A Linear-chain CRF-based Learning Approach For Web Opinion Mining

Luole Qi

August 19, 2010

Abstract

The task of opinion mining from product reviews is to extract the product entities, opinions on the entities and determine whether the opinions are positive, negative or neutral. Reasonable performance on this task has been achieved by employing rule-based, statistical approaches or generative learning models such as hidden Markov model (HMMs). In this paper, we proposed a discriminative model using linear-chain Conditional random field (CRFs) for opinion mining and extraction. CRFs can naturally incorporate arbitrary, non-independent features of the input without making conditional independence assumptions among the features. This can be particularly important for opinion mining on product reviews. We evaluate our approach base on three criteria: recall, precision and F-score of extracted entities, opinions and their polarity. Compared to other methods, our approach is more effective for opinion mining tasks.

1 Introduction

The reviews are now well recognized increasingly useful in business, education, especially in e-commerce, since they contain valuable opinions and customers could assess a product by reading opinions of other customers, which will help them to decide whether to purchase the product or not. Nowadays, many e-commerce websites such as Amazon.com, Yahoo shopping, Epinions allow users to post their opinions freely. Thus, there are usually a large amount of product reviews available on the internet. The reviews number even could be thousands in some large websites for a hot product, which makes it difficult for a potential customer to go over all of them.

This is indeed a problem for the customers. In response, researchers have done some work on opinion mining which aims to extract the essential information of reviews. Previous works have been based on two major approaches: rule-based techniques [2] and statistic methods [10]. In [1], a new learning approach based on a sequence model named hidden Markov model (HMMs) was adopted and was proved more effective. However, HMMs have the limitation that it is difficult to model arbitrary, dependent features of the input sequence.

Conditional random field (CRFs)[11] are discriminative graphical models that can model these overlapping, non-independent features. A special case, linear-chain CRFs, can be thought of as the undirected graphical model version of HMMs. Motivated by the fact that CRFs have out-performed HMMs on language processing[12][13], we proposed a linear-chain CRF-based instead of HMM-based learning approach to mine and extract opinions from product reviews on the web in this paper. Our objective is to answer the following questions: (1) how to construct and restrict our linear-chain CRFs model by defining feature functions. (2) How to choose criteria to train our specific model from the manually labeled data. (3) How to automatically extract potential product entities and opinion entities from the reviews and identify opinion polarity with our trained model. In our work, we evaluate the model on recall, precision and F-score of extracted entities, opinions and their polarity, and the experimental results prove the proposed approach in web opinion mining and extraction from online product reviews is effective.

In summary, this paper has the following contributions: 1) we demonstrate linear-chain CRFs models performs better than L-HMMs approach integrating linguistic features in opinion mining and extraction. 2) The feature functions of CRFs we defined in our work for the model construction are proved to be robust and effective by our experimental results.

The rest of this paper is organized as follows: Section 2 discusses related work. Section 3 describes in detail our proposed CRFs Opinion Mining model. We describe in Section 4 experimental results and make a comparison among different methods. In the end, we give our conclusions and our future work in Section 5.

2 Related Work

Recently, many researchers have studied the problem. In Turney al et's work [2], they used pointwise mutual information (PMI) to calculate the average semantic orientation (SO) of extracted phrases for determing documents polarity. Pang al et [4] examined the effectiveness of applying machine learning techniques to the sentiment classification problem with movie review data. Hatzivassiloglou and Wiebe [6] studied the effects of dynamic adjectives, semantically oriented adjectives, and gradable adjectives on a simple subjectivity classifier, and proposed a novel trainable method that statistically combines two indicators of gradability. Wiebe and Riloff [7] proposed a system called OpinionFinder that performs subjectivity analysis, automatically identifying when opinions, sentiments, speculations, and other private states are present in text. Das and Chen [9] studied document level sentiment polarity classification on financial documents. All these works are related to sentiment classification, it uses the sentiment to represent reviewer's whole opinion and does not find what the reviewer liked and disliked. For example, an negative sentiment on an object does not mean that the reviewer dislike everything and also an positive sentiment does not mean that the reviewer like everything.

To solve this problem, some researchers try to mine and extract opinions on the feature level as well as the sentence level. Hu and Liu [10] proposed a feature-based opinion summarization system capturing high frequency feature words by using association rules under a statistical framework. It mines only the features of the product that customers have expressed their opinions on and a summary is generated by using high frequency feature words (the top ranked features) and ignoring infrequent features. Popescu and Etzioni [3] improved Hu and Liu's work by removing those frequent noun phrases that may not be features. It tries to identify part-of relationship and achieves a better precision but a small drop in recall. Christopher Scaffidi et al [8] presented a new search system called Red Opal which could examine prior customer reviews, identify product features, and score each product on each feature. Red Opal uses these scores to determine which products to show when a user specifies a desired product feature. However, all these work failed to identify infrequent entities effectively. Our approach can address this issue effectively.

Another work we want to mention here is a machine learning system called OpinionMiner which was designed by Weijin et al [1]. It was built under the framework of lexicalized HMMs integrating multiple important linguistic features into automatic learning. it's closely related to our work, but we employ CRFs instead of HMMs to avoid some limitations inherent in HMMs. For example, it can not represent neither distributed hidden state nor complex interaction among labels, it also can not use rich, overlapping feature sets.

3 The Proposed Approach

Fig. 1 gives the architecture overview for our approach, which performs the opinion mining in four main tasks: (1) Pre-work which include crawing raw review data and cleaning; (2) data processing; (3) train the liner-chain CRFs model; (4) Testing.

3.1 CRFs Models

Conditional random fields (CRFs) are conditional probability distributions that factorize according to an undirected model. It could be defined as follows: considering a graph $G = (V, E)$, let $Y = (Y_v)_{v \in V}$, and (X, Y) is a CRF, where X is the set of variables over the observation sequences to be labeled (e.g., a sequence of natural language words which form a sentence), and Y is the set of random variables over the corresponding labeling sequences which obey the Markov property with respect to the graph(e.g., part-of-speech tags for the words sequences). It models $p(y|x)$ globally conditioned on the

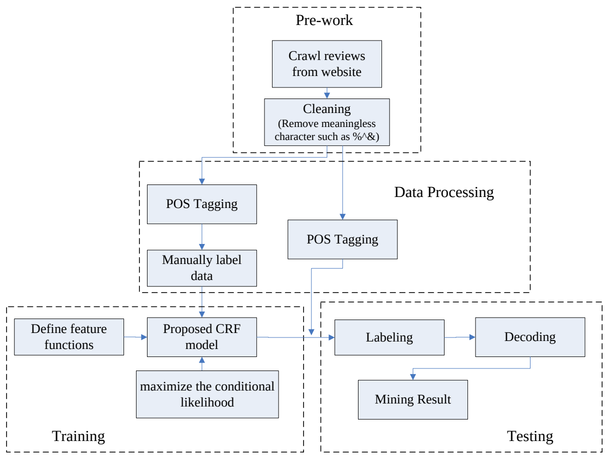


Figure 1: The architecture of proposed system

observation X:

$$p(y|x) = \frac{1}{Z(x)} \prod_{i \in N} \phi_i(y_i, x_i) \tag{1}$$

where $Z(x) = \sum_y \prod_{i \in N} \phi_i(y_i, x_i)$ is a normalization factor over all state sequences for the sequence x. We assume the potentials factorize according to a set of features f_k, as

$$\phi_i(y_i, x_i) = \exp(\sum_k \lambda_k f_k(y_i, x_i)) \tag{2}$$

Given such a model as defined in equation (1), the most probable labeling sequence for an input x is

$$\widehat{Y} = \arg \underset{y}{\mathrm{m}} axp(y|x) \tag{3}$$

3.2 Problem Statement

Our goal was to effectively extract product entities from reviews and identify opinion's polarity. The product entities could be divided into four categories according to [1]: Components, Functions, Features and Opinions. Please notice that the features mentioned here indicate product feature but feature functions for constructing CRFs models. In our work, we use the same classifications of product entities in [1]. Table 1 shows the four categories of entities and their examples. [1]

Table 1: Four entities and their examples

Components:	Physical objects of a product, such as cellphone's LCD
Functions:	Capabilities provided by a product, e.g., movie playback, zoom, automatic fillflash, auto focus
Features:	Properties of components or functions, e.g., color, speed, size, weight
Opinions:	Ideas and thoughts expressed by reviewers on product features, components, functions.

Our work employ three types of tags to represent the words: entity tags, position tags and opinion tags. We use the categories names of a product entity as entity tags. For the word which is not an entity, we use character 'B' to represent it. Usually, an entity could be a single word or a phrase (words chuck). For a phrase entity, we assign a position to each word of this phrase. Any word of a phrase could only be three types of positions: the beginning of the phrase, the middle of the phrase and the end of phrase. Here we use character 'B', 'M' and 'E' as position tags to represent the position of a word in beginning, middle and end of a phrase respectively. Considering each opinion has different orientations and whether it is explicit or implicit, we use character 'P', 'N' to represent Positive, Negative and use "Exp", "Imp" to represent explicit opinion and Implicit opinion. These tags are opinion tags. Combining all these tags, we could tag out any word and its role in a sentence. For example, we label the sentence "The image is good and it is ease of use" from a camera review:

The(B) image(Featue-B) is(B) good(Opinion-B-P-Exp) and(B) it(B) is(B) ease(Feature-B) of(Feature-M) use(Feature-E).

In this sentence, 'image' and 'ease of use' are both features of this camera and 'ease of use' is a phrase, thus we add '-B', '-M' and '-E' to the end of categories tag 'Feature to specify the position of each word in the phrase. 'Good' is a positive explicit opinion expressed on the feature 'image', so its tag is 'Opinion-B-P-Exp'. All other words which are not any category of entity are given the tag 'B'.

In this way, if we know each word's hybrid tag, we could extract the product entities and identify the opinions' orientations. Thus, the task of opining mining and extraction is transformed to a labeling task.

We now describe the model topology and the feature functions used to construct a CRFs opinion mining system. Our problem can be described as follows: given a sequence of words $W = w_1 w_2 w_3 ... w_n$ and corresponding parts of speech $S = s_1 s_2 s_3 ... s_n$, and the objective is to find an appropriate sequence of hybrid tags which maximize the conditional likelihood according to equation (3)

$$\widehat{T} = \arg \underset{T}{m} axp(T|W,S) = \arg \underset{T}{m} ax \prod_{i=1}^{N} p(t_i|W,S,T^{(-i)}) \quad (4)$$

where $T^{(-i)} = \{t_1 t_2 ... t_{i-1} t_{i+1} ... t_N\}$. However, we could see that the hybrid tag in position i depends on all the words $W = w_{1:N}$, part-of-speeches $S = s_{1:N}$ and tags except itself, which makes it very hard for computing in practice as this involves too many parameters. To simply our problem, we employ linear-chain CRFs as an approximation to restrict the relationships within tags. It's graphical structure shown in Figure 2. We could see that in a linear-chain CRF, all the nodes in the graph form a linear chain and each feature involves only two consecutive hidden states. Thus equation (4) could be rewritten as

$$\widehat{T} = \arg \underset{T}{m} axp(T|W,S) = \arg \underset{T}{m} ax \prod_{i=1}^{N} p(t_i|W,S,t_{i-1}) \quad (5)$$

3.3 Feature Functions

From the model above, we can see there are still many parameters need to be considered. To make the model computable, we need to define the relationships among the observation states $W = w_{1:N}$, $S = s_{1:N}$ and hidden states $T = t_{1:N}$ to reduce the unnecessary calculations. Thus, as important components of CRFs, feature functions' definition is crucial to our problem. Let $w_{1:N}$, $s_{1:N}$ be the observations (e.g., words sequence and corresponding parts of speech respectively), $t_{1:N}$ the hidden labels (e.g., hybrid tags). In our case of linear-chain CRF, the general form of a feature function is $f_i(t_{n-1}, t_n, w_{1:N}, s_{1:N}, n)$, which looks at a pair of adjacent states t_{n-1},t_n, the whole input sequence $w_{1:N}$ as well as $s_{1:N}$ and where we are in the sequence. For example, we can define a simple feature function which produces binary values: the returned value is 1 if the current word w_n is "good", the corresponding part-of-speech s_n is "JJ" which means single adjective word and if the current state t_n is "Opinion":

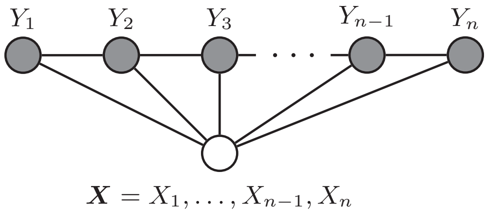


Figure 2: liner-CRFs graphic structure

$$f_i(t_{n-1}, t_n, w_{1:N}, s_{1:N}, n) = \begin{cases} 1 \quad if \ w_n = good, \quad s_n = JJ \\ and \ t_n = Opinion \\ 0 \quad otherwise \end{cases} \quad (6)$$

Combining feature function with equation (1) and equation (2), we have:

$$p(t_{1:N}|w_{1:N}, s_{1:N}) = \frac{1}{Z} \exp(\sum_{n=1}^{N} \sum_{i=1}^{F} \lambda_i f_i(t_{n-1}, t_n, w_{1:N}, s_{1:N}, n)) \quad (7)$$

According to equation (7), the feature function f_i depends on its corresponding weight λ_i. That is if $\lambda_i > 0$, and f_i is active, it will increase the probability of the tag sequence $t_{1:N}$ and if $\lambda_i < 0$, and f_i is inactive, it will decrease the probability of the tag sequence $t_{1:N}$.

What worth mentioned here is another example of feature function, please consider

$$f_i(t_{n-1}, t_n, w_{1:N}, s_{1:N}, n) = \begin{cases} 1 \quad if \ w_n = good, \quad s_{n+1} = NN \\ and \ t_n = Opinion \\ 0 \quad otherwise \end{cases} \quad (8)$$

For the phrase ”good image”, the features in equation (6) and (8) are both active if the current word is ”good”. It boosts up the belief of $t_i = Opinion$ to both λ. This is an example of overlapping features which HMMs can not do: HMMs cannot consider the next word, nor can they use overlapping features.

Thus in addition to employing linear-chain CRFs to simplify the relations within hidden states T, we also define several different types of feature functions to specify state-transition structures among W, S and T. As different state transition features can be defined to form different markov-order structures, our different state transitions features are based on different markov order for different classes of features. Here we will describe first order features in detail:

1. The assignment of current tag t_i is supposed to only depend on the current word. The feature functions are represented as $f(t_i, w_i)$.

2. The assignment of current tag t_i is supposed to only depend on the current part-of-speech. The feature functions are represented as $f(t_i, s_i)$.

3. The assignment of current tag t_i is supposed to depend on both the current word and current part-of-speech. The feature functions are represented as $f(t_i, s_i, w_i)$.

All the three types of feature functions are first-order, of which the inputs are examined in the context of the current state only. There are no separate parameters for state transitions at all. We also define first-order+transitions features and second-order features which are examined in the context of the current and previous states. We did not define third-order or higher-order features because they create more data sparse problem and require more memory in training. Table 2 shows all the features we defined in our model.

3.4 CRFs Training

After the graph and feature functions are defined, the model is fixed. Thus the purpose of training is find out all the values of $\lambda_{1:N}$. Usually One may set $\lambda_{1:N}$ by domain knowledge, however, in our case, we would learn $\lambda_{1:N}$ from data as there is little knowledge available for us. The fully labeled data sequence is $\{(w^{(1)}, s^{(1)}, t^{(1)}), ..., (w^{(m)}, s^{(m)}, t^{(m)})\}$, where $w^{(1)} = w^{(1)}_{1:N_1}$ the first words sequence, $s^{(1)} = s^{(1)}_{1:N_1}$ the first part-of-speech sequence, $t^{(1)} = t^{(1)}_{1:N_1}$ the first tags sequence respectively and so on. Since CRFs define the conditional probability $p(t|w, s)$, the appropriate objective for parameter learning is to maximize the conditional likelihood of the training data

$$\sum_{j=1}^{m} \log p(\mathbf{t}^{(j)}|\mathbf{w}^{(j)}, \mathbf{s}^{(j)}) \tag{9}$$

To avoid over-fitting, log-likelihood is usually penalized by some prior distribution over the parameters. A commonly used prior is a zero-mean Gaussian. If $\lambda \sim N(0, \sigma^2)$, the objective becomes

$$\sum_{j=1}^{m} \log p(t^{(j)}|w^{(j)}, s^{(j)}) - \sum_{i}^{F} \frac{\lambda_i^2}{2\sigma^2} \tag{10}$$

The objective is concave, so the λ have a unique set of global optimal values. We learn parameters by computing the gradient of the objective function, and using the gradient in an optimization algorithm like L-BFGS. The

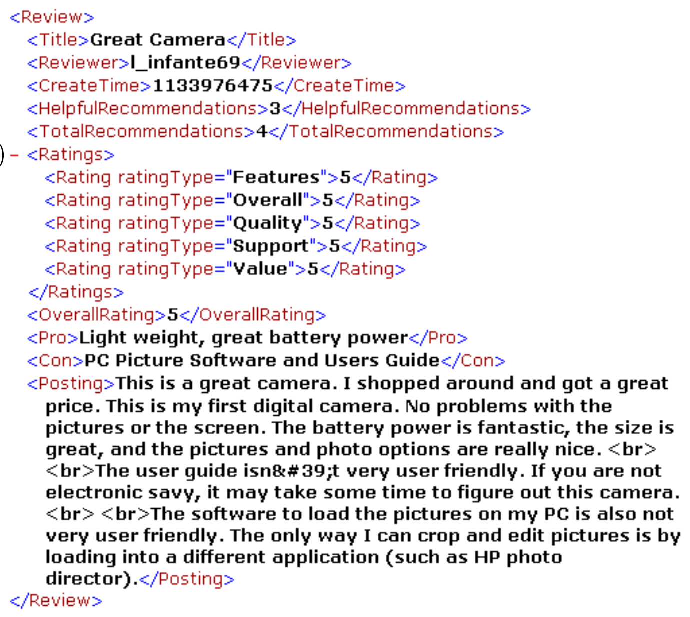
Table 2: The feature functions types and their expressions

Feature type	Expressions
First-oder	$f(t_i, w_i)$, $f(t_i, s_i)$, $f(t_i, s_i, w_i)$
First-order+transitions:	$f(t_i, w_i)f(t_i, t_{i-1})$, $f(t_i, s_i)f(t_i, t_{i-1})$, $f(t_i, s_i, w_i)f(t_i, t_{i-1})$
Second-order:	$f(t_i, t_{i-1}, w_i,)$, $f(t_i, t_{i-1}, s_i)$, $f(t_i, t_{i-1}, s_i, w_i)$

gradient of the objective function is computed as follows:

$$\frac{\partial}{\partial\lambda_k}\sum_{j=1}^{m}\log p(t^{(j)}|w^{(j)},s^{(j)}) - \sum_{i}^{F}\frac{\lambda_i^2}{2\sigma^2}$$
$$= \frac{\partial}{\partial\lambda_k}\sum_{j=1}^{m}(\sum_n\sum_i \lambda_i f_i(t_{n-1},t_n,w_{1:N},s_{1:N},n) - \log T^{(j)}) -$$
$$= \sum_{j=1}^{m}\sum_n f_k(t_{n-1},t_n,w_{1:N},s_{1:N},n)$$
$$- \sum_{j-1}^{m}\sum_n E_{t'_{n-1},t'_n}[f_k(t'_{n-1},t'_n,w_{1:N},s_{1:N},n)] - \frac{\lambda_k}{\sigma^2} \qquad (11)$$

In equation (10), The first term is the empirical count of feature i in the training data, the second term is the expected count of this feature under the current trained model and the third term is generated by the prior, we could ignore it. Hence, the derivative measures the difference between the empirical count and the expected count of a feature under the current model. Suppose that in the training data a feature f_k appears A times, while under the current model, the expected count of f_k is B. When $|A| = |B|$, the derivative is zero. Therefore, training can be thought of as finding λs that match the two counts.



```
<Review>
  <Title>Great Camera</Title>
  <Reviewer>l_infante69</Reviewer>
  <CreateTime>1133976475</CreateTime>
  <HelpfulRecommendations>3</HelpfulRecommendations>
  <TotalRecommendations>4</TotalRecommendations>
  <Ratings>
    <Rating ratingType="Features">5</Rating>
    <Rating ratingType="Overall">5</Rating>
    <Rating ratingType="Quality">5</Rating>
    <Rating ratingType="Support">5</Rating>
    <Rating ratingType="Value">5</Rating>
  </Ratings>
  <OverallRating>5</OverallRating>
  <Pro>Light weight, great battery power</Pro>
  <Con>PC Picture Software and Users Guide</Con>
  <Posting>This is a great camera. I shopped around and got a great
    price. This is my first digital camera. No problems with the
    pictures or the screen. The battery power is fantastic, the size is
    great, and the pictures and photo options are really nice. <br>
    <br>The user guide isn&#39;t very user friendly. If you are not
    electronic savy, it may take some time to figure out this camera.
    <br> <br>The software to load the pictures on my PC is also not
    very user friendly. The only way I can crop and edit pictures is by
    loading into a different application (such as HP photo
    director).</Posting>
</Review>
```

Figure 3: Example of one full review

4 Evaluation

In this section, we present a detailed comparison of recall, precision and F-score of extracted entities, opinions and their polarity. Recall is $\frac{|C \cap P|}{|C|}$ and Precision is $\frac{|C \cap P|}{|P|}$, where C and P are the sets of correct and predicted hybrid tags, respectively. F score is the harmonic mean of precision and recall, $\frac{2RP}{R+P}$. Three methods would be evaluated: the baseline method, the L-HMM model in [1] and our proposed CRF model. We crawl product reviews from Yahoo shopping as datasets for the evaluation. The review format is semi-structured and it consists of several parts: Title, Reviewer, Ratings, Pro, Con, Posting, etc. Fig. 3 shows a whole review format.

In our work, we only use the Posting part since it is free text which could express costumer's opinion fully. After some cleaning work such as removing meaningless characters, we then use the LBJPOS tool by natural language processing group of University of Illinois, Urbana-Champaign [16] to produce the part-of-speech tag for each word in every review. We also use the corpus of Liu and Hu's as datasets for evaluation. As their data only tag the entities for each product Thus we need to retag their data. We randomly choose 476 reviews in total for three cameras and one cellphone and manually label all of them using the hybrid tags. All tagged data are then divided into 4 four sets to perform a 4-fold crossvalidation.The whole process is as the following: a single set is retained as the validation data for testing, and the remaining 3 sets are used as training data. The cross-validation process is then repeated 4 times, with each of the 4 subsamples used exactly once as the validation data. The 4 results then would be averaged to produce a single estimation.

4.1 Rule-base Method

Motivated by [2], we design a straightforward rule-based method as the baseline system for comparison. The first step is performing Part-of-Speech task, one example of POS tagging result is here:

(PRP I) (VBD used) (NNP Olympus) (IN before) (, ,) (VBG comparing) (TO to) (NN canon) (, ,) (PRP it)

(VBD was) (DT a) (NN toy) (, ,) (NNP S3) (VBZ IS) (VBZ is) (RB not) (DT a) (JJ professional) (NN camera) (, ,) (CC but) (RB almost) (VBZ has) (NN everything) (PRP you) (VBP need) (, ,) (TO to) (PRP me) (, ,) (PRP it) (: ;) (VBZ s) (JJ professional) (, ,) (NN 6mb) (VBZ is) (JJ great) (, ,) (PRP I) (VB don) (: ;) (NN t) (VBP need) (DT a) (NN 10mb) (, ,) (VB zoom) (VBZ is) (JJ outstanding) (, ,) (NN night) (NNS snapshots) (VBP are) (RB really) (JJ good) (. .)

The baseline system then will extract product entities, opinions and their orientations.

4.1.1 Product Entities Extraction

This system will apply for some basic rules to extract product entities: 1. a single noun which follows an adjective word or consecutive adjective words will be seen as a product entity. (JJ + NN or JJ + JJ + NN) 2. Any single noun word connects an adjective word by a linking verb will be seen as a product entity. (NN + VBZ +JJ) 3. Any consecutive noun words which appear in the position described above will be seen as a product entity phrase.

4.1.2 Opinion Extraction and orientations determination

Rules for determining opinion word and their orientations are described as follow: 4. All the adjective words appearing in rule 1 and 2 will be seen as opinion entities. 5. The orientations of opinion entities will be determined by a lextion which indicate the polarity of over 8000 adjective words. This is made by Pang et al's work [15].

4.2 L-HMMs Approach

The work in [1] integrated linguistic features such as part-of-speech and lexical patterns into HMMs. It also aims to find out an appropriate sequence of hybrid tags $T = t_1t_2t_3...t_n$ that maximize the conditional probability described in equation (4). They rewrite equation (4) as

$$\widehat{T} = \arg\underset{T}{m}axp(W,S|T)p(T) = \arg\underset{T}{m}axp(S|T)$$
$$p(W|T,S)p(T)$$

Three hypotheses are then made for simplifying the problem: (1) the assignment of current tag is supposed to depend not only on its previous tag but also previous J words. (2) The appearance of current word is assumed to depend not only on the current tag, current POS, but also the previous K words. (3) The appearance of current POS is supposed to depend both on the current tag and previous L words and J=K=L. Then the objective is

$$\arg\underset{T}{m}ax\prod_{i=1}^{N}\left\{\begin{array}{l} p(s_i|w_{i-1},t_i)\times \\ p(w_i|w_{i-1},s_i,t_i)\times \\ p(t_i|w_{i-1},t_{i-1}) \end{array}\right\}$$

Maximum Likelihood Estimation (MLE) is used to estimate the parameters. Some techniques are also used in this approach such as information propagation using entitys synonyms, antonyms and related words, token transformations and etc.

4.3 Experimental Results and Discussion

Table 3 shows the evaluation results of the 4-fold cross-validation based on the recall, precision and F-score of extracted entities, opinions and their polarity. In the table, line 1 lists each method we want to evaluate: the baseline system, the L-HHMs model with POS tagging proposed by [1] and our approach. the following lines give the recall, precision and F-scores of the four kinds of product entities.

We observe that CRFs increase the performance on nearly all the four entities. CRFs performs better than L-HMMs, with the overall word precision improved from 83.9% to 90.0% and the F-score improved from 77.1% to 84.3%. There are two major reasons that lead to its results. First of all, HMMs assume that each feature is generated independently by some hidden process, that's to say, only tags can affect each other and ignore the underlying relationships among the words and POS tags. But, this is in general not the case in a labeling task, the words and POS tags also play an important role in predicting the hidden states. Secondly, HMMs does not model the overlapping features. We also achieved a high recall in the proposed approach. The average recall of our method for the four entities was improved from 72.0% to 79.8%. This is promising as the recall rate for labeling task is easily affected by errors in tagging. For example, if the tags "Opinion-B-P-Exp, Opinion-M-P-Exp, Opinion-M-P-Exp, Opinion-E-P-Exp" is labled as "Opinion-B-P-

Exp, Opinion-E-P-Exp, Opinion-M-P-Exp, Opinion-E-P-Exp", the labeling accuracy is 75%, but recall is 0.

CRFs also perform better than Baseline approach, yielding new state of the art performance on this task. After a close inspection of the terms generated by baseline system, we see that there are lots of errors in entities such as it takes "seller" as a product feature because it's a noun word. Our method can avoid this problem fully.

We also conduct a comparison using different datasets with our approach (Hu's 238 documents and yahoo shopping's 238 documents), Fig. 4 shows the precision, recall and F-scores of four entities. We can see in all three criteria, the corresponding values generated by different datasets are close. The biggest absolute value of their difference is the precision of functions, but it is only 3%. This shows our method achieves a stable performance with different datasets which proves the robustness of our approach.

Our work could also identify some infrequent entities since the overlapping feature functions can increase the impact of infrequent entities. This greatly avoid the affect of the low counts of such entities. For example, the entity "ISO" only appears once in our data, which will be omitted in other approaches. In our method, both function $f(t_i, s_i, w_i)$ and $f(t_i, w_i)$ would model this feature which makes f_1 and f_2 can be both active for the tagging.

In this paper, we also extracted the potential non-noun product entities, such as "adjust" and non-adjective opinions, such as "strongly recommended". These non-noun entities and non-adjective opinions were ignored by previously proposed approaches which were based on the assumption that product entities must be noun or noun phrases and the opinions must be adjective words. Our system can well identify these overlooked product entity information.

5 Conclusion

In this paper, we proposed a novel linear-chain CRFs-based learning approach for opinion mining and extraction. Contrasting to HMMs which assume that each feature is generated independently by some hidden process, CRFs can handle non-independent input features, which can be beneficial in complex domains. The experiment results showed the effectiveness of the proposed approach.

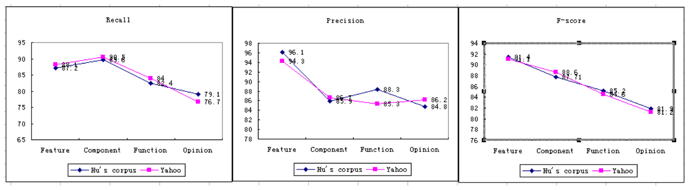


Figure 4: The Precision, Recall and F-scores of two datasets

Table 3: 4-fold crossvalidation results(R: recall, P:precision, F: F-score)

Methods		Baseline	L-HMM+POS	CRF+POS
Feature Entities(%)	R	-	78.6	81.8
	P	-	82.2	93.5
	F	-	80.4	87.2
Component Entities(%)	R	-	96.5	91.8
	P	-	95.3	98.7
	F	-	96.0	95.1
Function Entities(%)	R	-	58.9	80.4
	P	-	81.1	83.7
	F	-	68.2	82.0
Opinion Entities(%)	R	-	53.7	65.3
	P	-	76.9	84.2
	F	-	63.2	73.5
All Entities(%)	R	27.2	72.0	79.8
	P	24.3	83.9	90.0
	F	25.7	77.1	84.3

In our future work, we would consider incorporating richer features to improve our model. Due to the complicacy of human natural language, some long-distance features of the input are beyond our hypotheses. Thus, if some rich features are included, we could further employ automatic feature induction techniques to find non-obvious conjunctions of features that may improve performance. Moreover, the mining result of our system can also be used for us to develop some feasible web applications such as recommender systems. We will also consider training our proposed model to assign a score for each entity based on the degree of the opinion word which describe it.

References

[1] Jin, W., Ho, H., and Srihari, R. K.: OpinionMiner: a novel machine learning system for web opinion mining and extraction. In: 15th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining, pp. 1195–1204. ACM, New York, NY, USA (2009)

[2] Turney, Peter D. : Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In: 15th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining, pp. 417–424. Association for Computational Linguistics, Morristown, NJ, USA (2002)

[3] Popescu, A. and Etzioni, O. : Extracting Product Features and Opinions from Reviews. In: Conference on Empirical Methods in Natural Language Processing, pp. 339-346. Association for Computational Linguistics, Morristown, NJ, USA (2005)

[4] Pang, B. and Lee, L. and Vaithyanathan, Shivakumar. : Thumbs up?: sentiment classification using machine learning techniques. In: the ACL-02 conference on Empirical methods in natural language processing, pp. 79–86. Association for Computational Linguistics, Morristown, NJ, USA (2002)

[5] Dave, K. and Lawrence, S. and Pennock, David M. : Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In: the 12th international conference on World Wide Web, pp. 519–528. ACM, New York, NY, USA (2002)

[6] Hatzivassiloglou, Vasileios and Wiebe, Janyce M. : Effects of adjective orientation and gradability on sentence subjectivity. In: the 18th conference on Computational linguistics, pp. 299–305. Association for Computational Linguistics, Morristown, NJ, USA (2000)

[7] Wilson, Theresa and Hoffmann, Paul and Somasundaran, Swapna and Kessler, Jason and Wiebe, Janyce and Choi, Yejin and Cardie, Claire and Riloff, Ellen and Patwardhan, Siddharth. : OpinionFinder: a system for subjectivity analysis. In: HLT/EMNLP on Interactive Demonstrations, pp. 34–35. Association for Computational Linguistics, Morristown, NJ, USA (2005)

[8] Scaffidi, Christopher and Bierhoff, Kevin and Chang, Eric and Felker, Mikhael and Ng, Herman and Jin, Chun. : Red Opal: product-feature scoring from reviews. In: the 8th ACM conference on Electronic commerce, pp. 182–191. ACM, New York, NY, USA (2007)

[9] Das, Sanjiv and Mike Chen. : Yahoo! for Amazon: Extracting market sentiment from stock message boards. In: Asia Pacific Finance Association Annual Conf, (2001)

[10] Hu, M. and Liu, B. : Mining and Summarizing Customer Reviews. In: 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 168-177. ACM, New York, NY, USA (2004)

[11] John Lafferty, Andrew McCallum, and Fernando Pereira. In: International Conference on Machine Learning, pp. 282–289 ACM, New York, NY, USA (2001)

[12] Fuchun P. and Andrew McCallum. : Accurate information extraction from research papers using conditional random fields. In: Human Language Technology Conference and North American Chapter of the Association for Computational Linguistics, Association for Computational Linguistics, Morristown, NJ, USA (2004)

[13] Fei S. and Fernando Pereira. : Shallow parsing with conditional random fields. In: the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, pp. 134 - 141. Association for Computational Linguistics, Morristown, NJ, USA (2003)

[14] McCallum, A. : Efficiently inducing features of conditional random fields. In: the conference on Conference on Uncertainty in Artificial Intelligence, (2003)

[15] `http://www.cs.cornell.edu/People/pabo/movie-review-data/review_polarity.tar.gz`

[16] `http://l2r.cs.uiuc.edu/~cogcomp/software.php`