Generative Feature Language Models for Mining Implicit Features from Customer Reviews

Shubhra Kanti Karmaker Santu
Univeristy of Illinois at Urbana-Champaign
karmake2@illinois.edu

Parikshit Sondhi
WalmartLabs
psondhi@walmartlabs.com

ChengXiang Zhai
Univeristy of Illinois at Urbana-Champaign
czhai@illinois.edu

ABSTRACT

Online customer reviews are very useful for both helping consumers make buying decisions on products or services and providing business intelligence. However, it is a challenge for people to manually digest all the opinions buried in large amounts of review data, raising the need for automatic opinion summarization and analysis. One fundamental challenge in automatic opinion summarization and analysis is to mine implicit features, i.e., recognizing the features implicitly mentioned (referred to) in a review sentence. Existing approaches require many ad hoc manual parameter tuning, and are thus hard to optimize or generalize; their evaluation has only been done with Chinese review data. In this paper, we propose a new approach based on generative feature language models that can mine the implicit features more effectively through unsupervised statistical learning. The parameters are optimized automatically using an Expectation-Maximization algorithm. We also created eight new data sets to facilitate evaluation of this task in English. Experimental results show that our proposed approach is very effective for assigning features to sentences that do not explicitly mention the features, and outperforms the existing algorithms by a large margin.

1. INTRODUCTION

Online customer reviews are a great resource for both shoppers and business owners to share feedback about products and services. Shoppers often read them prior to making purchase decisions, while business owners use them to obtain valuable business intelligence regarding customer satisfaction. However the sheer volume and primarily unstructured nature of online reviews hinder users' ability to obtain a comprehensive understanding of detailed opinions about specific features/aspects of a product or service.

To address this problem, researchers have studied approaches for automatically summarizing/analyzing opinions buried in the review data [8, 9, 10, 14]. A fundamental requirement for such approaches is to recognize opinions at the feature level. This requires a) identification of key product features, and b) association of each opinion expressing sentence with one or more identified features. The first task is relatively simple, since we can directly extract key product features from its specification (spec) document which is typically a structured list of attribute-value pairs provided by the seller (see Figure 1).

The second task of associating review sentences with product features they mention is more challenging. We refer to a sentence that mentions a product feature as a 'feature sentence'. Clearly not all review sentences are feature sentences. Prior work has classified feature sentences as having either *explicit* or *implicit* feature mentions [7, 15, 20]. The difference between the two is subtle, but can be clarified through an example.

Consider the following sentence, where the feature 'size' is mentioned explicitly in the following sentence.

"I like the size of the phone, it's really small."

An explicit feature mention, by definition, can be detected by simply performing a boolean check on whether the feature name, or its synonyms, or certain informative keywords appear in a sentence. For the feature 'size' these may include keywords like 'size', 'dimensions', 'length', 'width' etc.

Consequently prior approaches have focused on identifying explicit mentions by extracting such informative keywords for each feature of interest, and using them for tagging mentions in sentences [11, 13, 15]. This strategy however does not work for the relatively unexplored problem of identifying implicit feature mentions in sentences which contain no informative keywords. For example the following sentence also implicitly mentions 'size', but one cannot detect that by boolean checking individual keywords.

"The phone fits nicely into any pocket without falling out."

Our manually tagged dataset shows that such implicit mentions cover a substantial portion ($\sim 19\%$) of feature sentences. Their accurate identification therefore is critical to ensuring high recall of all automated review analysis applications. Due to its importance, the implicit feature extraction problem has recently started attracting interest, predominantly with Chinese review texts [16, 5, 19, 17]. However, current methods tend to rely on heuristically designed measures such as correlation counts, association rules etc. and require significant manual tuning of parameters, making them hard to generalize.

In this paper, we propose a new probabilistic method for identification of implicit feature mentions. Our main idea is to model the review data with a generative probabilistic model, representing sentence-feature associations via hidden

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CIKM'16 , October 24 - 28, 2016, Indianapolis, IN, USA

© 2016 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ISBN 978-1-4503-4073-1/16/10...$15.00

DOI: http://dx.doi.org/10.1145/2983323.2983729

Technical Details Collapse all

Summary

Screen Size	13.3 inches
Screen Resolution	1280 x 800
Max Screen Resolution	1280x800 pixels
Processor	2.3 GHz Intel Core i5
RAM	4 GB SDRAM
Memory Speed	1333 MHz
Hard Drive	320 GB Serial ATA
Graphics Coprocessor	Intel HD Graphics 3000
Chipset Brand	Intel
Card Description	Graphics Media Accelerator 3000
Wireless Type	802.11A
Number of USB 2.0 Ports	2
Average Battery Life (in hours)	7 hours

Figure 1: Sample product specification of a Laptop

variables. We further assume that a subset of explicit feature mentions have already been tagged and made available as training data. These are then used to estimate the model parameters via an iterative EM-style approach. Eventually, the inferred values of model parameters and hidden variables provide the implicit mentions[1].

To evaluate the proposed method, we created eight standardized data sets for English language reviews. Five by extending existing datasets from Hu *et. al.* [7] by manually tagging all feature mentions, and three by extending datasets from [12] through an automated tagging approach (details in section 5.1). We then compared our proposed approach with two baseline algorithms based on current state-of-the-art approaches. Results show that our proposed method outperforms the baselines by a significant margin in both extraction accuracy and robustness.

2. RELATED WORKS

Sentiment analysis/opinion mining in online user reviews has been studied largely in the past decade [3, 7, 11, 14, 18]. Most of the existing researches focus on finding explicit feature mentions [11, 13, 15]; our work is different from them because we exclusively focus on the identification of implicit mentions of features hidden inside the review sentences of online customer reviews, which can potentially improve all the opinion analysis and mining applications.

A few attempts have been made to extract implicit feature mentions from online customer reviews [16, 5, 19, 17]. One limitation is that all these works have used only Chinese review text as the experimental dataset; in this paper, we construct new English test collections to enable evaluation on English text data. Previous approaches used different heuristics for extraction of implicit feature mentions. For example, one line of work primarily looks at the correlation between opinion words and feature words and infer implicit feature mentions based on the occurrence of highly correlated words. Hai *et.al.* [5] proposed a two-phase co-occurrence association rule mining approach to identify implicit feature mentions. Their rules contained opinion words in the rule antecedent and explicit feature mentions as the rule consequent. In the first phase of rule generation, a set of association rules of the form [opinion-word, explicit-feature] is generated by looking at the co-occurrence matrix. In the second phase, consequents (explicit feature mentions) were clustered to generate more robust rules for each opinion word. One particular limitation of their approach is that they only considered the associations between feature words and opinion words, but ignored the associations between feature words and the rest of the factual/notional words in the clause. But, factual/notional words often provide important clue about the implicit feature mentions [17, 19].

To address this limitation, Wei *et.al.* [17] extended the works of [5] by considering factual words along with the opinion words and forming association rules by five different collocation extraction algorithms. With similar motivation, Yu *et.al* proposed a co-occurrence association-based method [19], where they also try to consider any factual-word in addition to the opinion words to identify implicit feature mentions. They tried to exploit co-occurrence count of factual-word and explicit-feature-mention pairs to imply the presence of implicit feature mentions. However, the set of factual words considered in [19] was pre-selected and limited; also no detail was outlined about how to extract these factual words. One major limitation of the works [17] and [19] is that they do not provide any principled way to filter out the large amount of noisy words present in natural language texts. They mostly handled this problem by relying on heuristically tuned parameters and thresholds which lacks enough theoretical ground to justify the correctness. There are many decision choices (such as parameters, thresholds) that have to be made in a somewhat arbitrary or empirical manner. These decision choices hardly generalize over different datasets. The authors of [17] themselves pointed out this limitation and urged for future research to solve this problem.

Another line of work exploits the unsupervised learning technique to extract implicit feature mentions. For example, Su *et.al.* developed a mutual reinforcement approach which clusters product features and opinion words simultaneously and iteratively by exploiting the hidden sentiment association between product feature category and opinion word group. The learned association was then further used to determine implicit feature mentions [16]. However, one limitation of their approach is that they only considered the opinion words discarding all other notional/factual words similar to what was done in [5]. On top of that, their algorithm assumes that the set of opinion words is pre-determined and supplied beforehand. Thus, the success of their approach largely depends on this pre-supplied restricted opinion word set and will hardly generalize well over different datasets.

In contrast to all the existing approaches which suffer from many ad hoc manual tuning or configuration of components, our approach exploits statistical modeling and machine learning to frame the problem as a more principled optimization problem. As a result, our approach does not pose any restriction on the set of words (opinion or notional) we consider and can learn the model parameters automatically without manual or heuristic tuning. Noisy words can be naturally handled in our approach by introducing a background model.

3. PROBLEM FORMULATION

Our problem setup assumes that for a given product, we are provided with a collection of reviews, and it's associ-

[1]In principle our approach could be used to label both unknown explicit and implicit feature mentions. But in this work we focus only on detecting implicit mentions, since approaches for identifying explicit mentions already exist.

ated features of interest. In addition, review sentences with explicit feature mentions have already been labeled using a traditional dictionary lookup based approach. Our goal is to label all remaining implicit feature mentions in the dataset.

Formally, let $R = \{r_1, r_2....r_N\}$ be a collection of reviews of a product or a service, where r_j is a review. Let s_{ij} be the ith sentence in the jth review $r_j \in R$. Let $F = \{f_1, f_2, f_3, ..., f_M\}$ be the set of known product features. For example, for cell phones, f_i may be "size" or "battery life". Finally $A_{ijk} \in \{1, 0\}$ are binary variables which take a value 1 if s_{ij} mentions feature f_k (either explicitly or implicitly) and 0 otherwise.

The general problem of tagging sentences with feature mentions involves assigning a binary value to all possible A_{ijk}. Since explicit feature mentions are provided, a subset of A_{ijk} entries are already known to be 1. The problem of tagging implicit feature mentions is to fill up all the remaining entries.

Input: Review set R, feature set F, a subset of A_{ijk} entries already known to be 1

Output: 0 or 1 values assigned to the remaining A_{ijk} entires.

With our problem formulation, the problem is in nature similar to traditional multi-label categorization problem, where each instance (i.e. sentence) can be assigned multiple labels (i.e. product features). In practice product review sentences don't usually mention more than $2-3$ features. Such problems are normally solved using a supervised machine learning setup. In our case the available explicit mentions serve as training data and the implicit mentions serve as test instances. To this end, we use a typical supervised learning method (Naive Bayes) as one of our baselines. Our precise approach to identify explicit feature mentions and convert them into training instances is discussed in the next section.

Our formulation also has similarities to traditional topic modeling approaches in that we try to probabilistically assign multiple topic labels to each sentence. However in our setting, topics i.e. product features are fixed and already known. Traditional topic models such as PLSA [6] or LDA [1] do not guarantee that their automatically discovered topics will align with our predefined set of product features, making them unsuitable for the task.

Note that both our problem setup and proposed generative model are agnostic to whether the known A_{ijk} entries (training data) provided to the method are in fact explicit feature mentions. In principle any subset of known mentions whether explicit or implicit may be provided, and the method is expected to fill up the remaining. The partitioning of feature mentions as explicit or implicit is a characteristic of the domain. Since relatively accurate methods for identifying explicit feature mentions already exist, it is easy to obtain A_{ijk} entries that correspond to explicit feature mentions as training data. Implicit feature mentions on the other hand are harder to obtain, and hence will typically form the test set.

4. PROPOSED APPROACH

In this section, we describe the proposed approach in detail. Our approach is based on a generative model of review text, so we will first describe this model, then describe how to estimate parameters, and finally discuss how to infer the implicit features using the model.

4.1 A Generative Feature Language Model

The key idea of the proposed approach is to model the vocabulary occurring in sentences describing some feature using a unigram language model (called a feature language model), or a word distribution, denoted by γ_1, ..., γ_k to represent k features. For example, the unigram language model for a feature such as "size" may assign high probabilities to words that we expect to see frequently in sentences discussing the "size" feature (e.g., "pocket", "small" etc), while the model for the feature "customer service" can be expected to assign small probabilities to a word like "small", but high probabilities to other words more associated with "customer service" such as "call", "friendly", or "representative". Since each feature has its own feature language model, we will have as many feature language models as the number of features, each modeling the words we expect to see in sentences discussing the corresponding feature.

Suppose we can estimate an accurate feature language model for every feature, it would be easy to solve the problem of extracting implicit features as follows. We can compute the probability of a candidate sentence using each feature language model, and tag the sentence with features whose feature language models give the sentence a sufficiently large probability. Essentially, we can treat each feature as a category and use the Naive Bayes classifier to classify a sentence into one or more categories.

However, how can we estimate an accurate feature language model for every feature? If we had labeled data that tell us which sentences mention which (implicit) feature, we would have been able to estimate the corresponding feature language model easily based on the observed word frequencies in the sentences mentioning the same implicit feature. Unfortunately, we do not assume that we have such training data which require manual labor to create.

To solve this problem, we can fit a mixture model with these feature language models as components to all the review text data we have and learn these feature language models in an unsupervised way similar to how probabilistic topic models such as PLSA [6] or LDA [1] can be used to learn the latent topics from text data. The basic idea here is to search for a specific set of concrete feature language models that can give the observed review data the maximum probability, or use the Maximum Likelihood estimator to estimate all the parameters that specify the probabilities in each feature language model.

However, if we are to use PLSA or LDA directly to model our review data, we would not be able to ensure that a learned topic corresponds to a feature. To solve this problem, we pre-specify the word distributions in the mixture model by estimating them based on sentences explicitly mentioning phrases describing a feature so that each would correspond to a feature; as a result, our mixture model would only have the mixing weights as parameters to be estimated, and once these parameters are estimated, they can be directly used to assign features to a sentence.

To model the noisy words in the review sentences, we also introduce a special language model called background language model γ_B, which can be estimated based on the entire collection of review data so that it would assign high probabilities to frequent words like "the", "a", and "we" and thus help explain/attract such noisy words when used in a mixture model together with feature language models.

Our overall idea is thus to assume that our review data

are generated using a generative mixture model with feature language models as components (which we would refer to as a **Generative Feature Language Model** (GFLM)) according to the following process as illustrated in Figure 2:

1. Each sentence is generated by generating each of the words in the sentence independently.

2. To generate a word w in sentence S, we first decide whether we would generate the word using the background model γ_B or a feature language model γ_i. We make this choice according to $\lambda_B \in [0, 1]$, which is a parameter indicating the probability of using the background model instead of a feature language model. Thus the probability of choosing a feature language model would be $1 - \lambda_B$.

3. If we have chosen the background language model, we would sample the word from the distribution $p(w|\gamma_B)$; otherwise, we would further make a decision on which of the k feature language models to use, and this decision is made based on another set of parameters $\{\pi_{S,i}\}$, where $i = 1, ..., k$, and $\sum_{i=1}^{k} \pi_i = 1$. $\pi_{S,i}$ is the probability of choosing feature language model γ_i to generate the word. Thus with probability $\pi_{S,i}$, we would sample the word using $p(w|\gamma_i)$.

4. This process would be repeated to generate all the words in a sentence, and all the sentences would be generated in the same way, each sentence being generated using a set of sentence-specific topic choice parameters $\pi_{S,i}$.

The design of such a generative model is based on two reasonable assumptions: First, each sentence may contain words drawn from multiple feature language models; this is reasonable since a sentence may mention multiple features. Second, each sentence has its own feature choice parameters (π); this not only reflects the real situation (i.e., different sentences tend to mention different features), but also allows us to infer the features implicitly mentioned in the sentence based on its feature choice parameter values. Specifically, $\pi_{S,i}$ indicates to what extent sentence S can be explained by the feature language model γ_i, thus it would be reasonable to assume that if we are to assign a feature to sentence S, we should choose the feature that has the maximum $\pi_{S,i}$. In our experiments, we used a threshold θ to determine whether we should assign a feature to a sentence and assign all features whose π parameter values are above θ.

According to our generative model, the probability of observing a word w in a review sentence S is:

$$P_S(w) = \lambda_B P(w|\gamma_B) + (1 - \lambda_B) \sum_{i=1}^{k} \pi_{S,i} P(w|\gamma_i). \quad (1)$$

The log-likelihood of observing the entire set of reviews R from a mixture model with unknown parameters Λ is thus:

$$\log P(R|\Lambda) = \sum_{S \in R} \sum_{w \in V} [c(w, S) \times \log\{\lambda_B P(w|\gamma_B) \quad (2)$$
$$+ (1 - \lambda_B) \sum_{i=1}^{k} (\pi_{S,i} P(w|\gamma_i))\}]$$

where V is the set of words in our vocabulary, $c(w, S)$ is the number of times word w appears in sentence S, and the parameter set Λ is defined as $\Lambda = \{\pi_{S,i} | S \in R, 1 \leq i \leq k\}$. Note that we assume that all the parameters of the component language models, i.e., γ_1, ..., γ_k, and γ_B, are known because they would be estimated by using the pseudo training data obtained by finding sentences *explicitly* mentioning keywords describing features as we will discuss below.

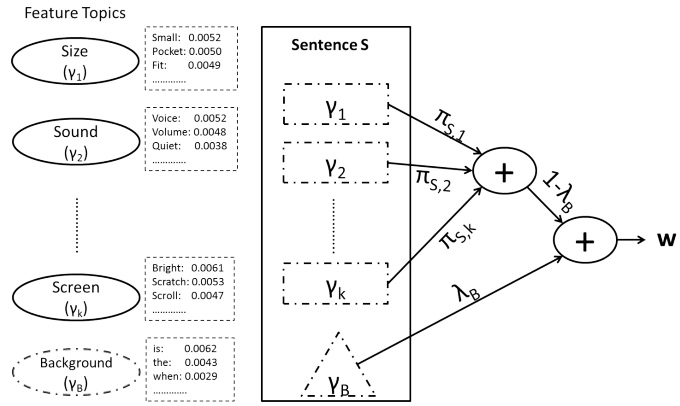


Figure 2: Schematic diagram of the generative model for a hypothetical "Cell Phone" Dataset

We can then estimate the parameters Λ using the Maximum Likelihood estimator which finds a setting of Λ that maximizes the log-likelihood:

$$\hat{\Lambda} = \arg\max_{\Lambda} \log P(R|\Lambda) \quad (3)$$

This optimization problem can be solved by using an Expectation-Maximization (EM) algorithm which will be described in the next section.

4.2 Parameter Estimation:

Equation 1 reveals that our generative model incorporate four different parameters, i.e., $P(w|\gamma_B)$, $P(w|\gamma_i)$, λ_B and $\pi_{S,i}$. We outline there estimation process in the following paragraphs sequentially.

First, all the unigram language models, including the background model γ_B and the feature language models γ_i, would be estimated based on pseudo training data and kept as constants when computing the maximum likelihood estimate using Equation 3. This is, in effect, to impose an infinitely strong prior on these parameters of the generative model from the perspective of Bayesian estimation.

$P(w|\gamma_B)$**:** This is estimated using the whole collection R as $p(w|\gamma_B) = \frac{\sum_{S \in R} c(w,S)}{\sum_{w' \in V} \sum_{S \in R} c(w',S)}$. Note that, $p(w|\gamma_B)$ is constant throughout the learning process and doesn't change during the later estimation procedure.

$P(w|\gamma_i)$**:** As mentioned in section 3, we assume that we already know the set F of product features we are interested in. Thus, we can automatically tag each sentence in the review corpus with the explicitly mentioned features in that sentence to generate some "labeled data", which can be used to estimate the feature topic models, i.e., $P(w|\gamma_i)$. Specifically, $P(w|\gamma_i)$ is estimated as follows.

1. For each feature $f_i \in F$ and each word $w \in V$, we count the number of times f_i and w occur explicitly in the same sentence. Let this count be $C(f_i, w)$.

2. For each word $w \in V$, we count the number of review sentences where word w appears at least once. Let this count be $C(w)$.

3. For each feature $f_i \in F$ and each word $w \in V$, we compute their TF-IDF weight by the following formula:

$$TFIDF(f_i, w) = \log\{1 + C(f_i, w)\} \times \log\left\{1 + \frac{N}{C(w)}\right\} \tag{4}$$

N is the total # of sentences in the review corpus.

4. Finally, we compute the feature language models by normalizing the $TFIDF$ weights as shown in the following formula:

$$P(w|\gamma_i) = \frac{TFIDF(f_i, w) + 1}{\sum_{w' \in V} TFIDF(f_i, w') + |V|} \tag{5}$$

Note that, we used Laplacian smoothing to avoid zero counts for unseen words in the review corpus.

The rationale for this estimation method is to give words that have strong associations with a feature f_i higher probabilities. The use of TFIDF weighting is to penalize those common words which may occur frequently in all the sentences for all the features.

λ_B encodes our belief about the level of noise in the review data and can thus be set to any reasonable value reflecting the percentage of words to be treated as noise instead of requiring cross-validation to empirically tune the parameter. In our experiments, we set $\lambda_B = 0.6$ which means that we assume, on average, 60% of the words can be treated as noise; this setting is intuitively reasonable for our problem setup and has worked well empirically.

The parameters to be estimated now are $\Lambda = \{\pi_{S,i} | S \in C, 1 \leq i \leq k\}$, which can be estimated using the Expectation Maximization (EM) algorithm [2] based on Equation 3. The EM algorithm is a general algorithm for maximum-likelihood estimation where the data are incomplete (implicit features) or the likelihood function involves latent variables (z). Informally, the EM algorithm starts with randomly assigning values to all the parameters to be estimated, i.e., Λ. It then iteratively alternates between two steps, called the expectation step (i.e., the E-step) and the maximization step (i.e., the M-step), respectively. In the E-step, it computes the expected likelihood for the complete data (the so-called Q-function) where the expectation is taken w.r.t. the computed conditional distribution of the latent variables (i.e., the hidden variables) given the current settings of parameters and our observed (review text) data. In the M-step, it re-estimates all the parameters by maximizing the Q-function. Once we have a new generation of parameter values, we can repeat the E-step and another M-step. This process continues until the likelihood converges, i.e., reaching a local maxima.

We can use the following iterative updating formulas to estimate all the parameters. Intuitively, if we know the identity of each word in the collection, i.e., which feature topic (or background) model generated a particular word, then it is quite easy to estimate Λ. We thus introduce a hidden variable for the identity of each word, $\{z_{S,w}\}$. $P(z_{S,w} = f)$ indicates the probability that word w in sentence S is generated from feature topic γ_f given that w is not generated from the background model γ_B. Note that, $\sum_{f' \in F} P(z_{S,w} = f') = 1$. $P(z_{S,w} = B)$ is the probability that word w in sentence S is generated from the background model . Here we changed the notation to use variable f, instead of i which we used before, as the feature index variable to make it easier to interpret the EM algorithm equations.

We can write down the E-step and M-step as follows:

E-Step:

$$P(z_{S,w} = f) = \frac{\pi^{(n)}_{S,f} P(w|\gamma_f)}{\sum_{f'=1}^{k} \pi^{(n)}_{S,f'} P(w|\gamma_{f'})} \tag{6}$$

$$P(z_{S,w} = B) = \frac{\lambda_B P(w|\gamma_B)}{\lambda_B P(w|\gamma_B) + (1 - \lambda_B) \sum_{f'=1}^{k} \pi^{(n)}_{S,f'} P(w|\gamma_{f'})} \tag{7}$$

M-Step:

$$\pi^{(n+1)}_{S,f} = \frac{\sum_{w \in V} c(w, S)(1 - P(z_{S,w} = B))P(z_{S,w} = f)}{\sum_{f'=1}^{k} \sum_{w \in V} c(w, S)(1 - P(z_{S,w} = B))P(z_{S,w} = f')} \tag{8}$$

In E-Step, we are actually estimating the distribution of the hidden variables, (or estimating the identity of each word). This distribution is simply computed by the likelihood of a word, how much proportion is contributed by the background model, or by feature topic f if not contributed by the background. Thus, each word is assumed to be divided into fractions and the fractions are assumed to be generated from different feature topics.

M-step is essentially aggregating such fractions to estimate a new set of values for Λ. To estimate the new $\pi_{S,f}$ (the mixing weights for a sentence), we just aggregate all the fractions of words generated by feature topic f in sentence S, and normalize $\{\pi_{S;f}\}_{f=1\ldots k}$ to make $\sum_{f'=1\ldots k}\{\pi_{S;f'}\} = 1$ (Equation 8).

Before convergence, each iteration of the EM algorithm will yield a larger likelihood value in Equation 2. The algorithm will terminate when it achieves a local maximum of the log likelihood. In our experiments, we used multiple trials with random initializations to improve the local maximum we obtain, where in each trial we begin with a new starting point of Λ (by assigning random values to Λ).

To give an intuitive example of how the EM framework actually works, lets consider the generation of a word "pocket" in the hypothetical "Cell Phone" dataset with reference to Figure 2. Given the current parameter values, the E-Step essentially computes what is the probability that the background model generated word "pocket" and if not generated by the background model, what fraction of it is generated from which feature topics, e.g., "size", "sound", "screen" etc. For word "pocket", the fraction for feature "size" is likely to be high as $P(\text{"}pocket\text{"}|\text{"}size\text{"})$ is likely to be higher compared to $P(\text{"}pocket\text{"}|\text{"}sound\text{"})$ or $P(\text{"}pocket\text{"}|\text{"}screen\text{"})$.

To illustrate how M-step works, let us consider the following review sentence which talks about the feature "screen" implicitly:

"There were scratches on the display. Its very annoying."

M-step here re-estimates the proportion of different feature topics, e.g., "size", "sound", "screen" etc., in the generation of this sentence. To give an intuitive hypothetical simulation, for the words "scratches" and "display", the fraction for feature "screen" is most likely to be larger than the features "size" and "sound". For the rest of the words, the distribution of the three features is likely to be somewhat uniform as these are mostly background words. Thus, gathering these fractions and aggregating them would cause feature "screen" to achieve a higher proportion than the other two features in the generation process of the sentence.

4.3 Implicit Feature Prediction:

After the EM algorithm converged, we know the identities of each word $P(z_{S,w} = f)$ and $P(z_{S,w} = B)$, i.e., the degree to which the background model or some feature topic model contributed to the generation of a particular word. We also know the feature topic distributions $\pi_{S,f}$, i.e., to what proportion, a particular sentence S is generated from some feature topic f.

Based on these quantities, we can infer the implicit feature mentions within various sentences in two different ways, which would be called **GFLM-Word** and **GFLM-Sentence**, respectively.

In case of "GFLM-Word", given a sentence S, it looks at each word w and adds a feature f to the inferred implicit feature list $X(S)$ if and only $p(z_{S,w} = f) \times (1 - p(z_{S,w} = B))$ is greater than some threshold θ for at least one word in S. The philosophy behind this formula is that if any particular word w has a small probability of being generated by a background model but has higher probability of being generated from some feature topic model γ_f, then word w is likely referring to feature f implicitly. Here, the decision is made solely by looking at individual words, not the entire sentence.

In case of "GFLM-Sentence", given a sentence S, it looks at the contribution of each feature f in the generation of the sentence, i.e.,$\pi_{S,f}$ and infers f^* as an implicit feature only if π_{S,f^*} is greater than some threshold θ. Here, the decision is made at the sentence level, not at individual word level, thus may be more robust which is indeed confirmed in our experiments.

5. EXPERIMENT DESIGN

5.1 Dataset

To the best of our knowledge, no standardized English language datasets exist for evaluating methods that mine implicit feature mentions from online product reviews. To solve this problem, we created eight new data sets which will be made available for researchers to use, including both human annotated data sets and automatically annotated datasets using a computational method.

5.1.1 Human Annotated Dataset

Our gold standard consists of assignments of features to sentences. To leverage existing resources to create such a data set, we used the review data made available by [7] as our starting point. The collection contained five different datasets associated with five different electronic products. In total, there were 314 reviews containing 4259 sentences. Each sentence was tagged with the features that were explicitly present in that particular sentence. These explicit feature mentions were tagged by the authors of [7]. See Table 1 for more details.

This data set already has review sentences and features extracted, but it does not have a comprehensive tagging of implicit feature mentions, which we need for our evaluation. Our main task is thus to create such tags of implicit feature mentions (i.e., tag a sentence with features that are implicitly/indirectly mentioned in the sentence). We recruited two (non-author) volunteers to perform this task. Each volunteer was provided with the list of product features and asked to review all 4259 sentences. For each sentence they were requested to specify all product features that were being referenced implicitly in the sentence. The tagging results of the two annotators were merged based on unanimous voting so that only the features assigned by both volunteers would be used as the final tags; tags with disagreements were discarded. Detailed information about tags for each product is provided in Table 1. We obtained 687 total tags from volunteer one and 703 tags from volunteer two. Out of these, the two agreed on 649 tags spanning a total of 569 sentences. When a sentence explicitly mentions a feature, the feature is automatically assigned to the sentence, so the human annotators did not tag any explicit feature mentions, rather they focused on assigning implicit feature mentions.

Ideally, we would like our data set to be as large as possible, but the size of our data set was inevitably limited by the manual effort involved in tagging. Nevertheless as we show in section 6, our data set is sufficiently large to allow us to observe statistically significant improvements over baselines.

5.1.2 Automatically Annotated Datasets

In order to construct larger data sets without incurring too much manual work, we used an automated tagging approach. We created three new data-sets through this automated technique. The idea is to delete an explicitly mentioned feature from a sentence and see if our algorithm can "recover" it.

Specifically, we collected a comparatively larger set of $4,699$ reviews containing $38,056$ sentences from the dataset provided by [12]. The reviews were associated with three different products in the "Electronics" genre. We also obtained the relevant lists of product features from the amazon product specification web-page, i.e., for each particular product, we searched it on amazon to find the product specification page and collected the list of features mentioned there. We also ignored features with low frequency of occurrence, specifically, only the features that are mentioned in at least 30 sentences are used for the simulation evaluation.

For each feature, we randomly selected $\sim 40\%$ of its explicit mentions and deleted them from their respective sentences, thus artificially creating sentences that do not mention the feature explicitly, but are known to refer to the feature. We can then assume the deleted feature should be assigned to such an artificially created sentence from which the corresponding feature-phrase has been deleted.

For example, we can remove "battery" from the following sentence

"The battery holds charge for the whole day, you don't need to carry a charger with you.".

to obtain

"The holds charge for the whole day, you don't need to carry a charger with you.".

and will assume that this ungrammatical sentence should be assigned the feature "battery". Since our approach is based on bag of words representation, the fact that the new sentence is no longer grammatical is not a concern. See Table 2 for specific statistics about these datasets.

5.2 Baseline Approaches

The main research questions we want to answer in our experiments are: 1) How effective are our proposed methods for exploiting the unlabeled data, which is the main motivation of the proposed new methods? 2) How are the proposed new methods compared with the current state of the

Table 1: Dataset Description of five datasets extended from the work [7] and annotated by humans.

Dataset	Total no of reviews	Total no of sentences	No of Explicit features	Sentences with at least one implicit feature	Total no of occurrences of Implicit Features
Cellular phone1	41	587	67	80	91
DVD player	99	839	49	129	141
Mp3 player1	95	1811	57	226	271
Digital camera1	45	642	96	74	80
Digital camera2	34	380	79	60	66
Total	314	4259	348	569	649

Table 2: Dataset Description of three new large datasets created from raw data in [12] and annotated automatically.

Dataset	Total no of reviews	Total no of sentences	No of Explicit features selected	Total Explicit feature occurrence	Implicit features (Deleted explicit occurrence)	percentage of deletion	Sentences with at least one implicit feature
Cellular phone2	966	9,856	14	2,825	1,123	39.75%	1057
Mp3 player2	1495	10,347	10	1,596	633	39.66%	607
Router	2238	17,853	10	3,193	1,275	39.93%	1214
Total	4,699	38,056	34	7,614	3,031	39.78%	2878

art method? To answer the first question, we compare our methods with a baseline method representing a representative supervised learning method (i.e., a naive bayes classifier), which is known to perform well for many text classification tasks. To answer the second question, we compare our methods with another baseline method representing the current state of the art, i.e., the approach of correlation counts between feature words and opinion/factual words proposed in [19] for Chinese review data.

5.2.1 Naive Bayes Classifier Based Ranking

This approach uses all the sentences explicitly mentioning a feature as the (positive) training examples for the corresponding feature so that we can train a classifier to use the words in such sentences as clues to classify additional sentences that may not mention the features explicitly into one or more of these features. Specifically, the ranking function takes the words in a review sentence S as its input and generates a score for each of the candidate implicit feature f' which indicates how much likely it is that f' has actually been mentioned implicitly. The score is computed based on the following formula:

$$score(f'|S) = P(w_1|f') \times P(w_2|f') \times .... \times P(w_n|f') \times P(f') \\ = P(f') \times \Pi_{i=1}^{n} P(w_i|f')$$

Here, $S = \{w_1, w_2, ..., w_n\}$. Some threshold θ is chosen such that, for a candidate feature f', if $score(f'|S) > \theta$, f' would be assigned to sentence S.

5.2.2 Correlation Based Ranking

Given the same setup discussed in Section 5.2.1, the ranking function here is a sum of correlation measurements between context words (w_i's) and candidate feature (f') as shown in the following Equation:

$$score(f'|S) = \frac{\sum_{i=1}^{n} \frac{M(f', w_i)}{count(w_i)}}{n}$$

Here, $M(f', w_i)$ is the number of times both f' and w_i appear together in a sentence and $count(w_i)$ is the number of times word w_i is mentioned in the entire corpus. Finally, some threshold θ is chosen such that, for a candidate feature f', if $score(f'|S) > \theta$, f' would be assigned to sentence S.

5.3 Performance Measures

To measure the performance of the proposed approach, we use three popular measures available in the literature: Precision, Recall and the F1 measure [4]. Following is how these measures were computed: For each review sentence, the implicit features assigned by our approach (controlled by the thresholding parameter) were compared against the list of "gold" implicit features to compute the true positive, false positive and false negative statistics for that sentence. Note that, explicit features mentions were not considered in any way during the computation of these statistics. Then, all such statistics for all the sentences in a dataset were aggregated and used to compute the final Precision, Recall and F1 measure. For each dataset, we ran "GFLM-Word" and "GFLM-Sentence" five times, each time with a different random initialization of the parameter values for the generative model. All the results reported for "GFLM-Word" and "GFLM-Sentence" are calculated by averaging the results obtained for these five different runs.

6. EXPERIMENT RESULTS

This section presents the results on the five human-annotated datasets and the three larger automatically-annotated datasets. For clarity and conciseness, we use "NB" as an abbreviation for the Naive Bayes classifier baseline method, "CR" for the Correlation-based baseline method, "FM-W" for GFLM-Word, and "FM-S" for GFLM-Sentence.

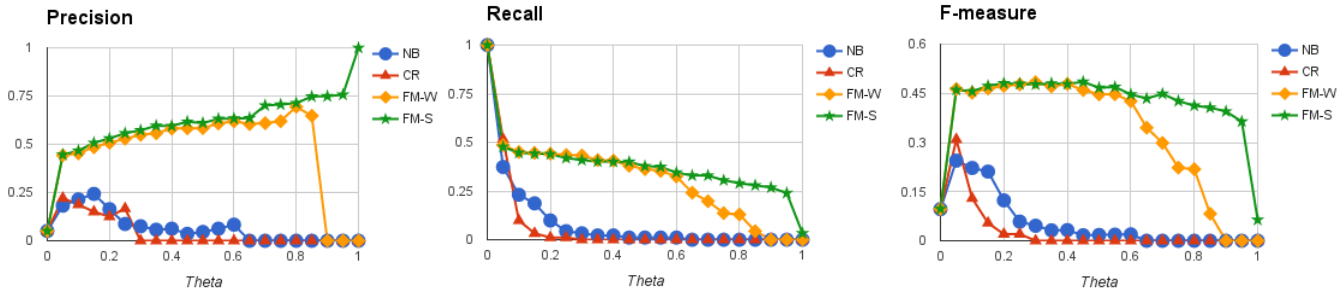
6.1 Comparison with baselines

We first compare our two proposed approaches, GFLM-Word and GFLM-Sentence, with the baseline algorithms mentioned in section 5.2. Table 3 presents a summary of the results on different datasets. The thresholding parameter θ for each algorithm was varied between $[0, 1]$ with an increment of 0.05. Table 3 reports the maximum F1 obtained by each algorithm for some particular value of θ and also reports the precision and recall corresponding to the maximum F1. For example, in case of Cellular Phone 1 dataset, NB (naive bayes) achieved maximum F1 of 0.2446 with corresponding Precision and Recall of 0.1818 and 0.3736 respectively, while FM-S (GFLM-Sentence) obtained 0.4853, 0.6169 and 0.4 for F1, Precision and Recall respectively.

Table 3 shows that our proposed approach outperforms

Table 3: Performance Comparison of GFLM with Baseline Approaches. Gains were found to be statistically significant via t-test at p-value < 0.0001 (†) and < 0.02 (‡)

Dataset	F1				Corresponding Precision				Corresponding Recall			
	NB	CR	FM-W	FM-S	NB	CR	FM-W	FM-S	NB	CR	FM-W	FM-S
Cellular phone1	0.2446	0.3092	0.4840†	**0.4853**†	0.1818	0.2206	0.5487	0.6169	0.3736	0.5164	0.4329	0.4
DVD player	0.3147	0.3420	**0.5125**†	0.5106†	0.2157	0.2268	0.6543	0.5559	0.581	0.6950	0.4212	0.4723
Mp3 player1	0.2947	0.2671	0.5570†	**0.5612**†	0.2406	0.2390	0.6397	0.6383	0.3800	0.3025	0.4933	0.5007
Digital camera1	0.3312	0.3380	**0.3831**‡	0.3808‡	0.3253	0.3870	0.5705	0.5504	0.3375	0.3	0.2886	0.2911
Digital camera2	0.2177	0.2555	0.4439†	**0.4559**†	0.1483	0.1619	0.5940	0.5303	0.4090	0.6060	0.3545	0.4
Cellular phone2	0.4051	0.4134	**0.597**†	0.5805†	0.3775	0.4156	0.6791	0.5795	0.4371	0.4112	0.5325	0.5816
Mp3 player2	0.4604	0.4634	**0.6666**†	0.6480†	0.4477	0.6495	0.6574	0.6195	0.4739	0.3601	0.6761	0.6793
Router	0.4805	0.5291	**0.6686**†	0.6439†	0.4213	0.8009	0.7487	0.6339	0.5593	0.3951	0.6040	0.6543



(a) Precision (b) Recall (c) F1 Measure

Figure 3: Performance Comparisons for Cell Phone 1 dataset; the results on other data sets are very similar. The x-axis represents the thresholding parameter θ and y-axis plots the corresponding performance measure.

the baseline approaches in terms of F1 by a large margin. Close observation also reveals that GFLM-Word and GFLM-Sentence perform significantly better than the baseline approaches in terms of Precision, which is a desired property we expect for opinion summarization. However, in some cases, the Recall value corresponding to the maximum F1 for baseline approaches were found to be larger than the same for GFLM-Word and GFLM-Sentence. One should note that this higher recall has little practical value as the corresponding precision is very low. On the other hand, GFLM-Word and GFLM-Sentence achieved comparatively very high precision while preserving reasonable recall.

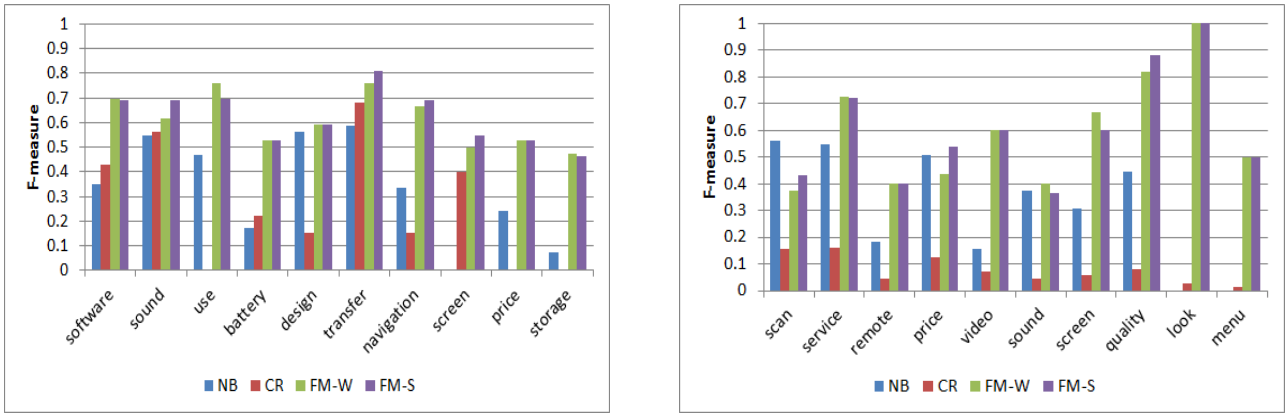
The precision-recall trade-off is better demonstrated via Figure 3. Here, we show how Precision, Recall and F1 vary with respect to the thresholding parameter for different compared approaches in case of "Cell phone 1" dataset. The plots on other data sets are very similar (which are not shown due to the space limit). In these figures, the thresholding parameter (we define it as θ) is represented along the x-axis, while the y-axis represents the performance measure, i.e., precision, recall or F1. Note that θ is varied between 0 to 1, which is appropriate for a probability cutoff. The score returned by GFLM-Word and GFLM-Sentence for a candidate feature f is naturally a probability. However, the output from naive bayes based and correlation based approaches are not necessarily probabilities, thus to apply the threshold, we normalized the output scores of the baseline approaches to make them fall into the range of 0 to 1.

From Figure (3a - 3c), we see that both GFLM-Word and GFLM-Sentence outperform the naive bayes and correlation baselines approaches by a significant margin in all measures, Precision, Recall, and F1. Another important observation is that both GFLM-Word and GFLM-Sentence attain this high performance for a wide range of θ variation. This is true for all the three measures. This suggests that our proposed approach is less sensitive to the variation of the thresholding parameter compared with the baseline approaches, and thus, is more robust.

Analysis: To better understand whether the reason why our approach outperforms baseline approaches is indeed due to its ability to learn from unlabeled sentences and its more principled way to remove noisy words from review sentences compared to the simple heuristic approaches, we provide a detailed analysis of the behaviors of these algorithms. Let us consider one example sentence from the "Cell Phone 1" dataset which talks about the feature "customer service":

"i returned to the store , whereupon the clerk insisted that he could make it work ; he could n't".

This is not a straightforward sentence that talks about a particular feature directly, therefore predicting the implicit feature "customer service" is non-trivial. In our experiments, the GFLM methods could infer the feature "customer service" correctly with a high probability, i.e., 0.85, while the baseline approaches could not do the same. Here is an intuitive explanation for that: First, The background model of GFLM filtered out all words except the two words "clerk" and "returned". For words "clerk" and "returned", the probabilities of being generated by the background model was found to be pretty small (less than 0.23). Whereas, the same probability was pretty high for the remaining words in the sentence. Next, words "clerk" and "returned" were found to be more likely generated by the feature "customer service" than any other features as suggested by the feature topic models. Thus, GFLM successfully inferred "customer service" as an implicit feature. On the other hand, The NB (Naive Bayes based) and CR (Correlation based) approaches inferred features "phone" (probability 0.61) and "sound" (probability 0.15) respectively. Note that, feature "phone" captures users comments about the overall product. It's interesting to see that both NB (Naive Bayes based) and CR (Correlation based) approaches fails to capture the actual implicit feature. This signifies that the noisy words ap-



(a) Mp3 Player 1: Feature-wise F1

(b) DVD Player: Feature-wise F1

Figure 4: Feature-wise performance of compared algorithms. The x-axis represents the top ten frequent features with explicit mentions sorted by their frequency from left to right, y-axis plots the corresponding F1 per feature.

pearing in the sentence drove these approaches into wrong prediction. It's also notable that NB inferred the feature ("phone") which is highly frequent compared to feature "customer service". Thus, the high prior probability of feature "phone" would have likely driven the decision for NB approach.

6.2 GFLM-Sentence vs. GFLM-Word

Next, we compare GFLM-Sentence and GFLM-Word. The results in Table 3 show that overall the two approaches are comparable in F1 on the manually tagged test sets, but GFLM-Word has consistently higher F1 on the three automatically tagged data sets. When examining their precision and recall, we note that GFLM-Sentence tends to be better in Recall, while GFLM-Word is often better in Precision. This suggests that a single correlated "signal word" with a feature can be very accurate in predicting an implicit feature, but there are also implicit features that may not have "obvious" strong signal words, and to predict such a feature, we will need to look at the overall evidence in a sentence as GFLM-Sentence does.

We further compare their sensitivity to the variation of the thresholding parameter and find that GFLM-Sentence seemed to be less sensitive than GFLM-Word as signified by the apparently larger "area under curve" attained by GFLM-Sentence for different datasets shown in Figure 3 [Similar results were obtained for other datasets too]. However, both NB (naive bayes) and CR (Correlation) based methods were found to be severely sensitive to the thresholding parameter variation. A slight change in the thresholding parameter resulted in big changes in the performance measures for both NB (naive bayes) and CR (Correlation).

6.3 Feature Level Analysis

Next, we tried to see the behavior of the algorithms at the feature level in more detail. Figure 4a-4b show these analyses in summary. Here, we plot the top ten frequent features (explicit mentions) along the x-axis sorted by their frequency from left to right in descending order and the y-axis plots the F1 of the corresponding feature obtained by different algorithms being compared. For space constraints, we picked two datasets to show, i.e., "Mp3 Player 1" and "DVD Player". However, we obtained similar results for other datasets too. Close observation of figure 4a-4b reveals that both GFLM-Word and GFLM-Sentence perform superior compared to the baseline approaches at the feature level too for almost all the features. For example, in case of features "price" and "storage" of the dataset "Mp3 Player 1", both GFLM-Word and GFLM-Sentence achieved a high value ($[0.45-0.53]$) for F1, while the corresponding F1 for the baseline approaches were significantly poor ($[0.0-0.25]$).

To dig more into the feature level analysis, we next tried to look at which words played the most important role for inferring different implicit features, i.e., the "signal words". For this analysis, we considered only the GFLM-Word approach. As discussed in section 4.3, when $P(z_{S,w} = f) \times (1 - P(z_{S,w} = B)) > \theta_f$ for some feature f and word w, GFLM-Word inferred f as an implicit feature and w is the word which contributed in making the inference. First we set θ_f to 0.6 and ran GFLM-Word on different datasets. Then we tried to find out $< f, w >$ pairs for which $P(z_{S,w} = f) \times (1 - P(z_{S,w} = B)) > \theta_f$ holds true. Next, for each feature f, we found out the top three words with the highest value for $P(z_{S,w} = f) \times (1 - P(z_{S,w} = B))$. This result is shown in Table 4. Here, we show the results for the human and automatically annotated datasets by combining similar product types. For example, the top three words which resulted in inferring "sound" as an implicit feature for "Cell Phone" dataset were found to be "quiet","loud" and "earpiece". This makes perfect sense as any human would also suggest the feature "sound" based on similar context words. Lets consider another example of feature "streaming" in case of "Router" data. Here, the top three words we found are "Netflix", "video" and "HD". These examples justify that the inference that our approach makes is based on sensible choices which mimic human intelligence to some extent, suggesting that the proposed generative model is quite effective to mine the review text to pick up the "right" signals for distinguishing and predicting implicit feature mentions. Note that our approach does not need any manual effort or tuning of parameters except for the threshold θ, to which the algorithms are not very sensitive as we discussed earlier. Thus it is reasonable to assume that our approach would be effective for mining implicit features in reviews of other natural languages, an important direction that we will pursue in the future.

Table 4: Identity association between feature words and other words

Dataset	Feature	word 1	word 2	word 3
Cellular phone (1 & 2)	sound	quiet	loud	earpiece
	size	small	pocket	fit
	battery	charge	life	long
	internet	wap	hotspot	end
	button	back	press	push
	design	robust	sleek	weight
	camera	image	flash	picture
Mp3 player (1 & 2)	software	install	update	xp
	button	cracked	press	pause
	storage	disk	huge	space
	warranty	date	replacement	repair
	transfer	kbps	usb	load
	battery	recharge	charge	plug
Digital camera (1 & 2)	use	access	highly	fun
	price	worth	cheaply	point
	design	flaw	plastic	superb
	battery	continue	solid	nice
	software	raw	os	consistently
	memory	fit	reader	large
Router	software	XP	proxy	program
	streaming	Netflix	video	HD
	installation	physically	error	manual
	price	worth	refund	Amazon
	port	usb	SharePort	Compatible
	speed	high	HD	mbps
DVD player	format	avi	file	able
	sound	optical	cd	vcd
	service	answer	busy	response
	price	worth	dollar	guess
	screen	light	silver	load
	quality	build	pretty	begin

7. CONCLUSION AND FUTURE WORK

Mining implicit feature mentions from review data is an important task required in any applications involving summarizing and analyzing review data; without an automated method for recognizing the implicit mentions of features in many review sentences would make it impossible to understand such opinions in detail at the feature level. We proposed a novel generative feature language model to solve this problem in a general and unsupervised way. In contrast with the previous heuristic approaches which require much ad hoc manual work in tuning parameters, our new approach is grounded in statistical learning and optimizes the parameters automatically through an EM algorithm. Our approach also elegantly solves the problem of filtering noisy words by using a background language model.

Another contribution of our work is the creation of eight new English datasets, including five manually annotated data sets and three automatically annotated large data sets, which will facilitate future research on the problem.

Evaluation on these data sets show that the proposed approach is very effective and outperforms both a state-of-the-art baseline and a straightforward adaptation of a standard supervised learning approach by a large margin. The new approach is also more robust with just one parameter to set, to which our algorithm is not very sensitive (setting it to a value of 0.5 would work well on all the data sets).

Although we evaluated the proposed method using English reviews, the approach is general and can thus be expected to work well on other natural languages as well. Further evaluation of our method using reviews in other natural languages would be an interesting future direction. Another direction is to study the impact of the implicit feature mining on applications (e.g., to see how opinion summaries may be improved through incorporating mined implicit features).

8. ACKNOWLEDGMENTS

This work is supported by the National Science Foundation under Grant Numbers CNS-1513939 and CNS-1408944. We thank the four anonymous reviewers for their valuable feedback to help us improve the paper.

9. REFERENCES

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[2] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[3] A. Esuli and F. Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, volume 6, pages 417–422, 2006.

[4] T. Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[5] Z. Hai, K. Chang, and J.-j. Kim. Implicit feature identification via co-occurrence association rule mining. In *Computational Linguistics and Intelligent Text Processing*, pages 393–404. Springer, 2011.

[6] T. Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 289–296. Morgan Kaufmann Publishers Inc., 1999.

[7] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of ACM KDD 2004*, pages 168–177. ACM, 2004.

[8] H. D. Kim, K. Ganesan, P. Sondhi, and C. Zhai. Comprehensive review of opinion summarization. 2011.

[9] B. Liu. *Web data mining: exploring hyperlinks, contents, and usage data*. Springer Science & Business Media, 2007.

[10] B. Liu. Sentiment analysis and subjectivity. *Handbook of natural language processing*, 2:627–666, 2010.

[11] B. Liu, M. Hu, and J. Cheng. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of WWW 2005*, pages 342–351. ACM, 2005.

[12] J. McAuley, R. Pandey, and J. Leskovec. Inferring networks of substitutable and complementary products. In *Proceedings of ACM KDD 2015*, pages 785–794. ACM, 2015.

[13] Q. Mei, X. Ling, M. Wondra, H. Su, and C. Zhai. Topic sentiment mixture: modeling facets and opinions in weblogs. In *Proceedings of WWW 2007*, pages 171–180. ACM, 2007.

[14] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135, 2008.

[15] A.-M. Popescu and O. Etzioni. Extracting product features and opinions from reviews. In *Natural language processing and text mining*, pages 9–28. Springer, 2007.

[16] Q. Su, X. Xu, H. Guo, Z. Guo, X. Wu, X. Zhang, B. Swen, and Z. Su. Hidden sentiment association in chinese web opinion mining. In *Proceedings of WWW 2008*, pages 959–968. ACM, 2008.

[17] W. Wang, H. Xu, and W. Wan. Implicit feature identification via hybrid association rule mining. *Expert Systems with Applications*, 40(9):3518–3531, 2013.

[18] T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of HLT and EMNLP*, pages 347–354. Association for Computational Linguistics, 2005.

[19] Y. Zhang and W. Zhu. Extracting implicit features in online customer reviews for opinion mining. In *Proceedings of WWW 2013*, pages 103–104, 2013.

[20] L. Zhuang, F. Jing, and X.-Y. Zhu. Movie review mining and summarization. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 43–50. ACM, 2006.