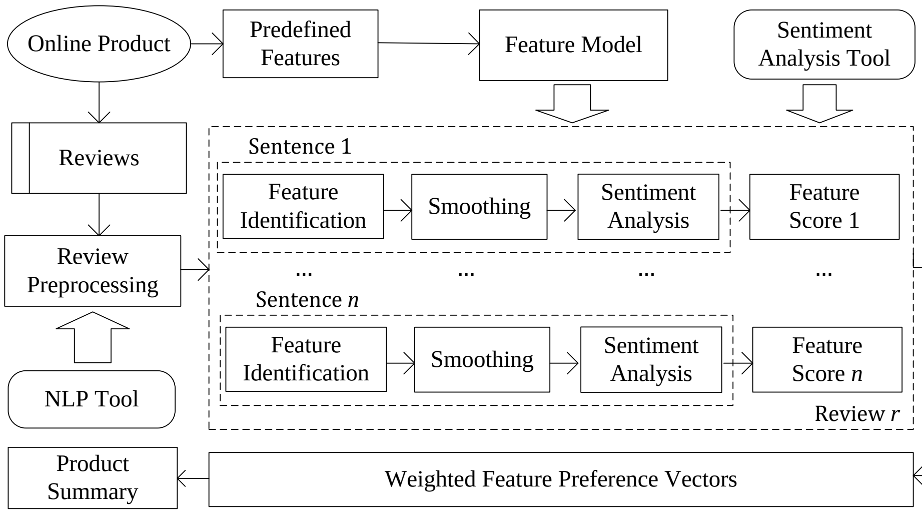


Figure 1: A Framework for Feature-based Sentence Modeling

Once we have calculated the weighted feature scores for each review r, we combine them in weighted feature preference vectors, and derive the product summary that shows how strongly each of the product features is supported or not supported by former buyers.

3.2. Review Preprocessing

A consideration in processing online reviews is to make sure sentences are correctly identified, and to understand how grammatical structures may affect the results. Prior research on processing of online reviews found that a single feature (or aspect in that work) is typically discussed in a single sentence, implying a consistent sentence structure [Jo & Oh, 2011]. Although online reviews are written by individuals, who might not fully comply with grammar and punctuation rules [Lacob & Harrision, 2013], since our approach adopts the bag-of-words model for feature identification, improper grammatical structures will not affect the results. In addition, word choice may contain slang, but if the word is identifiable by a POS tagger, it is reasonable to expect that it will be identified during sentiment analysis.

The review preprocessing module processes a single review at a time, each of which is split into a number of sentences. For each sentence, we use an existing POS tagger (e.g., Stanford CoreNLP [Manning et al. 2014]) to indicate each word in the sentence as noun, verb, adjective, adverb or other tags. We use the following sentence as an example to show how this POS tagger parses a sentence:

"The resolution is high and adjustable."

Using the CoreNLP toolkit, the parser outputs the sentence with the following POS tags:

The_DT resolution_NN is_VBZ high_JJ and_CC adjustable_JJ ._.

In the above tagged sentence, the tags DT, NN, VBZ, JJ, and CC stand for "Determiner", "Noun", "Verb, third person singular present", "Adjective", and "Coordinating Conjunction", respectively. After detecting all sentences in a review and parsing each sentence into POS tags, we save each sentence along with the POS tag information into a local file, which will be used to identify product features as described in the following sections.

Based on our observations, a feature described in a sentence is usually determined by nouns or adjectives. Thus, we simplify the model by removing those words that are not nouns or adjectives including the noise words (also called stop words), such as "you", "I", "a", "the", etc. In our approach, a sentence is considered as a "bag of words", and only the words with either tag NN or JJ are output to the feature identification module for further processing. In addition, for each word, we consider the lemma only, which is the dictionary form of a set of words. For example, words "likes", "like", "liked" are forms of the same lexeme, with "like" as the lemma. Finally, to deal with some sentences that are not likely describing a feature, processed sentences with less than two words are ignored since a meaningful sentence should at least contain a topic word and a sentiment word.

4. Feature Identification Using FSM

4.1. Feature-Word Matrix

To develop a feature-based sentence model for a product type P, we predefine a list of product features F based on customers' interests as well as product specifications. Suppose we have defined K product features for P. We can create a *feature-word* matrix Φ with K rows and V columns, where V is a dynamic number that increases by 1 each time a new word appears. Initially, Φ has K rows and 0 column, which is an empty matrix due to the emptiness of the initial vocabulary list. After the matrix has been expanded, each row of the matrix represents the distribution of words in the corresponding feature, while each column indicates how the corresponding word has co-occurred with different features. The meanings of each row and each element in Φ are summarized as follows.

Φ_j: distribution of words in feature F_j (j ranges from 0 ~ K-1)

$\Phi[j, l]$: counts of the word with index l occurring in feature F_j (l ranges from 0 ~ V-1)

The matrix is used to record feature-word co-occurrence information with each cell initially set to 0. To expand the column (vocabulary) in matrix Φ and increase the number in each cell of Φ, we first manually create a training dataset with labeled sentences as follows. For a review sentence, we tag it with a proper product feature if it contains one. For example, the sentence "The resolution is high and adjustable." describes the product feature "Resolution"; therefore, we put a "#Resolution" tag at the end of the sentence as shown below.

"*The resolution is high and adjustable*." #Resolution

As additional examples, the following two review sentences are tagged by the product features "Price" and "ViewScreen", respectively.

"*The price is right for the quality, and it is a fantastic everyday type of camera.*" #Price

"*The LCD screen does a great job showing a high quality image.*" #ViewScreen

Note that a customer may use various words or phrases to describe a feature. For example, words or phrases "display" and "LCD screen" all refer to the same feature "ViewScreen". Since only nouns and adjectives are considered inputs of the FSM model, each labeled data point is simplified as a bag of nouns and adjectives with its associated feature tag. Thus, the data points for the aforementioned three tagged sentences are recorded as follows:

"*resolution, high, adjustable*" #Resolution

"*price, everyday, type*" #Price

"*LCD, screen, job*" #ViewScreen

Each product type is accompanied by its own set of features, and there tends to be a degree of consistency between the terminologies associated with each feature. For example, in the final training set of 1500 labeled sentences used in the case studies (described in Section 6), the word "price" represented 35% of the total word frequency for that feature but only 1.8% for all other features combined. While, this word does occur across features, it contributes to the classification of "Price" most significantly. Similarly, the word "shutter" represented 38% of the word frequency for that feature and did not occur in any other. Some words might occur more commonly across features. For example, the word "big" occurred in four separate features, "Setup", "Lens", "Viewscreen" and "Battery". However, in the features other than "Viewscreen", it occurred much less often. We can see that words which occur across features are not uncommon, but the frequency with which they occur tends to differ, as in the preceding examples.

The algorithm for setting up the feature-word matrix Φ is presented as Algorithm 1. In this algorithm, we first initialize Φ with an empty vocabulary list *vl*. Then for each data point *d*, let *f* be the feature (tag) of *d*, and *featureIndex* be the index of *f* in the feature list *fl*. For each word *w* in *d*, check if *w* appears in *vl*. If *w* is in *vl*, let *wordIndex* be the index of *w*; otherwise, add *w* to *vl* and expand Φ by one column for the new word *w*, and let *wordIndex* be the index of the last column of the matrix. Finally, the feature-word co-occurrence count Φ[*featureIndex*][*wordIndex*] is increased by one. Note that the feature-word matrix is scalable; in other words, when new labeled data points become available, the feature-word matrix Φ can be easily updated.

Algorithm 1: Set up Feature-Word Matrix Φ

Input: Matrix Φ, feature list *fl*, vocabulary list *vl*, and a training dataset with labeled data points
Output: updated matrix Φ

```
1.  Initialize matrix Φ with vl as an empty list
2.  for each data point d from the training dataset
3.    let f be the feature of d, and featureIndex be the index of f in fl
4.    for each word w in d
5.      if vl contains w
6.        let wordIndex be the index of w in vl
7.      else
8.        add w to vl, add 1 column to Φ, and set wordIndex = |vl| - 1
9.      increase Φ[featureIndex][wordIndex] by 1
10. return matrix Φ
```

Table 2 shows an example of the feature-word matrix Φ created using several sentences as a training dataset. In this example, we record 13 predefined features including a "Null" feature for sentences that do not address any product feature. In addition to the list of features, we also record a vocabulary, which is used to keep track of 19 words that are related to the 13 features. Note that in the vocabulary list, there are no duplicated words. The number for a feature-word pair denotes the number of occurrences of a word appearing for a certain feature. For example, Φ[6][12] equals 3 indicates that in the training data set, the word "bag" appears 3 times in review sentences related to the product feature "Accessory".

Table 2: An Example of the Feature-Word Matrix Φ

Feature \ Word	**hand**	**heart**	**directions**	**learning**	**price**	**Display**	**big**	**Features**	**modes**	**Light**	**lens**	**slim**	**bag**	**flash**	**Zoom**	**battery**	**shutter**	**speed**	**resolution**
Null	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SetUp	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Price	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Mode	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
SizeWeight	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
Lens	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0
Accessory	0	0	0	0	0	0	0	0	0	0	0	1	**3**	0	0	0	0	0	0
Flash	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0
ViewScreen	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
Stabilization	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Battery	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
Shutter	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	3	0
Resolution	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3

4.2. Feature Identification

Once we have updated the feature-word matrix using all data points from the training dataset, we can use it to create a discriminative model to classify sentences into different classes. Let F_0, …, $F_{K\text{-}1}$ be the list of features including the “Null” feature, and $P(F_j)$, where $0 \leq j \leq K\text{-}1$, be the probability that feature F_j appears in a sentence. Assume each sentence describes at most one feature, then the probability $P(F_j)$ can be calculated as in Eq. (1).

$$P(F_j) = \frac{N(F_j)}{\sum_{i=0}^{K-1} N(F_i)} \tag{1}$$

where $N(F_j)$ is the number of times that feature F_j has appeared in the training dataset, and $\sum_{i=0}^{K-1} N(F_i)$ is the total number of data points in the training dataset. Obviously, the summation of probabilities $\sum_{j=0}^{K-1} P(F_j) = 1$.

Let $P(F_j \mid S)$ be the conditional probability that feature F_j appears given sentence S, where $0 \leq j \leq K\text{-}1$, and $\sum_{j=0}^{K-1} P(F_j \mid S) = 1$. The feature in sentence S *Feature_S* can be determined by Eq. (2).

$$Feature\_S = \begin{cases} F_m & \text{if } P(F_m \mid S) > 0.5 \\ Null & \text{otherwise} \end{cases} \tag{2}$$

$$\text{where} P(F_m \mid S) = \max(P(F_1 \mid S), P(F_2 \mid S), \ldots, P(F_{K-1} \mid S))$$

Note that in Eq. (2), 0.5 is the threshold for feature identification. If a sentence contains words that describe two or more features, the feature with the highest probability given the set of words S from that sentence is determined as Fm if $P(Fm \mid S) > 0.5$. However, if there is a tie among the features, none of them would satisfy the condition $P(Fm \mid S) > 0.5$. In this case, the feature in sentence S is set as F_0, i.e., the “*Null*” feature.

Now consider the calculation of conditional probability of sentence S having feature F_j. By the Bayes’ rule, $P(F_j \mid S)$ can be calculated as in Eq. (3.1).

$$P(F_j \mid S) = \frac{P(S \mid F_j) * P(F_j)}{P(S)} \tag{3.1}$$

In our model, we treat S as a bag of n independent words, where each word has a unigram meaning. Let S be a set of words $\{w_1, \ldots, w_n\}$, where $n < V$, then Eq. (3.1) can be rewritten as in Eq. (3.2).

$$P(F_j \mid S) = \frac{P(w_1, \ldots, w_n \mid F_j) * P(F_j)}{P(w_1, \ldots, w_n)} \tag{3.2}$$

where $P(w_1, \ldots, w_n \mid F_j)$ and $P(w_1, \ldots, w_n)$ are defined as in Eq. (4.1-4.2).

$$P(w_1, \ldots, w_n \mid F_j) = \Pi_{i=1}^{n} P(w_i \mid F_j) \tag{4.1}$$

$$P(w_1, \ldots, w_n) = \Pi_{i=1}^{n} P(w_i) \tag{4.2}$$

Note that $P(w_i \mid F_j)$, where $1 \leq i \leq n$, and $0 \leq j \leq K\text{-}1$ is the conditional probability that word w_i appears in a sentence S given feature F_j contained in S. Since Φ_j is the distribution of words in feature F_j, $P(w_i \mid F_j)$ can be derived from Φ_j as in Eq. (5).

$$P(w_i \mid F_j) = \frac{\Phi[j, index(w_i)]}{\sum_{l=0}^{V-1} \Phi[j, l]} \tag{5}$$

where $index(w_i)$ is the column index of word w_i in Φ and $\sum_{l=0}^{V-1} \Phi[j, l]$ is the sum of word counts in feature F_j. Since each $P(w_i \mid F_j)$ could be a very small decimal, to avoid accuracy overflow errors in calculating $P(w_1, \ldots, w_n \mid F_j)$, we first calculate the logarithm of $P(w_1, \ldots, w_n \mid F_j)$ as in Eq. (6.1), and then derive $P(w_1, \ldots, w_n \mid F_j)$ as in Eq. (6.2).

$$\ln(P(w_1, \ldots, w_n \mid F_j)) = \sum_{i=1}^{n} (\ln(\Phi[j, index(w_i)]) - \ln(\sum_{l=0}^{V-1} \Phi[j, l])) \tag{6.1}$$

$$P(w_1, \ldots, w_n \mid F_j) = e^{\ln(P(w_1, \ldots, w_n \mid F_j))} \tag{6.2}$$

Finally, $P(w_i)$, where $1 \leq i \leq n$, is the probability that word w_i appears in a sentence. $P(w_i)$ can also be calculated using the feature-word matrix; however, this calculation is not necessary, as it is a constant for all features, so is $P(w_1, w_1, \ldots, w_n)$. Since $\sum_{j=0}^{K-1} P(F_j \mid S) = 1$, we can calculate $P'(F_i \mid S)$ as in Eq. (7.1), and then normalize it into $P(F_j \mid S)$ using Eq. (7.2).

$$P'(F_j \mid S) = P(w_1, \ldots, w_n \mid F_j) * P(F_j) \tag{7.1}$$

$$P(F_j \mid S) = \frac{P'(F_j \mid S)}{\sum_{j=0}^{K-1} P'(F_j \mid S)} \tag{7.2}$$

4.3. An Effective Smoothing Algorithm for Feature Identification

While calculating $P(w_i \mid F_j)$ as in Eq. (5), it is possible that w_i does not appear for feature F_j according to the training dataset. In this case, $\Phi[j, index(w_i)] = 0$, which results in both of the values $P(w_i \mid F_j)$ and $P(w_1, \ldots, w_n \mid F_j)$ being 0. By Eq. (3.2), the probability $P(F_j \mid S)$ becomes 0 too. This issue is due to the imperfection of the training dataset that fails to cover all relationships between words and features. To resolve this issue, we need to apply a smoothing algorithm to the feature-word matrix Φ.

There are quite a few existing smoothing algorithms, and one simple algorithm is called additive smoothing (also known as Laplace smoothing) [Manning et al. 2008]. Using additive smoothing, while calculating the probability $P(w_1, \ldots, w_n \mid F_j)$, a constant number (usually 1) is added to all the numerators and n is added to the denominator. Therefore, all the numerators are greater than 0 and the denominator is still equal to the sum of all the numerators. Another useful smoothing method is called Kneser-Ney smoothing [Teh 2006], which is primarily used to calculate the probability distribution of n-grams in a document based on their histories. Let m be the number of elements in Φ_j that equal 0, and n be the number of elements in Φ_j that do not equal 0. This algorithm sets $\Phi[j, i]$ to the smallest positive number φ in Φ_j if $\Phi[j, i] = 0$, and deducts $\varphi*(m/n)$ from all other elements in Φ_j. Although Kneser-Ney smoothing algorithm has been widely considered the most effective method for n-grams, our approach uses unigram, where the words are considered independent, and their order does not matter. Therefore, for an element with $\Phi[j, i] = 0$, i.e., no evidence to show the associated words (at column i) is related to feature F_j, it makes sense to replace it only by a minimal value of 1, as in the additive smoothing algorithm. A new combined smoothing algorithm that takes advantages of both the two approaches is illustrated in Algorithm 2.

In Algorithm 2, elements equal to 0 are increased by 1 as in the additive smoothing algorithm; while elements that are not equal to 0, an offset m/n, where m is the number of elements in Φ_j that equal 0, and n is the number of elements in Φ_j that do not equal 0, is deducted from the elements as in the Kenser-Ney smoothing algorithm. Comparing to the Kenser-Ney smoothing algorithm, when the smallest positive number φ in Φ_j is greater than 1, adding 1 only to those elements equal to 0 can help to minimize the negative impacts of unrelated words to feature F_j. Note that after the smoothing process, $\sum_{l=0}^{V-1}\Phi[j,l]$) remains the same for each feature F_j.

Algorithm 2: Smoothing Algorithm
Input: Matrix Φ, feature list *fl*, vocabulary list *vl*
Output: Smoothed matrix Φ

```
1.  set $K = |fl|$ and $V = |vl|$
2.  for each feature $F_j$ in fl, where $0 \le j \le K-1$
3.    initialize m and n to 0.
4.    for each element $\Phi[j, i]$, where $0 \le i \le V-1$
5.      if $\Phi[j, i] == 0$, increase m by 1
6.      else increase n by 1
7.    set $offset = m/n$
8.    for each element $\Phi[j, i]$, where $0 \le i \le V-1$
9.      if $\Phi[j, i] == 0$ increase $\Phi[j, i]$ by 1
10      else $\Phi[j, i] = \Phi[j, i] - offset$
11. return Φ
```

4.4. Limitations of FSM for Feature Identification

The primary limitation of the FSM feature identification methodology is the degree of supervision required. For a product, the features must first be enumerated, and an initial training dataset must be assembled. We mitigate the overall supervision by using a semi-supervised approach in which additional classifications are made automatically using trained FSM. On the other hand, unsupervised learning approaches such as LDA do not need an initial training dataset, but still require as input the number of underlying hidden topics. This choice will affect the results, and the methodology is most accurate when the predefined number of underlying hidden topics is close to the true value. Thus, there is a tradeoff between using the FSM model and an unsupervised learning approach such as LDA, but the case studies in Section 6 show our approach outperforms major existing approaches.

5. Sentimental Analysis and Review Summary

5.1. Sentiment Analysis

Once we have identified product features from review sentences, the next task is to analyze the customer's preference of the identified feature. Here we utilize Stanford Sentiment Analysis toolkit [Socher et al. 2013] for this purpose. In previous research, most of the sentiment prediction approaches consider words as isolated ones, and assign them either positive or negative points according to a sentiment dictionary such as SentiWordNet. By doing this, the order of words is ignored and some critical information might be lost. In contrast, the Stanford Sentiment Analysis toolkit utilizes a deep learning approach, which builds up a representation of whole sentences based on the sentence structure. As their model is built on phrase level rather than word level, the model can be used to compute the sentiment of phrases based on the composition of words. More specifically, the Stanford Sentiment Analysis model uses a Recursive Neural Tensor Network (RNTN) based on the grammatical structure of a sentence. The training dataset of this model is from Stanford Sentiment Treebank, and it is worth mentioning that users can help to improve this model while using it. The toolkit first splits the sentence into phrases, then phrases into words, forming a binary tree. The sentiment is first calculated for each word at the lowest level. Then the results are recursively combined upward considering the effects of POS such as conjunctions and negations, and finally producing the sentence-level sentiment result. A more detailed description of this process can be found in reference [Socher et al. 2013]. Since we assume one sentence contains at most one feature, the sentiment score of a whole sentence is also the score of the feature, if existing. According to reference [Socher et al. 2013], the model's accuracy on a single sentence classification is above 80%, while the accuracy of predicting fine-grained sentiment for all phrases could be even higher.

5.2. Review Summary

Our approach is to extract features and calculate sentiment scores at the sentence level. Once we have recorded the feature score for each sentence, we can summarize the user preferences of the predefined product features for a certain product. The summary is based on the procedure of combining weighted feature preference scores. The procedure is described as in Algorithm 3.

In Algorithm 3, all review results are combined to evaluate how customers like or dislike the predefined product features of a product. Note that if a review contains multiple review sentences related to feature *f*, according to the algorithm, only the last sentence given by the reviewer is considered.

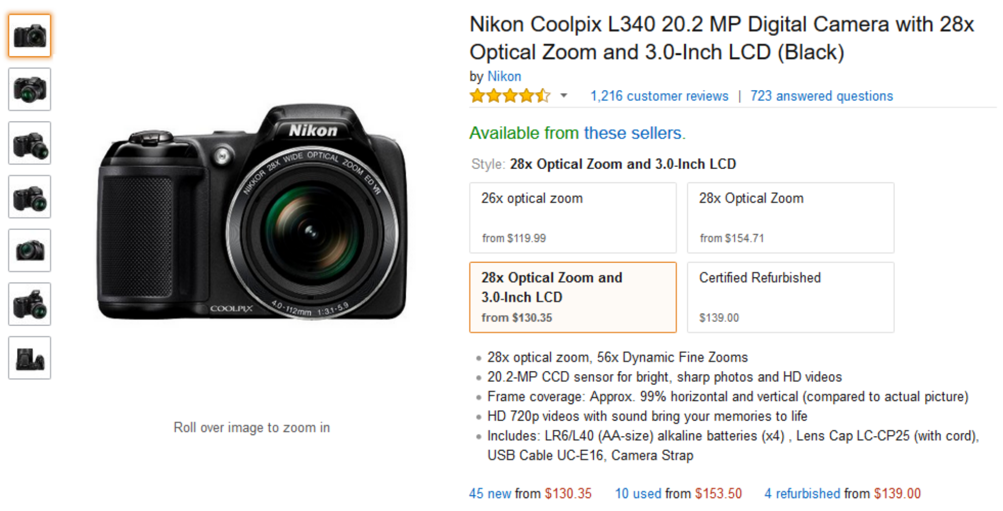
Algorithm 3: Feature Preference Score Combination

Input: Feature scores of all reviews for a product *p*
Output: Vectors LIKE and DISLIKE indicating how customers like or dislike product *p*, respectively, in terms of the product features.

```
1.  let K be the total number of predefined features of product p
2.  initialize two K-dimension vectors LIKE and DISLIKE to 0, where
3.  for each review r for product p
4.    initialize K-dimension vector PREFER to 0
5.    for each sentence s in r
6.      let f be the feature in s, and fIndex be the index of feature f in the vectors
7.      let sentiScore be the sentiment score of s
8.      switch sentiScore
9.        case 0: PREFER[fIndex] = -1;  break; // very negative
10.       case 1: PREFER[fIndex] = -0.7;  break; // negative
11.       case 1: PREFER[fIndex] = 0;  break; // neutral
12.       case 3: PREFER[fIndex] = 0.7;  break; // positive
13.       case 4: PREFER[fIndex] = 1;  break; // very positive
14.   for each element i in PREFER, where 0 ≤ i ≤ K-1
15.     if PREFER[i] > 0 LIKE[i] += PREFER[i]
16.     else DISLIKE[i] += PREFER[i] * (-1)
17. return vectors LIKE and DISLIKE
```

6. Case Studies

To demonstrate the feasibility of our approach, we use an example of online products from Amazon website. The product is "Nikon Coolpix L340 Digital Camera (Black)" with the product's basic information shown in Fig. 2.



Nikon Coolpix L340 20.2 MP Digital Camera with 28x Optical Zoom and 3.0-Inch LCD (Black)

by Nikon

1,216 customer reviews | 723 answered questions

Available from these sellers.

Style: 28x Optical Zoom and 3.0-Inch LCD

26x optical zoom from $119.99

28x Optical Zoom from $154.71

28x Optical Zoom and 3.0-Inch LCD from $130.35

Certified Refurbished $139.00

- 28x optical zoom, 56x Dynamic Fine Zooms
- 20.2-MP CCD sensor for bright, sharp photos and HD videos
- Frame coverage: Approx. 99% horizontal and vertical (compared to actual picture)
- HD 720p videos with sound bring your memories to life
- Includes: LR6/L40 (AA-size) alkaline batteries (x4) , Lens Cap LC-CP25 (with cord), USB Cable UC-E16, Camera Strap

45 new from $130.35 10 used from $153.50 4 refurbished from $139.00

Figure 2: An Example of Online Products from Amazon

From the description, it is easy to see that the product is a good one due to its high ASR; however, before making purchase decisions, a customer still would like to know different features of the product. For example, does the battery last long enough? Is the camera small and light enough for a long trip? What's the picture quality like? By automatically analyzing the review comments, our approach may help customers to answer such questions.

6.1. Product Features

To evaluate the product features of digital camera, we define 13 customer-interested features including the "Null" feature, listed as follows.

1. **Null**: is a dummy feature that indicates a sentence describing no specific feature.
2. **Accessory**: refers to the quality or availability of camera accessories, such as the availability of SD card option.
3. **ViewScreen**: is also known as LCD or viewfinder, which is a device used to display images.
4. **Price**: indicates whether the price of the camera is reasonable or not.
5. **SizeWeight**: refers to the appearance of a camera in terms of its size and/or weight.
6. **Stabilization**: is a feature that indicates whether a camera can effectively prevent or compensate for unwanted camera movement.
7. **Mode**: refers to manual mode or automatic mode that can be used in various situations.
8. **Battery:** refers to options related to powering the camera with portable batteries.
9. **Shutter**: is a device associated with a camera that allows light to pass for a determined period of time.
10. **Lens**: is an optical device on a camera that can change the focus of a light beam through refraction. The quality of lens has a strong impact on the quality of the camera.
11. **SetUp:** refers to the process of getting a camera working, such as understanding how to use the camera and how to use options like built-in wifi.
12. **Flash:** is a device that allows a camera to capture images in low light. This feature refers to the quality of the flash as well as the experience of using it.
13. **Resolution**: is a feature of picture quality, and higher resolution typically indicates higher picture quality.

Note that the "Null" feature is a dummy one that indicates a sentence does not describe any feature of the camera. For example, sentences such as "I love this camera so much," "My last camera was the Canon Powershot A495," or "So happy I got this," do not make comments on any product feature. Instead, they simply either state a fact or express a general feeling about the camera. Since our approach is evidence-based, sentences with the "Null" feature do not contribute as evidence for product quality, thus they are removed in the following case studies.

6.2. Case Study 1

To demonstrate the performance of our technique as compared with other text mining methodologies, we downloaded a series of reviews that contain over 2000 sentences. Initially, we labeled 441 sentences with product features used as an initial training dataset. In addition, we labeled 400 sentences to be included in a testing dataset to assess our approach as well as two related approaches, namely MLR and LDA. Table 3 shows the accuracy of the different methods and different training dataset sizes equal to 700, 900, 1100, 1300 and 1500 with the “Null” feature removed.

Table 3: Accuracy without the “Null” Feature for Different Training Dataset Size and Method

Size / Method	441	700	900	1100	1300	1500
MLR	0.674	0.826	0.872	0.895	0.907	0.913
LDA	0.756	0.866	0.837	0.843	0.855	0.831
FSM	0.851	0.891	0.886	0.886	0.922	0.946

From Table 3, we can see that LDA, as an unsupervised learning approach, did not perform well compared to the other two approaches, although our implementation of it uses a modified version of Gibbs sampling with improved performance [DeGroof & Xu 2017]. MLR generally performed worse than FSM but better than LDA when the training dataset is large enough. MLR is a supervised learning technique, like FSM. It fits a vector of weights for each feature in the feature/term matrix. Unknown samples are classified as the feature whose weight vector produces the greatest value when multiplied by the predictor values (word counts) for that sample. MLR may suffer from over-fitting, like pLSA, due to the number of parameters involved.

The results show that when the training dataset size is 441, FSM performs better than the other two approaches, with an accuracy of 85.1%. It is worth noting that our approach uses a semi-supervised learning method. To evaluate our approach with larger training datasets, we applied FSM to unlabeled data points, and utilized the classifier as a helper to label new data points. For those data points that contain features with high probabilities, say over 0.8, we assumed the labeling was correct; otherwise, we manually labeled them. Using this approach, we developed the different sizes of the training datasets. The performance of each methodology tended to increase as the training set size increased. In most cases, FSM had the highest accuracy that may go up to 94.6%, which is quite satisfactory.

6.3. Case Study 2

In our second case study, we compared six different digital cameras sold at Amazon, and provided the review summaries by analyzing their reviews using our FSM approach. The six selected digital cameras are listed in Table 4, which have similar prices around $200 and are all sold by top-100 camera sellers at Amazon. Moreover, they all have above 4.0 ASR; thus, it would be hard for a buyer to determine which one could be the most suitable one to purchase.

Table 4: Six Different Digital Cameras Listed at Amazon

Product Name	Average Star Ratings	Price	Reviews
Nikon Coolpix L340	4.3	$130	1,216
Canon PowerShot SX610	4.2	$229	108
Canon EOS Rebel T5	4.7	(used) $299	664
Samsung WB1100F	4.2	$263	485
Canon PowerShot SX530	4.4	(used) $204	191
Fujifilm FinePix HS25EXR	4.4	(used) $213	89

The review summaries for comparing the six selected cameras are presented in Fig. 3. In the figure, the left-side bars indicate the weighted DISLIKE scores for each product feature; while the right-side bars indicate the weighted LIKE scores for the features. The figure reflects the complete sentiment scores for all features and reviews of that product. A larger number of reviews will increase the scores but only if the sentiment is above or below certain thresholds. The scores for the same feature are comparable across the six different cameras that have different numbers of reviews. In addition, for each product, the scores for various features are also comparable, which indicate the strengths and weaknesses of the product in terms of its product features. From the figure, we can see that Nikon Coolpix L340 has a really bargain price, good size & weight; however, its view screen, battery and shutter are not good enough. Canon PowerShot SX610’s price is also great, it has a good size & weight and lens; however, its mode and view screen are not satisfactory. Cannon Rebel T5’s lens got more complaints proportionately than the others, and the camera also did not seem to be worth the cost; thus, it does not look like a good deal. The Samsung WB1100F has a great price score and good size & weight, but its setup and battery are its weaknesses. Canon PowerShot SX530 got a great lens score and a decent price score. Customers did not like its mode or size & weight, though. Fujifilm FinePix HS25EXR got great lens and size & weight scores and, in fact, many of its positive features outweighed its negative ones 2 to 1. Even though Nikon Coolpix L340 must be a very popular product considering the number of reviews it received, among the six products, Fujifilm FinePix HS25EXR, Samsung WB1100F and Canon PowerShot SX610 seem to be more desirable products than the other three because they have more strengths than weaknesses in terms of their product features.

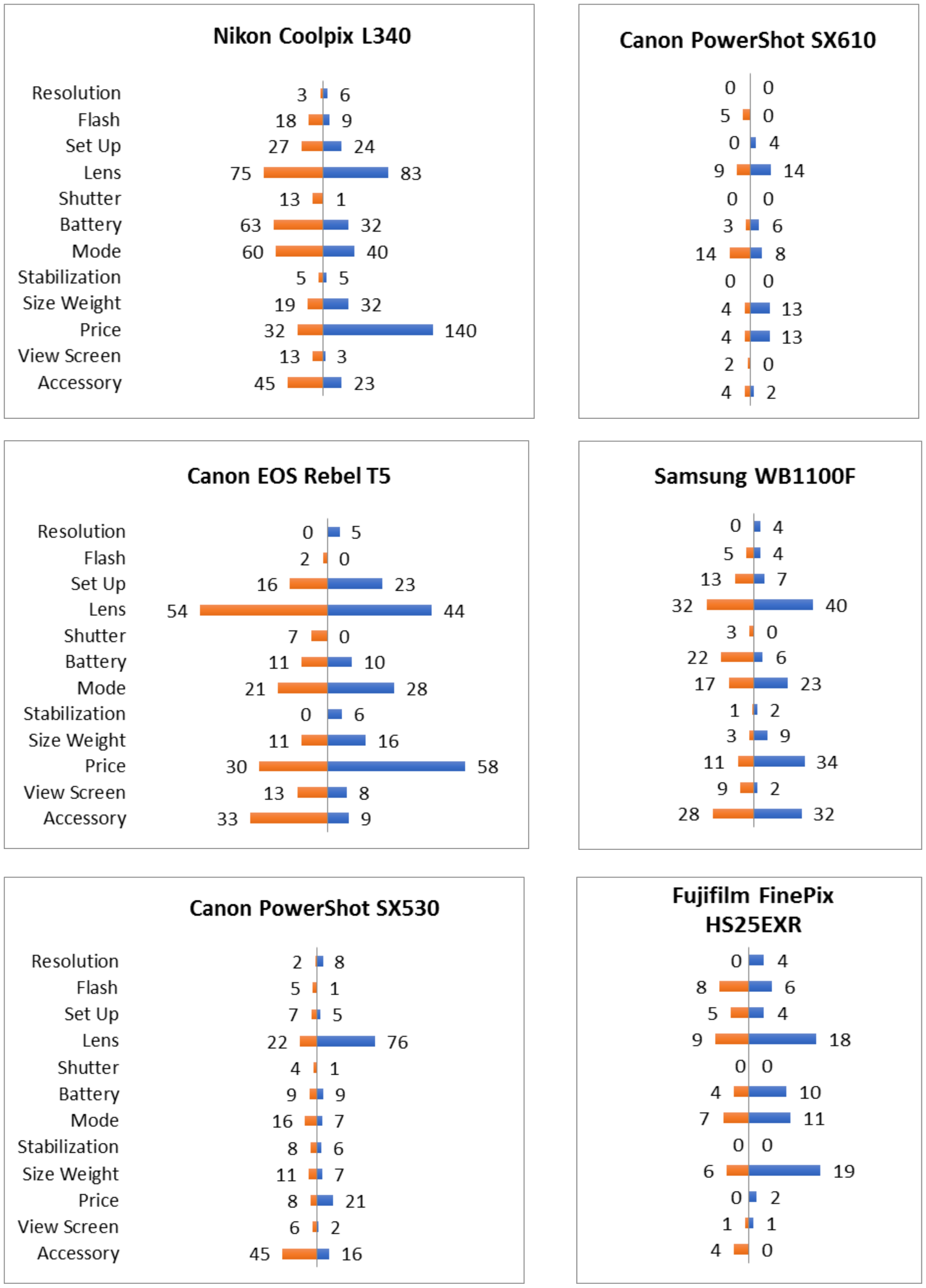


Figure 3: Review Summaries for the Six Selected Cameras

Note that this result is different from the one based on the ASR as shown in Table 4. According to the ASR, Cannon Rebel T5 has the highest ranking; however, when we investigated the reviews of this camera, we found that there were too many reviews having high star ratings but with the "Null" feature. Thus, its ASR does not truly reflect its actual product quality. As demonstrated in the review summary in Fig. 3, the camera Cannon Rebel T5 suffers from its major weaknesses such as low-quality lens and accessory, which gives a strong signal that a customer who cares about lens and accessory, shall stay away from this product.

7. Conclusions and Future Work

In this paper, we introduced a feature-based sentence model for evaluating and comparing online products based on their reviews. In our approach, we use the products' reviews as pieces of evidence to identify the strengths and weaknesses of products in terms of their product features. To illustrate the feasibility and effectiveness of our approach, we retrieved product information and review comments from Amazon. Our case study shows that our approach can achieve high accuracy with a sufficiently large training dataset. As a major benefit of our approach, customers can visually compare similar products based on their product features, and may greatly save their time on reading the large amount of product reviews.

In future research, we will seek effective ways to automatically extract product features from review comments using deep learning approaches such as Recurrent Neural Networks (RNN) [Graves et al. 2009][Erram 2017], and develop our own sentiment analysis tool to enhance system performance. We will also study the impacts on performance when considering a sentence as a sequence of related words using word embedding techniques such as the Word2vec model [Mikolov et al. 2013], rather than a bag of independent words. To create sufficient labeled data points to support our approach, we will develop useful tools to allow any user to help with the data labeling task. We will look into existing theories applied to examination of online reviews, including eWOM, information cascading theory and information economics theory [López & Sicilia 2014][Lee et al. 2015][Liu et al. 2016], and provide a solid theoretical background for our proposed methodology. Finally, since FSM, as a formal model, can be applied to other text-based data for text mining purpose, for example, to detect and filter spam emails by analyzing features of spam emails, we will investigate various online product classification methods [Kiang et al. 2011], and further validate the feasibility of our approach using labeled datasets from different domains.

Acknowledgment

This material is based upon work partially supported by the President's 2017 Creative Economy (CE) Fund, University of Massachusetts. We thank the editors and all referees for the careful review of this paper and the suggestions for improvements they provided.

REFERENCES

Akaichi, J., Z. Dhouioui and M. Perez, "Text Mining Facebook Status Updates for Sentiment Classification," *Proceedings of the 17th IEEE International Conference on System Theory, Control and Computing (ICSTCC)*, Sinaia, 640-645, 2013.

Blei, D., A. Ng. and M. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning Research*, Vol. 3: 993-1022, 2003.

Boiy, E., P. Hens, K. Deschacht and M. Moens, "Automatic Sentiment Analysis in Online Text," *Proceedings of the 11th International Conference on Electronic Publishing (ELPUB 2007)*, Vienna, 349-360, 2007.

Cao, Q., W. Duan and Q. Gan, "Exploring Determinants of Voting for the 'Helpfulness' of Online User Reviews: a Text Mining Approach," *Decision Support Systems*, Vol. 50, No. 2:511-521, 2011.

Deerwester, S., S. Dumais, G. Furnas, T. Landauer and R. Harshman, "Indexing by Latent Semantic Analysis," *Journal of the American Society for Information Science*, Vol. 41, No. 6:391-407, 1990.

DeGroof, R. and H. Xu, "Automatic Topic Discovery of Online Hospital Reviews Using an Improved LDA with Variational Gibbs Sampling," *Proceedings of the 2017 IEEE International Conference on Big Data (IEEE BigData 2017)*, Boston, MA, 3940-3947, 2017.

DeGroof, R., H. Xu, J. Zhang and R. Liu, "Mining Significant Terminologies in Online Social Media Using Parallelized LDA for the Promotion of Cultural Products," *Proceedings of the 14th International Conference on Data Science (ICDATA'18)*, Las Vegas, NV, 3-9, 2018.

Erram, V., "Automated Product Feature Extraction Using Recurrent Neural Networks," *Master's Thesis*, Computer and Information Science Department, University of Massachusetts Dartmouth, September 2017.

Esuli, A. and F. Sebastiani, "SentiWordNet: a Publicly Available Lexical Resource for Opinion Mining," *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-06)*, Genoa, 417-422, 2006.

Gelfand, A. "Gibbs Sampling," *Journal of the American Statistical Association*, Vol. 95, No. 452:1300-1304, 2000.

Graves, A., M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke and J. Schmidhuber, "A Novel Connectionist System for Improved Unconstrained Handwriting Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 31, No. 5:855-868, 2009.

Hasan, K., M. Sabuj and Z. Afrin, "Opinion Mining Using Naive Bayes," *Proceedings of the 2015 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE)*, Dhaka, 511-514, 2015.

Hofmann, T., "Probabilistic Latent Semantic Indexing," *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, 50-57, 1999.

Hu, H. W., Y. L. Chen and P. T. Hsu, "A Novel Approach to Rate and Summarized Online Reviews According to User-Specified Aspects," *Journal of Electronic Commerce Research*, Vol. 17, No. 2:132-152, 2016.

Jindal, N. and B. Liu, "Mining Comparative Sentences and Relations," *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-2006)*, Boston, MA, 1331-1336, 2006.

Jo, Y. and A. H. Oh, "Aspect and Sentiment Unification Model for Online Review Analysis," *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*, 815-824, Hong Kong, 2011.

Kamps, J., M. Marx, R. Mokken and M. De Rijke, "Using WordNet to Measure Semantic Orientations of Adjectives," *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC-04)*, 1115-1118, Libson, 2004.

Kiang, M. Y., Q. Ye, Y. Hao, M. Chen and Y. Li, "A Service-Oriented Analysis of Online Product Classification Methods," *Decision Support Systems*, Vol. 52, No. 1:28-39, 2011.

Lam, C., F. Lai, C. Wang, M. Lai, N. Hsu and M. Chung, "Text Mining of Journal Articles for Sleep Disorder Terminologies," *PLoS One*, Vol. 11, No. 5:e0156031, 2016.

Lacob, C. and R. Harrison, "Retrieving and Analyzing Mobile Apps Feature Requests from Online Reviews," *Proceedings of the 10th Working Conference on Mining Software Repositories*, San Francisco, CA, 41-44, May 2013.

Lee, Y.-J., K. Hosanagar and Y. Tan, "Do I Follow My Friends or the Crowd? Information Cascades in Online Movie Ratings," *Management Science*, Vol. 61, No. 9:2241-2258, 2015.

Liu, Q., S. Huang and L. Zhang, "The Influence of Information Cascades on Online Purchase Behaviors of Search and Experience Products," *Journal Electronic Commerce Research*, Vol. 16, No. 4:553-580, 2016.

López, M. and M. Sicilia, "Determinants of E-WOM Influence: The Role of Consumers' Internet Experience," *Journal of Theoretical and Applied Electronic Commerce Research*, Vol. 9, No. 1:28-43, 2014.

Manning, C., M. Surdeanu, J. Bauer, J. Finkel, S. Bethard and D. McClosky, "The Stanford CoreNLP Natural Language Processing Toolkit," *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Baltimore, MD, 55-60, 2014.

Manning, C., P. Raghavan and M. Schütze, *Introduction to Information Retrieval*, Cambridge University Press, 2008.

Mikolov, T., I. Sutskever, K. Chen, G. Corrado and J. Dean, "Distributed Representations of Words and Phrases and Their Compositionality," *Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS'13)*, Lake Tahoe, NV, 3111-3119, 2013.

Mudambi, S. M. and D. Schuff, "Research Note: What Makes a Helpful Online Review? A Study of Customer Reviews on Amazon.com," *MIS Quarterly*, Vol. 34, No. 1:185-200, 2010.

Nasukawa, T. and J. Yi, "Sentiment Analysis: Capturing Favorability Using Natural Language Processing," *Proceedings of the 2nd International Conference on Knowledge Capture (K-CAP-03)*, Sanibel Island, FL, 70-77, 2003.

Pang B. and L. Lee, "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts," *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics (ACL)*, Barcelona, 271-278, 2004.

Papadimitriou, C., P. Raghavan, H. Tamaki and S. Vempala, "Latent Semantic Indexing: a Probabilistic Analysis," *Journal of Computer and System Sciences*, Vol. 61, No. 2:217-235, 2000.

Papanikolaou, Y., T. Rubin and G. G. Tsoumakas, "Improving Gibbs Sampling Predictions on Unseen Data for Latent Dirichlet Allocation," *Technical Report*, Subjects: Statistics - Machine Learning, arXiv:1505.02065v2.

Pavlou, P. A. and A. Dimoka, "The Nature and Role of Feedback Text Comments in Online Marketplaces: Implications for Trust Building, Price Premiums, and Seller Differentiation," *Information Systems Research*, Vol. 17, No. 4:392-414, 2006.

Porteous, I., D. Newman, A. Ihler, A. Asuncion, P. Smyth and M. Welling, "Fast Collapsed Gibbs Sampling for Latent Dirichlet Allocation," *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Las Vegas, NV, 569-577, 2008.

Rigouste, L., O. Cappé and F. Yvon, "Inference and Evaluation of the Multinomial Mixture Model for Text Clustering," *Information Processing & Management*, Vol. 43 No. 5:1260-1280, 2007.

Socher, R., A. Perelygin, J. Wu, J. Chuang, C. Manning, A. Ng and C. Potts "Recursive Deep Models for Semantic Compositionality over a Sentiment Treebank," *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, Seattle, WA, 1631-1642, 2013.

Taddy, M., "Multinomial Inverse Regression for Text Analysis," *Journal of the American Statistical Association*, Vol. 108, No. 503:755-770, 2013.

Teh, Y. "A Bayesian Interpretation of Interpolated Kneser-Ney," *Technical Report TRA2/06*, NUS School of Computing, 2006.

Turney, P. "Thumbs up or Thumbs down?: Semantic Orientation Applied to Unsupervised Classification of Reviews," *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, Association for Computational Linguistics, Philadelphia, PA, 417-424, 2002.

Wei, R. and H. Xu, (2013) "A Formal Cost-Effectiveness Analysis Model for Product Evaluation in E-Commerce," *Proceedings of the 25th International Conference on Software Engineering and Knowledge Engineering (SEKE 2013)*, Boston, MA, 287-293, 2013.

Whitelaw, C., N. Garg, and S. Argamon, "Using Appraisal Groups for Sentiment Analysis," *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM)*, Bremen, 625-631, 2005.

Wiebe, J., R. Bruce and T. O'Hara, "Development and Use of a Gold-Standard Data Set for Subjectivity Classifications," *Proceedings of the 37th annual meeting of Association for Computational Linguistics (ACL-1999)*, Stroudsburg, PA, 246-253, 1999.

Wikarsa, L. and S. Thahir, "A Text Mining Application of Emotion Classifications of Twitter's Users Using Naïve Bayes Method," *Proceedings of the 2015 1st IEEE International Conference on Wireless and Telematics (ICWT)*, Manado, 77-83, 2015.

Yang, M., M. Kiang, H. Chen and Y. Li, "Artificial Immune System for Illicit Content Identification in Social Media," *Journal of the American Society for Information Science and Technology*, Vol. 63, No. 2:256-269, 2012.

Ye, Q., R. Law and B. Gu, "The Impact of Online User Reviews on Hotel Room Sales," *International Journal of Hospitality Management*, Vol. 28, No.1:180-182, 2009.

Yin, D., S. Bond and H. Zhang, "Anxious or Angry? Effects of Discrete Emotions on the Perceived Helpfulness of Online Reviews," *MIS Quarterly*, Vol. 38, No. 2:539-560, 2014.

Zhang, R. and T. Tran, "Helpful or Unhelpful: A Linear Approach for Ranking Product Reviews," *Journal of Electronic Commerce Research*, Vol. 11, No. 3:220-230, 2010.

Zhang, Y. and H. Xu, "SLTM: a Sentence Level Topic Model for Analysis of Online Reviews," *Proceedings of the 28th International Conference on Software Engineering and Knowledge Engineering (SEKE 2016)*, Redwood City, CA, 449-453, 2016.

Zhang, Z. and B. Varadarajan, "Utility Scoring of Product Reviews," *Proceedings of the 15th ACM International Conference on Information and Knowledge Management (CIKM)*, Arlington, VA, 51-57, 2006.