# A Linear-Chain CRF-Based Learning Approach for Web Opinion Mining

Luole Qi and Li Chen

Department of Computer Science,
Hong Kong Baptist University,
Hong Kong, China
{llqi,lichen}@comp.hkbu.edu.hk

**Abstract.** The task of opinion mining from product reviews is to extract the product entities and determine whether the opinions on the entities 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. 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 evaluated our approach base on three criteria: recall, precision and F-score for extracted entities, opinions and their polarities. Compared to other methods, our approach was proven more effective for accomplishing opinion mining tasks.

**Keywords:** Conditional Random Field (CRFs), Web Opinion Mining, Feature Function.

## 1 Introduction

User-generated reviews have been increasingly regarded useful in business, education, and e-commerce, since they contain valuable opinions originated from the user's experiences. For instance, in e-commerce sites, customers can assess a product's quality by reading other customers' reviews to the product, 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.com, allow users to post their opinions freely. The reviews' number has in fact reached to more than thousands in these large websites, which hence poses a challenge for a potential customer to go over all of them.

To resolve the problem, researchers have done some work on Web opinion mining which aims to discover the essential information from reviews and then present to users. Previous works have mainly adopted rule-based techniques [2] and statistic methods [10]. Lately, a new learning approach based on a sequence model named Hidden Markov model (HMMs) was proposed and proved more

L. Chen, P. Triantafillou, and T. Suel (Eds.): WISE 2010, LNCS 6488, pp. 128–141, 2010.

effective than previous works. However, the HMMs-based method is still limited because it is difficult to model arbitrary, dependent features of the input word sequence.

To address the limitation, in this paper, we have particularly studied the Conditional Random Field (CRFs) [11], because it is a discriminative, undirected graphical model that can potentially model the overlapping, dependent features. In prior work on natural language processing, CRFs have been demonstrated to out-perform HMMs [12][13]. Thus, motivated by the earlier findings, we propose a linear-chain CRF-based learning approach to mine and extract opinions from product reviews on the Web. Specifically, our objective is to answer the following questions: (1) how to define feature functions to construct and restrict our linear-chain CRFs model? (2) How to choose criteria for training a specific model from the manually labeled data? (3) How to automatically extract product entities and identify their associated opinion polarities with our trained model? In the experiment, we evaluated the model on three evaluation metrics: recall, precision and F-score, on extracted entities and opinions. The experimental results proved the higher accuracy of our proposed approach in accomplishing the web opinion mining task, in comparison with the related rule-based and HMMs-based approaches.

To highlight our contributions, we have demonstrated that the linear-chain CRF-based learning method can perform better than L-HMMs approach, in terms of integrating linguistic features for opinion mining. The feature functions defined in our work have been also indicated robust and effective for the model construction.

The rest of this paper is therefore organized as follows: we will discuss related work in Section 2 and describe in detail our proposed CRF-based opinion mining learning method in Section 3. In Section 4, we will present experiment design and results. Finally, we will conclude our work and give its future directions.

## 2 Related Work

Thus far, many researchers have attempted to extract opinion values from user reviews (or called documents in some literatures). At the document level, Turney et al. [2] used pointwise mutual information (PMI) to calculate the average semantic orientation (SO) of extracted phrases for determining the document's polarity. Pang et al [4] examined the effectiveness of applying machine learning techniques to address the sentiment classification problem for movie review data. Hatzivassiloglou and Wiebe [6] studied the effect of dynamic adjectives, semantically oriented adjectives, and gradable adjectives on a simple subjectivity classifier, and proposed a trainable method that statistically combines two indicators of grad ability. Wiebe and Riloff [7] proposed a system called OpinionFinder that automatically identifies when opinions, sentiments, speculations, and other private states are present in text, via the subjectivity analysis. Das and Chen [9] studied sentimental classification for financial documents. However, although the above works are all related to sentiment classification, they

just use the sentiment to represent a reviewer's overall opinion and do not find what features the reviewer actually liked and disliked. For example, an overall negative sentiment on an object does not mean that the reviewer dislikes every aspect of the object, which can indeed only indicate that the average opinion as summarized from this review is negative.

To in-depth discover a reviewer's opinions on almost every aspect that she mentioned in the text, some researchers have tried to mine and extract opinions at the feature level. Hu and Liu [10] proposed a feature-based opinion summarization system that captures highly frequent feature words by using association rules under a statistical framework. It extracts the features of a product that customers have expressed their opinions on, and then concludes with an opinion score for each frequent feature (one of the top ranked features) while ignoring infrequent features. Popescu and Etzioni [3] improved Hu and Liu's work by removing frequent noun phrases that may not be real features. Their method can identify part-of relationship and achieve a better precision, but a small drop in recall. Scaffidi et al [8] presented a new search system called Red Opal that examined prior customer reviews, identified product features, and then scored each product on each feature. Red Opal used these scores to determine which products to be returned when a user specifies a desired product feature. However, the limitation of these works is that they failed to identify infrequent entities effectively.

The work that is most similar to our focus is a supervised learning system called OpinionMiner [1]. It was built under the framework of lexicalized HMMs that integrates multiple important linguistic features into an automatic learning process. The difference between our method and it is that we employ CRFs in order to avoid some limitations that are inherent in HMMs. Indeed, it can not represent distributed hidden states and complex interactions among labels. It can neither involve rich, overlapping feature sets. Miao et al [15] have recently also attempted to adopt CRFs to extract product features and opinions, but they did not identify sentiment orientation using CRFs and did not make a comparison across different methods.

## 3 The Proposed Approach

Fig. 1 gives the architecture overview of our approach. It can be divided into four major steps: (1) pre-processing which includes crawling raw review data and cleaning; (2) review data tagging; (3) training the liner-chain CRFs model; and (4) applying the model to obtain opinions from new review data.

### 3.1 CRFs Learning Model

Conditional random fields (CRFs) are conditional probability distributions on an undirected graph model [11]. It can be defined as follows: considering a graph $G = (V, E)$ for which $V$ indicates the nodes and $E$ indicates the edges. Let $Y = (Y_v)_{v \in V}$, and $(X, Y)$ is a CRF, where $X$ is the set of variables over the

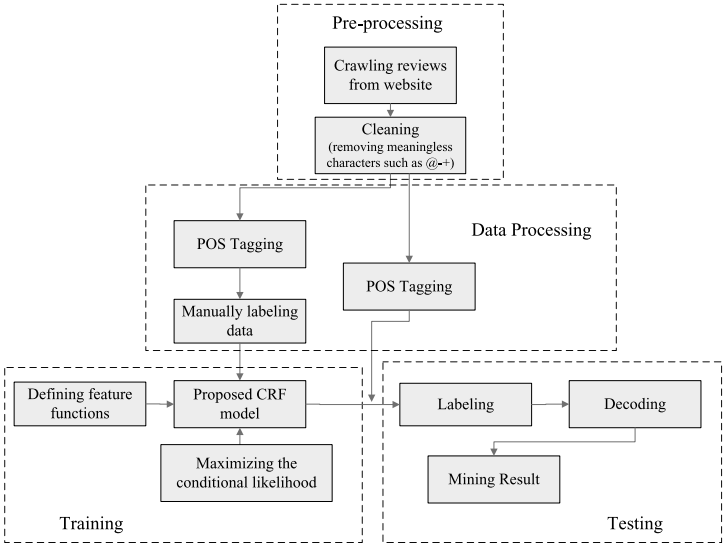


**Fig. 1.** The architecture of our proposed system

observation sequences to be labeled (e.g., a sequence of textual words that form a sentence), and Y is the set of random variables over the corresponding sequence. The $(X, Y)$ obeys the Markov property with respect to the graph(e.g., part-of-speech tags for the words' sequence). Formally, the model defines $p(y|x)$ which is globally conditioned on the observation of 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 $x$. The potentials are normally factorized on 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 the model defined in equation (1), the most probable labeling sequence for an input $x$ is

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

### 3.2 Problem Statement

Our goal was to extract product entities from reviews which also include the opinion polarities. The product entities can be divided into four categories according to [1]: Components, Functions, Features and Opinions. Please notice

that the features mentioned here refer to the product's features (e.g., the camera's size, weight) which are different from the meaning of features in the feature functions for constructing CRFs models (see the definition of feature functions later). Table 1 shows the four categories of entities and their examples. In our work, we follow this classification scheme.

**Table 1.** Four types of product entities and their examples (referred to [1])

| | |
|---|---|
| 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 or functions. |

We employ three types of tags to define each word: entity tag, position tag and opinion tag. We use the category name of a product entity to be the entity tag. As for a word which is not an entity, we use the character 'B' to represent it. Usually, an entity could be a single word or a phrase. For the phrase entity, we assign a position to each word in the phrase. Any word of a phrase has three possible positions: the beginning of the phrase, the middle of the phrase and the end of phrase. We use characters 'B', 'M' and 'E' as position tags to respectively indicate the three positions. As for "opinion" entity, we further use characters 'P' and 'N' to respectively represent Positive opinion and Negative opinion polarity, and use "Exp" and "Imp" to respectively indicate Explicit opinion and Implicit opinion. Here, explicit opinion means the user expresses opinion in the review explicitly and implicit opinion means the opinion needs to be induced from the review. These tags are called opinion tags. Thus with all of above defined tags, we can tag any word and its role in a sentence. For example, the sentence "The image is good and its ease of use is satisfying" from a camera review is labeled as:

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

In this sentence, 'image' and 'ease of use' are both features of the camera and 'ease of use' is a phrase, so we add '-B', '-M' and '-E' 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' (such tag combination is also called hybrid tags in [1]). Other words which do not belong to any entity categories are given the tag 'B'.

Therefore, when we get each word's tag(s), we could obtain the product entity it refers to and identify the opinion orientation if it is an "opinion" entity. In

this way, the task of opining mining can be transformed to an automatic labeling task. The problem can be then formualized as: given a sequence of words $W = w_1 w_2 w_3 ... w_n$ and its corresponding parts of speech $S = s_1 s_2 s_3 ... s_n$, the objective is to find an appropriate sequence of tags which can maximize the conditional likelihood according to equation (3).

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

In equation (4), $T^{(-i)} = \{t_1 t_2 ... t_{i-1} t_{i+1} ... t_N\}$ (which are tags in our case, and called hidden states in the general concept). From this equation, we can see that the tag of word at position $i$ depends on all the words $W = w_{1:N}$, part-of-speech $S = s_{1:N}$ and tags. Unfortunately it is very hard to compute with this equation as it involves too many parameters. To reduce the complexity, we employ linear-chain CRFs as an approximation to restrict the relationship among tags. It is a graphic structure as shown in Figure 2 (in the figure, $Y$ forms a simple first-order chain). In the linear-chain CRF, all the nodes in the graph form a linear chain and each feature involves only two consecutive hidden states. Equation (4) can be hence rewritten as

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

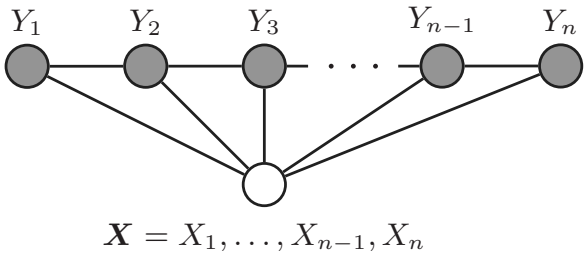


**Fig. 2.** Liner-CRFs graphic structure

### 3.3 Feature Functions

From the model above, we can see there are still many parameters to be processed. To make the model more computable, we need to define the relationships among the observation states $W = w_{1:N}$, $S = s_{1:N}$ and hidden states $T = t_{1:N}$ so as to reduce the unnecessary calculations. Thus, behaving as the important constructs of CRFs, feature functions are crucial to resolve our problem. Let $w_{1:N}$, $s_{1:N}$ be the observations (i.e., words' sequence and their corresponding parts of

speech), $t_{1:N}$ the hidden labels (i.e., tags). In our case of linear-chain CRF (see equation (5)), the general form of a feature function is $f_i(t_{j-1}, t_j, w_{1:N}, s_{1:N}, n)$, which looks at a pair of adjacent states $t_{j-1}$, $t_j$, the whole input sequence $w_{1:N}$ as well as $s_{1:N}$ and the current word's position. For example, we can define a simple feature function which produces binary value: the returned value is 1 if the current word $w_j$ is "good", the corresponding part-of-speech $s_j$ is "JJ" (which means single adjective word) and the current state $t_j$ is "Opinion":

$$f_i(t_{j-1}, t_j, w_{1:N}, s_{1:N}, j) = \begin{cases} 1 \; if \; w_j = good, \;\; s_j = JJ \; and \; t_j = Opinion \\ 0 \; otherwise \end{cases} \tag{6}$$

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

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

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

Another example of feature function can be like:

$$f_i(t_{j-1}, t_j, w_{1:N}, s_{1:N}, j) = \begin{cases} 1 \; if \; w_j = good, \;\; s_{j+1} = NN \; and \; t_j = Opinion \\ 0 \; otherwise \end{cases} \tag{8}$$

In this case, if the current word is "good" such as in the phrase "good image", the feature function in equation (6) and (8) will be both active. This is an example of overlapping features which HMMs can not address. In fact, HMMs cannot consider the next word, nor can they use overlapping features.

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 $T$. The different state transition features are based on different markov orders for different classes of features. Here we define the first order features:

1. The assignment of current tag $t_j$ is supposed to only depend on the current word. The feature function is represented as $f(t_j, w_j)$.
2. The assignment of current tag $t_j$ is supposed to only depend on the current part-of-speech. The feature function is represented as $f(t_j, s_j)$.
3. The assignment of current tag $t_j$ is supposed to depend on both the current word and the current part-of-speech. The feature function is represented as $f(t_j, s_j, w_j)$.

All the three types of feature functions are first-order, with which the inputs are examined in the context of the current state only. We also define first-order+transition features and second-order features which are examined in the context of both the current state and previous states. We do not define third-order or higher-order features because they create data sparse problem and require more memory during training. Table 2 shows all the feature functions we have defined in our model.

**Table 2.** The feature function 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 | $f(t_i, t_{i-1}, w_i, ),\ f(t_i, t_{i-1}, s_i),\ f(t_i, t_{i-1}, s_i, w_i)$ |

### 3.4 CRFs Training

After the graph and feature functions are defined, the model is fixed. The purpose of training is then to identify all the values of $\lambda_{1:N}$. Usually one may set $\lambda_{1:N}$ according to the domain knowledge. However, in our case, we learn $\lambda_{1:N}$ from training data. The fully labeled review data is $\{(w^{(1)}, s^{(1)}, t^{(1)}), ..., (w^{(M)}, s^{(M)}, t^{(M)})\}$, where $w^{(i)} = w^{(i)}_{1:N_i}$ (the $i$th words sequence), $s^{(i)} = s^{(i)}_{1:N_i}$ (the $i$th part-of-speech sequence), $t^{(i)} = t^{(i)}_{1:N_i}$ (the $i$th tags sequence) respectively. Given that in CRFs, we defined the conditional probability $p(t|w, s)$, the aim of parameter learning is to maximize the conditional likelihood based on 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 distributions over the parameters. A commonly used distribution is a zero-mean Gaussian. If $\lambda \sim N(0, \sigma^2)$, the equation of (9) 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 equation is concave, so $\lambda$ has a unique set of global optimal values. We learn parameters by computing the gradient of the objective function, and use the gradient in an optimization algorithm called Limited memory BFGS (L-BFGS).

The gradient of the objective function is formally computed as follows:

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

In equation (11), 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 distribution. 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, the training process is to find $\lambda$s that match the two counts.

## 4 Experiment

In this section, we present the measurements of recall, precision and F-score with our approach. 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 tags, respectively. F score is the harmonic mean of precision and recall, $\frac{2RP}{R+P}$. In the experiment, two related methods were compared to our method: the rule-based method as the baseline and the L-HMM model as described [1] We used two datasets of product reviews: one was crawled from Yahoo shopping, and another was the sharable corpus from Liu and Hu's work [10]. For example, Figure 3 gives one digital camera's user review (in XML format) crawled from Yahoo Shopping site, for which we mainly focused on the "Posting" part as it provides the user-generated textual comments.

We finally collected 476 reviews in total for three cameras and one cellphone, and manually labeled them by using the tags as defined in Section 3.2. After the pre-processing to remove meaningless characters (e.g., @-+), we applied the LBJPOS tool [17] to produce the part-of-speech tag for each word in every review. All tagged data were then divided into 4 four sets to perform 4-fold cross-validation: one set was retained as the validation data for testing, and the other 3 sets were used as training data. The cross-validation process was repeated four times, and every time one set was used as the validation data. Afterwards, the results were averaged to produce precision, recall and F score.

### 4.1 Compared Methods in the Experiment

**Rule-base Method.** Motivated by [2], we designed a rule-based method as the baseline system for comparison. The first step was performing Part-of-Speech (POS) task. One example of POS result is:

```
<Review>
  <Title>Great Camera</Title>
  <Reviewer>I_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>
```

**Fig. 3.** An example of one digital camera's user review in XML format. The <Posting>... </Posting>part gives the textual comment that was emphasized in our algorithm

(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) (..)

In the example, each word gets a tag of POS such as NN (noun word), JJ (adjective word) etc. We then applied several basic rules to extract objective product entities (i.e., components, functions and features as defined in Table 1).

1) One rule is that a single noun that follows an adjective word or consecutive adjective words will be regarded as a product entity, such as JJ + NN or JJ.

2) Any single noun word that connects an adjective word to a verb will be taken as a product entity, such as NN + VBZ +JJ.

3) Any consecutive noun words that appear at the position described in 1) or 2) will be taken as a product entity phrase.

The three rules were actually derived from observations obtained in [10][1].

As for opinion words, the adjective words that appear in rules 1 and 2 will be opinion entities, and their sentimental orientation was determined by a lexicon with polarities for over 8000 adjective words [16].

**L-HMMs Method.** The work in [1] integrated linguistic features such as part-of-speech results and lexical patterns into HMMs. Their aim was to maximize the conditional probability as defined in:

$$\widehat{T} = \arg\max_{T} axp(W, S|T)p(T) = \arg\max_{T} axp(S|T)p(W|T, S)p(T)$$
$$= \arg\max_{T} ax \prod_{i=1}^{N} \left\{ \begin{array}{l} p(s_i|w_1...w_{i-1}, s_1...s_{i-1}, t_1...t_{i-1}t_i)\times \\ p(w_i|w_1...w_{i-1}, s_1...s_{i-1}s_i, t_1...t_{i-1}t_i)\times \\ p(t_i|w_1...w_{i-1}, s_1...s_{i-1}, t_1...t_{i-1}) \end{array} \right\}$$

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

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

Maximum Likelihood Estimation (MLE) was used here to estimate the parameters. Other techniques were also used in this approach, including information propagation using entity synonyms, antonyms and related words, and token transformations, in order to in order to lead to an accurate extraction.

### 4.2 Experimental Results and Discussion

Table 3 shows the experimental results of recall, precision and F-score through the 4-fold cross-validation, from which we can see that the CRF-based learning method (henceforth CRF) increases the accuracy regarding almost all the four types of entities, except the slightly lower recall land F-score than L-HMMs method (henceforth L-HMM) in respect of component entity. Please note that the empty value of the baseline approach in terms of the four entity types is because the baseline method does not support to extract the four types of entities, so it only has the results averaged on all entities.

More specifically, CRF improved the precision from 83.9% to 90.0% on the average value and the F-score from 77.1% to 84.3% in comparison with L-HMM. Two major reasons can lead to this result. Firstly, L-HMM assumes that each feature is generated independently of hidden processes. That is, only tags can affect each other and the underlying relationships between tags and words/POS-tags are ignored. Secondly, HMMs does not model the overlapping features. As for recall, it was also averagely improved (from 72.0% by L-HMM to 79.8% by CRF). This is promising because the recall can be easily affected by errors in tagging. For example, if a sentence's correct tags should be "Opinion-B-P-Exp, Opinion-M-P-Exp, Opinion-M-P-Exp, Opinion- E-P-Exp" and it was labled

**Table 3.** Experimental results from the comparison of the three approaches: Baseline - the rule-based opinion mining method, L-HMM - the Lexicalized-Hidden Markov model based learning method, CRF - the Conditional Random Field based learning method (R: recall, P: precision, F: F-score)

| Methods | | Baseline | L-HMM | CRF |
|---|---|---|---|---|
| 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 |

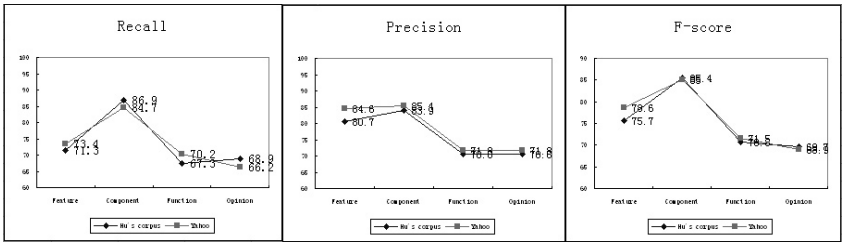


**Fig. 4.** Recall, Precision and F-score resulting from training CRF respectively with the two datasets

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.

We also conducted a comparison between the two datasets: 238 documents from [10] and 238 documents from Yahoo Shopping. The procedure is that we first trained the CRF on the first dataset and then tested it with the second one, and vice versa. Fig. 4 shows the precision, recall and F-scores averaged over four entities. It can be seen that there is no big difference between the two datasets. The largest distance occurs with the precision on feature entity, but it is only 3.9%. The result infers that our approach can achieve a stable performance with different datasets and can be hence easily scalable to mine review data from various sources (e.g., from other sites like Amazon), without the need of training examples for each site.

It is also worth noting that the overlapping feature functions as defined in our CRF model can likely increase the discovery of infrequent entities, which however were largely ignored in related approaches. For example, although the entity "ISO" only appears once in our data, functions $f(t_i, s_i, w_i)$ and $f(t_i, w_i)$ can be still active in finding this feature. Moreover, the uneasily discoverable entities, such as non-noun product entities (e.g., "flashing") and non-adjective opinions (e.g., "strongly recommended"), can be also identified by our approach.

## 5 Conclusion

Thus, in conclusion, we proposed a novel linear-chain CRF-based learning approach for Web opinion mining. Contrasting to L-HMM-baesd method which assumes that each feature is generated being independent of hidden states, CRF-based approach can more effectively handle with dependent input features . The experiment results indeed demonstrated the effectiveness of our proposed approach in comparison with the rule-based and L-HMM-based ones. In the future, we will target to incorporate more feature functions to improve our model. Due to the complicacy of natural languages, some long-distance features are beyond our assumption, but if some appropriate feature functions can be included, we could possibly employ automatic feature induction techniques to find non-obvious conjunction of features, which may hence further improve the algorithm's performance. Moreover, we will apply the opinion mining result from product reviews to enhance existing recommender systems, such as to address the rating sparsity limitation and infer users' preferences on products from their expressed opinions.

## Acknowledgement

Great thanks to my colleague Victor Cheng for his insightful suggestions and comments on our work.

## References

1. Wei, J., Hung, H., Rohini, S.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 (2009)
2. Turney, P.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 (2002)
3. Popescu, A., Etzioni, O.: Extracting product features and opinions from Reviews. In: Conference on Empirical Methods in Natural Language Processing, pp. 339–346 (2005)
4. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: sentiment classification using machine learning techniques. In: The ACL 2002 Conference on Empirical methods in Natural Language Processing, pp. 79–86 (2002)

5. Dave, K., Lawrence, S., Pennock, D.M.: Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In: 12th International Conference on World Wide Web, pp. 519–528 (2002)
6. Hatzivassiloglou, V., Wiebe, J.M.: Effects of adjective orientation and gradability on sentence subjectivity. In: 18th Conference on Computational linguistics, pp. 299–305 (2000)
7. Wilson, T., Hoffmann, P., Somasundaran, S., Kessler, J., Wiebe, J., Choi, Y., Cardie, C., Riloff, E., Patwardhan, S.: OpinionFinder: a system for subjectivity analysis. In: HLT/EMNLP on Interactive Demonstrations, pp. 34–35 (2005)
8. Scaffidi, C., Bierhoff, K., Chang, E., Felker, M., Ng, H., Jin, C.: Red Opal: product-feature scoring from reviews. In: 8th ACM Conference on Electronic Commerce, pp. 182–191 (2007)
9. Das, S., Mike, C.: Yahoo! for Amazon: Extracting market sentiment from stock message boards. In: Asia Pacific Finance Association Annual Conference (2001)
10. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 168–177 (2004)
11. John, L., Andrew, M., Fernando, P.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: International Conference on Machine Learning, pp. 282–289 (2001)
12. Fuchun, P., Andrew, M.: 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 (2004)
13. Fei, S., Fernando, P.: 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 (2003)
14. McCallum, A.: Efficiently inducing features of conditional random fields. In: Conference on Uncertainty in Artificial Intelligence (2003)
15. Miao, Q., Li, Q., Zeng, D.: Mining fine grained opinions by using probabilistic models and domain knowledge. In: 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (2010)
16. `http://www.cs.cornell.edu/People/pabo/movie-review-data/review_polarity.tar.gz`
17. `http://l2r.cs.uiuc.edu/~cogcomp/software.php`