Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

Extracting Verb Expressions Implying Negative Opinions

Huayi Li[†], Arjun Mukherjee[‡], Jianfeng Si[§], Bing Liu[†]
[†]Department of Computer Science
University of Illinois at Chicago, IL, USA
[‡]Department of Computer Science
University of Houston, TX, USA
[§]Institute for Infocomm Research, Singapore
lhymvp@gmail.com, arjun@cs.uh.edu, thankjeff@gmail.com, liub@cs.uic.edu

Abstract

Identifying aspect-based opinions has been studied extensively in recent years. However, existing work primarily focused on adjective, adverb, and noun expressions. Clearly, verb expressions can imply opinions too. We found that in many domains verb expressions can be even more important to applications because they often describe major issues of products or services. These issues enable brands and businesses to directly improve their products or services. To the best of our knowledge, this problem has not received much attention in the literature. In this paper, we make an attempt to solve this problem. Our proposed method first extracts verb expressions from reviews and then employs Markov Networks to model rich linguistic features and long distance relationships to identify negative issue expressions. Since our training data is obtained from titles of reviews whose labels are automatically inferred from review ratings, our approach is applicable to any domain without manual involvement. Experimental results using real-life review datasets show that our approach outperforms strong baselines.

Introduction

Sentiment analysis has attracted a great deal of attention in recent years due to the rapid growth of e-commerce and social media services (Liu 2012; Pang and Lee 2008). There exists an extensive body of work on major tasks like aspect extraction (Hu and Liu 2004; Chen, Mukherjee, and Liu 2014; Popescu, Nguyen, and Etzioni 2005; Xu et al. 2013; Fang, Huang, and Zhu 2013), opinion and polarity identification e.g., (Hu and Liu 2004; Pang and Lee 2008; Wilson, Wiebe, and Hwa 2004; Yu, Kaufmann, and Diermeier 2008; Jiang et al. 2011) and subjectivity analysis (Hatzivassiloglou and Wiebe 2000). The task of discovering phrasal opinions has also been studied extensively. For example, Wilson et. al. (2009) investigate phrasal opinions with opinion lexicons. Fei, Chen and Liu (2014) used topic models to discover noun phrases. Zhang and Liu (2011) identify noun phrases implying inexplicit opinions but our problem emphasizes verb expressions where verbs play a significant role and verb expressions are more likely to convey an opinion towards important issues/problems of products and services. Based on our experimental data, we found 81% of the negative verb expressions describe product issues.

Nowadays apart from generic reviews, we also find dedicated feedback/review forums like Complaints.com, Pissedconsumer.com, which collect consumer complaints about products and services. It is worth noting that negative opinions often weigh more than positive ones to businesses as companies want to know the key issues with their products and services in order to make improvements based on users' feedback. Complaints or issues are usually expressed as verb expressions with fine-grained sentiments often implying product malfunctions and failing to work as expected. They are often not captured by opinion words or phrases. For instance, the sentence "*The mouse double-clicks on a single click*" implies a negative opinion about the mouse, which is also an issue or problem of the mouse. But the sentence does not have any opinion word. Discovering such issues is thus challenging because the problem setting departs from existing works, which mainly use opinion adjectives, adverbs, nouns and verbs (such as hate, dislike, etc.) for opinion analysis.

In this paper, we focus on tackling the problem of mining verb expressions implying negative opinions from customer reviews because they often describe product issues. We propose a two-step approach. First, we parse each sentence using a shallow parser to obtain phrasal chunks, which are then used to extract verb expressions. Next, we propose a supervised model with Markov Networks to rank verb expressions based on their probabilities of implying negative opinions. However, our approach does not use any manually labeled data for learning. Instead, our training verb expressions are from titles of negatively rated reviews and positive rated reviews. Titles of negative reviews often contain negative verb expressions, while titles of positive opinions may contain both positive and neutral ones. Clearly, the automatically extracted labels contain some noise, but we will see that they still enable us to build good models.

Two baselines that we compare with are Naïve Bayes and SVM. With n-gram features, they can coarsely model the sequential dependency of adjacent words to some extent. However, our proposed Markov Networks method is more powerful because it takes into consideration correlations between both adjacent and non-adjacent words of different

Algorithm 1 Extracting Verb Expressions From a Sentence

Input: a sentence $sent$, the maximum number words in any verb expression K
Output: a list of verb expressions VE represented by their starting and ending positions

1: $chunks \leftarrow Chunker.chunk(sent)$
2: $VE \leftarrow \varnothing$
3: **for** each $chunk_i \in chunks$ s.t. $i \in [1, chunks.length]$ **do**
4: **if** $chunk_i.label == VP$ and $chunk_i$ contains verbs other than be **then**
5: *// extraction begins from VP with be verbs excluded*
6: $chunkCnt \leftarrow 1$
7: $start \leftarrow chunk_i.start$
8: $end \leftarrow chunk_i.end$
9: *// add optional NP to the left of* $chunk_i$
10: **if** $i > 1$ and $chunk_{i-1}.label == NP$ **then**
11: $start \leftarrow chunk_{i-1}.start$
12: $chunkCnt \leftarrow chunkCnt + 1$
13: **end if**
14: *// add optional PP, PRT, ADJP, ADVP, NP to the right*
15: **for** $j = i + 1$ to $chunks.length$ **do**
16: **if** $chunkCnt == K$ **then** // the maxmimum number of tokens in verb expressions is reached
17: break
18: **end if**
19: **if** $chunk_j.label \in \{PP, PRT, ADJP, ADVP, NP\}$ **then**
20: $end \leftarrow chunk_j.end$
21: $chunkCnt \leftarrow chunkCnt + 1$
22: **end if**
23: **end for**
24: $VE \leftarrow VE \cup < start, end >$
25: **end if**
26: **end for**

parts-of-speech. The precision-recall curve shows that modeling rich linguistic features and long distance relationships among the expressions enable the proposed method to outperform two baselines. Further, since most verb expressions describe product issues, we use ranking to evaluate the results based on the issues identified by each algorithm. For this, we employed the popular information retrieval measure NDCG (Normalized Discounted Cumulative Gain). NDCG results show that our approach can discover more important issues better than the baselines.

Verb Expressions Extraction

The first step in our task is to extract candidate verb expressions that may imply opinions. Phrase extraction has been studied by many researchers, e.g., (Kim et al. 2010) and (Zhao et al. 2011). However the phrases they extract are mostly frequent noun phrases in a corpus. In contrast, our verb expressions are different because they are verb oriented and need not to be frequent. In our settings, we define a verb expression to be a sequence of syntactically correlated phrases involving one or more verbs and the identification of the boundary of each phrase is modeled as a sequence labeling problem. The boundary detection task is often called chunking. In our work, we use the OpenNLP chunker[1] to parse the sentence. A chunker parses a sentence into a sequence of semantically correlated phrasal chunks. The following example shows the chunk output of a sentence from the chunker:

[**NP** *Windows and Linux*][**VP** *do not respond*] [**PP** *to*][**NP** *its scroll button*]

Each chunk is enclosed by brackets with the first token being its phrase level bracket label. Details of the chunk labels can be found in (Bies et al. 1995). Our definition of verb expression is a sequence of chunks centered at a Verb Phrase (**VP**) chunk with an optional Noun Phrase (**NP**) chunk to its left and optional Prepositional Phrase (**PP**), Particle (**PRT**), Adjective Phrase (**ADJP**), Adverb Phrase (**ADVP**) and **NP** to its right. In order to distinguish our work from adjective-based opinion, verb expressions with only be verbs (*is, am, are*, etc.) are excluded because such clauses usually involve only adjective or other types of opinions. A verb expression is smaller than a clause or sentence in granularity but is enough to carry a meaningful opinion. We also don't want a verb expression to contain too many chunks. So we only extract up to K chunks. In our experiment, we set K equal to 5 because a typical verb expression has at most 5 chunks, namely, one **VP**, one **NP** and optional **PP**, **ADJP** or **ADVP**. Algorithm 1 details the extraction procedure. We first apply the chunker to parse the sentence into a sequence of chunks (line 1) and then expand all VP with neighboring **NP**s, **PP**s,

[1] http://opennlp.apache.org/

Table 1: Definitions of feature functions $f(X, Y)$ for $y \in \{+1, -1\}$

Feature function	**Feature Category**
hasWord$(X, w_i) \wedge$ isPOS$(w_i, t) \wedge (Y = y)$	unigram-POS feature
hasWord$(X, w_i) \wedge$ isNN$(w_i) \wedge$ hasWord$(X, w_j) \wedge$ isVB$(w_j) \wedge (Y = y)$	NN-VB feature
hasWord$(X, w_i) \wedge$ isNN$(w_i) \wedge$ isNegator$(w_i) \wedge$ hasWord$(X, w_j) \wedge$ isVB$(w_j) \wedge (Y = y)$	NN-VB feature with negation
hasWord$(X, w_i) \wedge$ isNN$(w_i) \wedge$ hasWord$(X, w_j) \wedge$ isJJ$(w_j) \wedge (Y = y)$	NN-JJ feature
hasWord$(X, w_i) \wedge$ isNN$(w_i) \wedge$ isNegator$(w_i) \wedge$ hasWord$(X, w_j) \wedge$ isVB$(w_j) \wedge (Y = y)$	NN-JJ feature with negation
hasWord$(X, w_i) \wedge$ isRB$(w_i) \wedge$ hasWord$(X, w_j) \wedge$ isVB$(w_j) \wedge (Y = y)$	RB-VB feature
hasWord$(X, w_i) \wedge$ isRB$(w_i) \wedge$ isNegator$(w_i) \wedge$ hasWord$(X, w_j) \wedge$ isVB$(w_j) \wedge (Y = y)$	RB-VB feature with negation

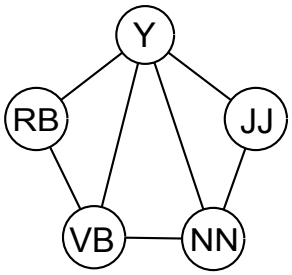

Figure 1: Representation of a Markov Network Segment

ADJPs and **ADVP**s (line 4-25). Because Noun phrases(**NP**) often function as subjects, we first seek to the left of the **VP** to include the optional **NP** (line 10-13) and then we seek to the right for phrases that are dependencies of the **VP** (line 15-23). Note that the final output verb expression is a sequence of continuous chunks. We evaluate the results of the extraction of verb expressions and find that 82.3% of verb expressions are extracted correctly which paves the way for the supervised learning task in the next section.

Proposed Markov Networks

Verb expressions implying opinions are structured phrases having a fine-grained and specific sentiment. To determine their sentiments, the grammatical dependency relations of words matter. Hence, we need a model that can encode the grammatical dependency between words.

Our training data is collected from titles of reviews whose ratings are 1(lowest) or 5(highest). The labels in our training data are obtained automatically from review ratings rather than from manual labeling because any large scale manual labeling is very expensive and domain dependent. With the automatically labeled training data, we propose to use a Markov Networks (abbreviated as MN from here on) based approach to identify negative verb expressions. MN is an undirected graphical model that deals with inference problems with uncertainty in observed data. Each node in the networks represents a random variable and each edge represents a statistical dependency between the connected variables. A set of potential functions are defined on the cliques of the graph to measure compatibility among the involved nodes. MN thus defines a joint distribution over all the nodes in the graph/network encoding the Markov property of a set of random variables corresponding to the nodes. Unlike similar models such as Markov Chains and Bayesian Networks, Markov Networks are not limited to specifying one-way casual links between random variables. Markov networks are undirected graphs meaning that influence may flow in both directions. We choose to use the undirected graph because words in the verb expressions have mutual causalities. The sentiment orientation of a verb expression is jointly determined by its words. For example, in the sentence "*lights of the mouse would go out*". It is "*light*", "*go*" and "*out*" as a whole that assign a negative opinion to the verb expression. Without "*lights*", this verb expression does not imply any sentiment because "*go out*" is a general phrase and can be interpreted in many different ways. Likewise without "*go*" and "*out*", we would not know it is issue of the lights of the mouse.

Intuitively nouns, verbs, adjectives and adverbs are the most important parts-of-speech(POS). So we want to only model them in the Markov Networks. As shown in Figure 1 , nodes in the Markov Networks are verbs (**VB**), adjectives (**JJ**), nouns (**NN**), adverbs (**RB**), and the class variable Y. Edges depict the correlation between the words. Every node has an edge to Y but only verbs and adjectives have connections to nouns. There are no edges from adjectives to verbs or adverbs, and no edges from nouns to adverbs according to the grammatical dependency relation between different parts-of-speech.

As a result, there are three types of maximal cliques defined in our graph: $C_{VN} = \{VB, NN, Y\}$, $C_{VR} = \{VB, RB, Y\}$ and $C_{NJ} = \{NN, JJ, Y\}$. We define feature function f_c for each clique c as listed in Table 1. Note that in order to deal with sentiment with negations, we add the boolean function to determine whether a noun (such as "*nothing*"), a verb (such as "*never*") is a negator for the feature indicator functions. Words like "*in*", "*out*", "*up*", "*down*" etc., are not always prepositions(**IN**). They can also be used as adverbial particles (**RP**), e.g. ("*lights went out*", "*router locked up*"). For simplicity, we use **RB** to denote both adverbs and particles. Similarly, in our setting, **NN** is defined in a broad sense including **NN**, **NNS**, **NNP**, **NNPS**,

PRP(Personal pronoun), and **WP**(Wh-pronoun) etc.

We denote $Y = +1$ as the verb expression implying a negative opinion, and $Y = -1$ for non-negative opinion. Each feature function is a boolean indicator of properties of observed verb expression X and class label Y pairs.

A verb expression X associated with the class Y is a subgraph of the Markov Networks and its unnormalized measure of joint probability $\tilde{P}(X, Y)$ can be computed by the product of factors over the maximal cliques $\mathcal{C}$ as seen in equation 1, according to Hammersley-Clifford theorem (Murphy 2012).

$$\tilde{P}(X, Y) = \exp\Big(\sum_{c \in \mathcal{C}} \lambda_c f_c(X, Y)\Big) \tag{1}$$

Then we can derive the conditional probability of class Y given the observed verb expression X by equations 2 and 3 where $Z(X)$ is the normalization term.

$$P(Y|X) = \frac{1}{Z(X)} \tilde{P}(X, Y) \tag{2}$$

$$Z(X) = \sum_{Y} \tilde{P}(X, Y) \tag{3}$$

Finally, the predicted class $\hat{y}$ is most probable label :

$$\hat{y} = \underset{y}{\operatorname{argmax}}\ P(Y = y|X) \tag{4}$$

As we will see in the next section that our MN based approach can capture opinions that are implied by multiple words as a whole in a verb expression together rather than by any single word alone. For example, in our experiments, we find some specific idiomatic phrases implying negative opinions such as "*fingers get off kilter*" in mouse reviews as "*get off kilter*" is well encoded by Markov Networks. Note that even each individual word in the expression is not an indicator of negative opinion nor a frequently used term, the phrase appears much more frequently in negative review titles than in non-negative ones. So it is not so difficult for our model to detect as MN estimates the joint distribution of the observed expressions and hidden variable Y by measuring compatibility among the involved random variables with the class label. In contrast, failing to model the local dependency between words, the two baseline models have less expressive power than Markov Networks, especially in the absence of explicit opinion indicative words.

We also examined bigram features which can coarsely capture some dependencies between adjacent words in some way. The results show that bigrams still cannot model longer range dependencies becaue n-grams can only capture adjacent sequential dependencies, while in a MN, the dependencies captured are of much wider range because MN can capture grammar dependencies covering different combinations of cliques defined in them.

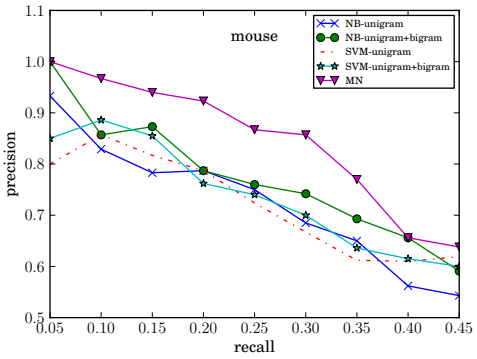

Figure 2: Precision recall curve for the mouse domain

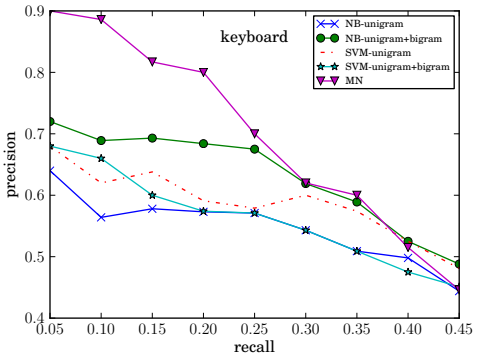

Figure 3: Precision recall curve for the keyboard domain

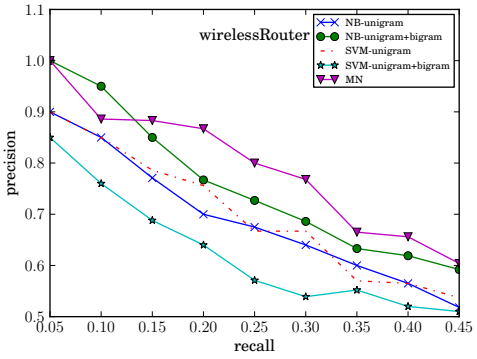

Figure 4: Precision recall curve for the router domain

Experiments

Datasets

We conduct our experiments using customer reviews from three electronic product domains: mouse, keyboard, and wireless router, collected from Amazon.com. A review consists of its date and time, author, rating (1-5), title and text body. A review is positive if it is rated 4 or 5 stars or negative if it is rated 1 or 2 stars. We only utilize the last three features of a review. Table 2 shows the statistics of our data.

Table 2: Number of verb expressions extracted for training and testing for each of the three domains

Domain	Training		Testing	
	Negative	other	Negative	other
mouse	687	5334	299	536
keyboard	799	7434	312	1446
router	1845	9686	324	782

Experiment Settings

Training instances for our models are verb expressions extracted from titles of positive (5 stars) and negative (1 star) reviews. We observed that negative verb expressions are abundant in titles of reviews whose rating is 1 star, but both positive and neutral verb expressions can occur in 5-star review titles. So verb expressions in 1-star review titles are used as negative and those in 5-star review titles are used as non-negative. Test instances are verb expressions from both the titles and bodies of reviews whose ratings are 1 or 2 stars and they are labeled manually by two human judges. We also performed an agreement study of the manual labeling using Cohen's Kappa (κ), which gives us $\kappa = 0.86$, indicating substantial agreement.

Baselines

We compare our model with two baselines, Naïve Bayes (NB) and Support Vector Machines (SVM).

NB and SVM both assume the bag-of-words assumption. In our case, it means that the running words of a verb phrases are independent given its sentiment orientation. While this assumption is clearly false for most tasks, it turns out to work fairly well. We implemented NB by ourselves and used LIBSVM[2] library for SVM implementation. Note that the standard SVM does not provide probability estimates. But LIBSVM allows us to produce the probability estimate for each test instance for each class by using pairwise coupling (Wu, Lin, and Weng 2004).

We use only verbs, adjectives, nouns, adverbs and particles with their attached part-of-speech (POS) tags as features. For simplicity, we use RB to denote both adverbs and particles. Specifically, our features are lemmatized words of these types associated with their POS tags. For example, break-VB, breaks-VB, and broke-VB are represented with only their lemma break-VB. This helps reduce the effect of feature sparsity and thus improves classifications.

Quantitative Results

Our algorithm aims to find the most important and frequent product issues. We thus treat this task as a ranking problem. The probability of a verb expression that implies a negative opinion is $P(Y = +1|X)$. We rank the verb phrases extracted from the test data in a descending order of their probabilities of being negative opinions. Figure 2-4 show that the top ranked candidates are of high precision. In practice, in order to understand major defects of a product we don't need to identify every complaint from every customer. Frequent and important issues indicated by top verb expressions from our algorithm are sufficient to summarize the user feedback on the product.

Not surprisingly, MN outperforms SVM and NB at the top in all domains. Keyboard is a difficult domain where NB and SVM have low precisions at the top, while MN still give a relative high precision. But as the recall grows larger, the improvement becomes less. One of drawbacks of Naïve Bayes is its evidence over-counting. For example in the keyboard review, the terms "*space*" and "*bar*" occur frequently together. NB classifier counts "*space*" and "*bar*" separately but "*space bar*" means only one object which is not a good reflection of the true probability distribution. Although SVM does not suffer from the problem of redundant features, it does not capture complex correlations among words. It is these correlations that make the entire phrase opinionated. Failing to model the local dependency between words, the two base line models have less expressive power than Markov Networks, especially in the absence of explicit opinion words.

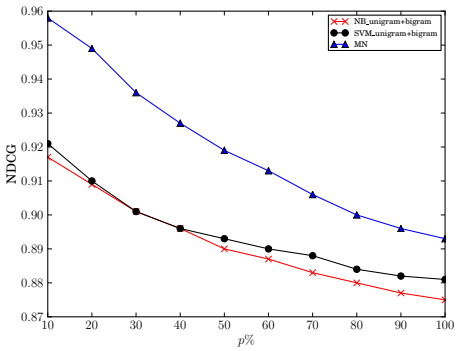

Figure 5: average NDCG score of three domains

At the same time, to show that our ranking strategy of negative verb expressions could discover the most frequent and important issues of products. We manually annotate each verb expression with an issue type. For example in the mouse domain, the most frequent issue types include "*durability*", "*compatability*", "*sensitivity*" and so on. We then apply Normalized Discounted Cumulative Gain (NDCG) to measure the ranking quality for the test data. The relevance score rel_i of the verb expression ranked at i-th position is the number of times its issue type being mentioned by reviewers which reflects the importance of the issue. If the expression is not about any issue, the relevance score is 0. In Figure 5 we show the average NDCG score of all the three domains at different percentage for the test data. Results show that our approach can discover important or common issues of products better than the baselines.

Qualitative Results

Note that negative verb expressions that contain explicit opinion indicating words (e.g., stop, fail, and break) are easy

[2]http://www.csie.ntu.edu.tw/~cjlin/libsvm/

Table 3: Probabilities of verb expressions implying negative opinions in the mouse domain

Example	NB	SVM	MN
Buttons wear out really fast	0.667	0.625	0.889
I developed bad wrist pain	0.172	0.195	0.800
your fingers get off kilter	0.190	0.146	0.722
trackball does not glide as well as previous model	0.298	0.335	0.679
(that experience) sends me over the edge	0.225	0.488	0.667

Table 4: Probabilities of verb expressions implying negative opinions in the keyboard domain

Example	NB	SVM	MN
Some keys don't get typed	0.376	0.714	0.780
keyboard does not recognize input Shift-Space-W	0.672	0.811	0.779
only works sporadically	0.371	0.510	0.681
The lights would go out	0.116	0.144	0.654
Working with win7 half the time	0.357	0.104	0.617

Table 5: Probabilities of verb expressions implying negative opinions in the router domain

Example	NB	SVM	MN
The router will stop handing out IP addresses	0.415	0.458	0.833
I still get marginal signal	0.144	0.215	0.800
I gave up on the WRT54G	0.045	0.260	0.750
router will lock up	0.132	0.356	0.667
waiting on hold for 40 minutes	0.322	0.394	0.667

to catch by all models and are naturally ranked higher, because these opinion words alone provide enough evidences for the two baseline models to identify explicit negative verb phrases. However, the key contribution of our work is to capture relatively implicit verb expressions that imply negative opinions with the help of dependency structures via MN. Hence, it is worthwhile to compare different methods for harder cases. In Table 3-5, we show some discovered expressions. We can see that the probabilities of these expressions for being negative are much higher for MN than for the baselines when explicit opinion words are absent. Besides, given enough training data, our proposed model can even catch idioms such as "*send someone over the edge*" (last row of Table 3).

Related Work

Existing work has studied the opinion mining problem in document, sentence, and aspect levels (Pang and Lee 2008; Liu 2012). However the aspects they found are mostly noun and noun phrases. Thus their approaches are not suitable for our problem.

Most current works on polarity analysis rely heavily on opinion lexicons. Kim and Hovy (2006) used opinion-bearing verbs to help identify opinion holders and aspects (which they call topics), but their task is different. Although domain independent opinion verbs (such as hate, and dislike) are also good indicators of opinion polarity, there are many domain-dependent or context-dependent verb expressions that also indicate opinions, e.g., "*keystroke doesn't register*" and "*space key continued to stick*."

There have been a large body of work done on the semantics of verbs and verb phrases. Sokolova and Lapalme (2008) incorporate semantic verb categories including verb past and continuous forms into features sets. However, hard coded grammar rules and categorization of verbs make it hard to generalize their algorithm to other domains. (Neviarouskaya, Prendinger, and Ishizuka 2009) also built a rule-based approach to incorporate verb classes from VerbNet (Schuler, Korhonen, and Brown 2009) to detect the sentiment orientation of sentences. However, its rule based approaches again suffer from domain-specificity problem.

Linguistic knowledge such as dependency relation among words used in our proposed model has been proven to yield significant improvements (Joshi and Rosé 2009). Markov Networks have also been successfully applied to image processing (Lan et al. 2006), information retrieval (Metzler and Croft 2005). Our method is inspired by these works.

There are several papers on extracting opinion expressions (Breck, Choi, and Cardie 2007; Choi and Cardie 2010; Johansson and Moschitti 2011). These works mainly use Conditional Random Fields (CRF). CRF is a supervised sequence labeling method and it requires labeled training data. Recently, (Yang and Cardie 2012) used semi-CRF to allow sequence labeling at the segment level rather than just at the words level. But all these existing works did not focus on verb expressions. Furthermore, our main goal is not only to extract verb expressions, but also to find their implied sentiments.

Conclusions

In this paper, we dealt with the problem of discovering verb expressions that imply negative opinions. Such expressions usually describe product issues. Our work differs from other works as it emphasizes the role of verbs and their correlations with other words. We proposed an algorithm to extract such verb expressions and employed Markov Networks to solve the problem. Experimental results showed that our model can effectively find negative verb expressions that prevail in reviews indicating critical product issues. Since our training data is obtained from titles of reviews whose labels are automatically inferred from review ratings, our approach can be easily applied to any review or product domain. This is beneficial for companies and business who would like to improve their products or service based on users' feedback.

References

Bies, A.; Ferguson, M.; Katz, K.; and MacIntyre, R. 1995. Bracketing Guidelines for Treebank II Style Penn Treebank Project. Technical report, Linguistic Data Consortium.

Breck, E.; Choi, Y.; and Cardie, C. 2007. Identifying Expressions of Opinion in Context. In *IJCAI*, 2683–2688.

Chen, Z.; Mukherjee, A.; and Liu, B. 2014. Aspect extraction with automated prior knowledge learning. In *ACL*, 347–358.

Choi, Y., and Cardie, C. 2010. Hierarchical Sequential Learning for Extracting Opinions and Their Attributes. In *ACL*, 269–274.

Fang, L.; Huang, M.; and Zhu, X. 2013. Exploring weakly supervised latent sentiment explanations for aspect-level review analysis. In *CIKM*, 1057–1066.

Fei, G.; Chen, Z.; and Liu, B. 2014. Review topic discovery with phrases using the pólya urn model. In *COLING*, 667–676.

Hatzivassiloglou, V., and Wiebe, J. M. 2000. Effects of adjective orientation and gradability on sentence subjectivity. In *COLING*, 299–305.

Hu, M., and Liu, B. 2004. Mining and summarizing customer reviews. In *KDD*, 168–177.

Jiang, L.; Yu, M.; Zhou, M.; Liu, X.; and Zhao, T. 2011. Target-dependent Twitter Sentiment Classification. In *ACL*, 151–160.

Johansson, R., and Moschitti, A. 2011. Extracting opinion expressions and their polarities - exploration of pipelines and joint models. In *ACL*, 101–106.

Joshi, M., and Rosé, C. P. 2009. Generalizing Dependency Features for Opinion Mining. In *ACL*, 313–316.

Kim, S.-M., and Hovy, E. 2006. Extracting opinions, opinion holders, and topics expressed in online news media text. In *Proceedings of the Workshop on Sentiment and Subjectivity in Text*, SST '06, 1–8. Stroudsburg, PA, USA: Association for Computational Linguistics.

Kim, S. N.; Medelyan, O.; Kan, M.-Y.; and Baldwin, T. 2010. SemEval-2010 task 5: Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, 21–26. Association for Computational Linguistics.

Lan, X.; Roth, S.; Huttenlocher, D. P.; and Black, M. J. 2006. Efficient Belief Propagation with Learned Higher-Order Markov Random Fields. In *ECCV*, 269–282.

Liu, B. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers.

Metzler, D., and Croft, W. B. 2005. A markov random field model for term dependencies. In *SIGIR*, 472–479.

Murphy, K. 2012. *Machine Learning: a Probabilistic Perspective*. MIT Press.

Neviarouskaya, A.; Prendinger, H.; and Ishizuka, M. 2009. Semantically distinct verb classes involved in sentiment analysis. In *IADIS*, 27–35.

Pang, B., and Lee, L. 2008. Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval* 2(1-2):1–135.

Popescu, A.-M.; Nguyen, B.; and Etzioni, O. 2005. OPINE: Extracting Product Features and Opinions from Reviews. In *HLT/EMNLP*.

Schuler, K. K.; Korhonen, A.; and Brown, S. W. 2009. VerbNet overview, extensions, mappings and applications. In *HLT-NAACL*, 13–14.

Sokolova, M., and Lapalme, G. 2008. Verbs Speak Loud: Verb Categories in Learning Polarity and Strength of Opinions. In *Canadian Conference on AI*, 320–331.

Wilson, T.; Wiebe, J.; and Hoffmann, P. 2009. Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis. *Computational linguistics* 35(3):399–433.

Wilson, T.; Wiebe, J.; and Hwa, R. 2004. Just How Mad Are You? Finding Strong and Weak Opinion Clauses. In *AAAI*, 761–769.

Wu, T.-F.; Lin, C.-J.; and Weng, R. C. 2004. Probability Estimates for Multi-class Classification by Pairwise Coupling. *Journal of Machine Learning Research* 5:975–1005.

Xu, L.; Liu, K.; Lai, S.; Chen, Y.; and Zhao, J. 2013. Mining opinion words and opinion targets in a two-stage framework. In *ACL*, 1764–1773.

Yang, B., and Cardie, C. 2012. Extracting opinion expressions with semi-markov conditional random fields. In *EMNLP-CoNLL*, 1335–1345.

Yu, B.; Kaufmann, S.; and Diermeier, D. 2008. Exploring the characteristics of opinion expressions for political opinion classification. In *Proceedings of the 2008 International Conference on Digital Government Research*, dg.o '08, 82–91. Digital Government Society of North America.

Zhang, L., and Liu, B. 2011. Identifying noun product features that imply opinions. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, 575–580. Stroudsburg, PA, USA: Association for Computational Linguistics.

Zhao, W. X.; Jiang, J.; He, J.; Song, Y.; Achananuparp, P.; Lim, E.-P.; and Li, X. 2011. Topical Keyphrase Extraction from Twitter. In *ACL*, 379–388.