World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:10, No:6, 2016

Product Features Extraction from Opinions According to Time

Kamal Amarouche, Houda Benbrahim, Ismail Kassou

***Abstract*—**Nowadays, e-commerce shopping websites have experienced noticeable growth. These websites have gained consumers' trust. After purchasing a product, many consumers share comments where opinions are usually embedded about the given product. Research on the automatic management of opinions that gives suggestions to potential consumers and portrays an image of the product to manufactures has been growing recently. After launching the product in the market, the reviews generated around it do not usually contain helpful information or generic opinions about this product (e.g. telephone: great phone...); in the sense that the product is still in the launching phase in the market. Within time, the product becomes old. Therefore, consumers perceive the advantages/ disadvantages about each specific product feature. Therefore, they will generate comments that contain their sentiments about these features. In this paper, we present an unsupervised method to extract different product features hidden in the opinions which influence its purchase, and that combines Time Weighting (TW) which depends on the time opinions were expressed with Term Frequency-Inverse Document Frequency (TF-IDF). We conduct several experiments using two different datasets about cell phones and hotels. The results show the effectiveness of our automatic feature extraction, as well as its domain independent characteristic.

***Keywords*—**Opinion mining, product feature extraction, sentiment analysis, SentiWordNet.

I. Introduction

TODAY, the web provides a good platform to retrieve information about products and their advantages/ disadvantages. After purchasing a product, many people share their satisfactions and critics about it. These comments often contain their opinions which become a new source of information for potential consumers and manufacturers. These opinions may, also, be analyzed for potential consumers to find out the advantages and disadvantages of a product of their interest. The large number of opinions is very difficult to analyse manually. This is why automatic methods were used to alleviate humans from this process. The main target is to solve this problem by mining these product opinions.

The product opinion mining can be performed at one of the three levels: Whole opinion level [1], sentence level [2], or feature level [3]. For the opinion level, the entire opinion generates a single value of polarity (positive, neutral or negative), which is not accurate, because an opinion can contain multiple sentences with various polarities and a general classification of the opinion is not suitable. But in the sentence level, polarities will be generated for each sentence. Feature level can be better in extracting an exact or useful summary about a product which can help the potential customers understand the product more clearly and compare it with competitors' products.

The extraction of product features is a fundamental step in product opinion mining at feature level. The polarity of each opinion within different features according to that has a wide variety of applications such as trust reputation system [4] product recommendation [5] and summary generation [6], [7]. In recent years, various product feature extraction approaches have been proposed by many researchers [3], [8]-[13]. Some research works [8]–[11] identified these features using relations dependencies approaches that are based on relations among terms appeared in reviews. Some of the problems of using these approaches are because of the complexity of languages that make the identification of all relations dependencies difficult to catch (that poses a problem of using these techniques). Therefore, after dependency relation analyses, some additional steps are used in order to extract these product features. Other approaches used or created an ontology to identify these features written in reviews [4], [14]–[17]. The creation of ontologies is a difficult step and its use needs a specific ontology for the domain we're working on to extract these features. For this reason, it is necessary that the right ontology be available if we want to make an effective product features extraction.

Other approaches based on machine learning mainly rely on terms occurrence frequencies that belong to opinions which become product features [3]. Some works based on this approach use the reviews from other domains to extract features for a specific one [3], [12], [13]. Some limitations of these works are the need to use the reviews from other domains to extract the features for a specific field based on the weight or the frequency of each feature for this specific domain compared with the other domains. Also, the choice of these comparative domains has an important role. e.g., if we use the dataset's 'tablet' as a comparative domain vis-à-vis the 'computer', the accuracy to extract the features of 'computer' will decrease; because there are many common features between both datasets. With regard to that, in most research works [18]–[20], only nouns and noun phrases are extracted as the feature word candidates.

In this paper, we present a method that extracts candidate product features from opinions. The basic idea of our method is to generate automatically a list of nouns and noun phrases that contains those candidate features. Then, the potential product features will be identified using an unsupervised

Kamal Amarouche, Houda Benbrahim and Ismail Kassou are in Al Bironi Research Team, ENSIAS, Mohammed V University, Rabat, Morocco (e-mail: amarouchekamal@gmail.com, houda.benbrahim@um5.ac.ma, ismail.kassou@um5.ac.ma).

method based on a combination of TF-IDF that identify the frequent candidate features and TW that depends on time when the opinion is expressed which gives more weight to new opinions expressed. The main idea behind our approach is that usually, right after the launching of the product in to the market, consumers who buy it do not really pin down its pros and cons with respect to its features; however, they automatically generate opinions that are more or less generic because the product is new to them. Despite their generic opinions, consumers form a strong opinion that contains their true feelings about the product and its features as time goes by. For that, a method that combines both TW and TFIDF has been proposed, giving the importance to the candidate features expressed in the recent opinions to extract the product features. Our method is more generic than TF-IDF because the value of TM is between 0 and 1. Therefore, the features extraction results are better or close to using TF-IDF only.

The remainder of the paper is organized as follows: Section II narrates the related work to extract product features. Section III presents our proposed system. An experimental study is presented in Section IV, as well as the analysis and the discussion of the results. Finally, conclusions and suggestions for further research are given in Section V.

II. Related Works

Previous studies have explored many different methods to extract the product features. Some works are based on dependency relation to extract these features. A phrase dependency tree has been constructed from results of chunking and dependency parsing, and based on this tree the candidate product features and opinion expressions have also been extracted [8]. It has been noticed that [11] used three types of dependency relations between each word in a sentence in order to identify product features or general opinion sentences in reviews. They assigned a product feature to each of the opinion sentences based on the frequently associated words that appear in the selected types of dependency relations. Besides, a semantic based approach has been proposed [10] that uses typed dependency relations to identify the product features based on the opinion word associated with them using different dependencies. Then, an algorithm is proposed to replace the pronoun present in the review sentence with the appropriate product feature. Another method [9] based on analysis of dependency relations that identifies the dependency relations between noun/noun phrases and sentiment terms has been proposed. Some filtering rules are applied to generate product features, then the implicit features are inferred based on the results of the NodeRank algorithm to finalize the product feature set.

Other works that use or create an ontology in order to extract the product features included in the opinions can be found; an ontology terminology to extract features about movie reviews has been used by Lili and Chunping [16]. Starting with preprocessed reviews, they identified related sentences which contain the ontology terminologies. From those sentences, they easily extracted the features. Also, Moreno et al. [15] used general domain ontology to extract features included in the opinions expressed about movies by users. Liu et al. [17] built automatically a domain-specific affective ontology by processing the online reviews to identify product features and corresponding opinions. First, they collected feature words and affective words as seed sets. Then, they extracted pairs of feature words and affective words from reviews by POS patterns including their sentiment polarity. Finally, they clustered the feature words and formed the affective ontology. Also, a product ontology about 'camera' and an online learning algorithm called Hierarchical Learning with Sentiment Ontology Tree (HL-SOT) has been manually constructed that has been developed to improve the effectiveness of product review classification [14]. It has been proposed [4] that a technique based on idiomatic ontology identifies product features and sub-features. First, the system used POS tagging to extract features and sub-features. Then, they compared these features and sub-features from the review with the already existing ones in the ontology database. If not exist, the tagging process will end and the feature text will be added to a seed list that a supervisor verifies in order to either add it as a feature in the ontology data base or not.

Other studies are based on machine learning to extract these features; Hu and Liu [19] proposed an approach based on association mining in order to extract product features. The main idea of this approach is that people often use the same words when they comment on the same product features. Then frequent itemsets of nouns in reviews are likely to be product features while the infrequent ones are less likely to be product features. However, the drawback of using this approach is to generate many features which do not actually represent true product features like e.g. 'comment'. A probabilistic approach to extract domain specific features has been used by [12]. The basic rationale of this approach application is that, features which are more specific to a certain field of the product have a higher probability of appearing in opinions belonging to this product field than the ones appearing in opinions belonging to other fields. Based on this observation, they defined the characteristic power of a feature for a product field, as the probability that an opinion containing that feature belongs to the field compared to the probability that the opinion belongs to other fields. Kushal and Durga [3] combined both previous approaches (association mining [19] and probabilistic [12]) to identify these features. Initially, they applied association rule mining to extract the most frequent occurring features in the user reviews for a product. Then, a probabilistic approach has been used in order to extract the potential features based on these weights in a specific field compared to generic domain. Moreover, it has been proposed [21] that OPINE (an unsupervised information extraction system) works with acquiring frequent candidate nouns by setting a threshold frequency. Then, frequent candidates are assessed by Point-wise Mutual Information (PMI) between a candidate and a product class. Changqin and Fuji [13] proposed an unsupervised method to extract the product feature using comparative domain corpora. The basic idea of this approach is to extract domain product features through the evaluation of their weights in different related domains. They applied a term

similarity measure to evaluate the association of candidate features and domain words. Based on these similarities, a domain vector for each candidate feature can be derived. Then the domain-specific features are extracted by measuring the distances between the domain vectors of features and the domain vector of a domain entity. A Term Frequency-Inverse Sentence Frequency (TF-ISF) on nouns and noun phrases that are identified on the sentences has been applied [22]. Then, top five features are treated as opinion features. Also, Saruladha [7] computed TF-ISF for each term in the candidate set based on the term-sentence matrix to measure its relevant importance in the sentence. The difference between the previous and this work is that, in this work top candidates with highest TF-ISF scores are opinion are considered as features set.

Our work is different from the aforementioned works because we propose a method which combines both methods that extract the frequent candidate product features using TF-IDF and TW giving a great weight to the features expressed in recent opinions compared to old opinions, because latest opinions expressed often contain product features. Based on this proposal, our method gives an importance to candidate features expressed from the latest to the oldest opinions.

III. Our System for Product Feature Extraction

As discussed in the introduction section, our system shown in Fig. 1 combines the TF-IDF and time when an opinion is expressed to extract the product/service features. The inputs of the system are products of the same type (for example phone or camera ...) that is crawled from different sources such as online sale websites or social networks. This system operates in five main steps that are explained in details in the following sub-section: (1) Pre-processing; (2) Brand Names Filtering; (3) Grouping Synonym; (4) Candidate Feature Extraction; (5) Product Feature Extraction.

A. Pre-Processing

The text documents that contain an opinion must be pre-processed and stored in more appropriate data structures for further processing. Frequently, these opinions contain several syntactic features that may not be useful for the next steps. After cleaning the data (delete @, !, ?, digits....), the message in the row data set must be extracted into text files. After that, we will detect [23] the language of each opinion in order to select only the opinions written in English. Then, we will detect and correct the spelling errors written in these opinions using a web service [24]. After this step, we will apply the Part-Of-Speech Tagger [25] (POS tagger) in order to associate each word with its grammatical function, because different functions act as indicators for different aspects of the content in which we need to know the word nouns. The main goal of using this tagger in our system is to identify and convert plural to singular nouns that is important to increase its weight in order to facilitate the product feature identification step (e.g. batteries to battery in phone reviews). Also, to select the words noun that is important to build the candidate feature list using our method.

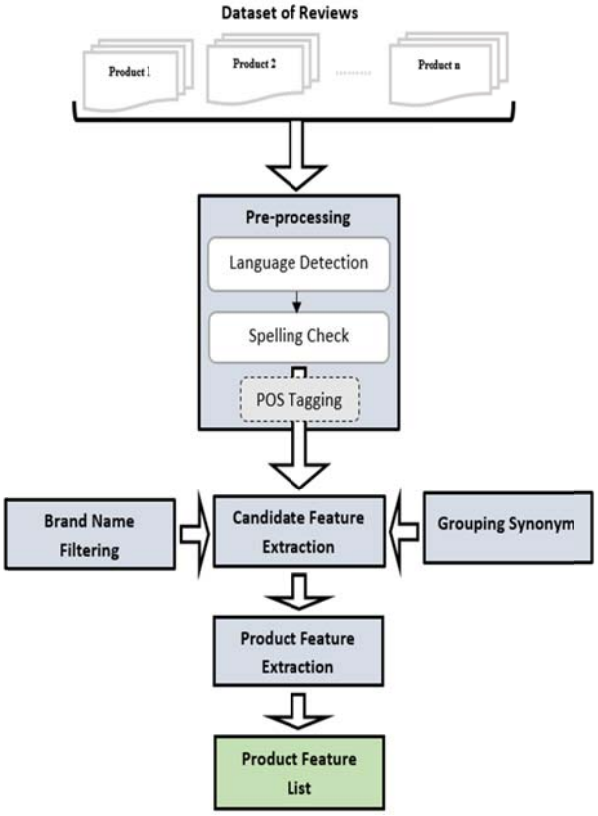

Fig. 1 The Overview of the Proposed Product Feature Extraction approach

B. Grouping Synonyms

The problem encountered is that people use many different word nouns to describe the same meaning of a word, e.g. 'image', 'photo' and 'picture' have the same meaning for phone products. To facilitate product feature extraction step, these words, which are synonyms, need to be grouped under the same word. After applying POS tagging that identify noun words, we will group these synonym nouns using SentiWordNet [26].

C. Brand Name Filtering

Some nouns or noun phrases (brand names that will be extracted using candidate feature extraction step) are not product features and they are available with high weight in the opinions. So, the main goal of this step is to filter the input of candidate features extraction step from these nouns using a list created manually that contains brand names.

D. Candidate Feature Extraction

The problem solved in this step is to extract automatically candidate features from opinions which are likely to become product features. We know that, usually, features of a product are nouns. So, the main goal of this step is to build a list that contains nouns composed of one or multiple words using our method described in Fig. 2 that will generate the most frequent nouns from our dataset about a specific product field. This method is derived from the work of Rakesh and Ramakrishnan [27].

After recovering our dataset of opinions, our method identifies, first, the nouns composed of one word to determine the unigram-itemsets. Then, it generates the candidate features that are composed of multiple words (respecting a minimum support minsup that represents the frequency of these nouns in the dataset of opinions). The detailed steps are described below:

- The first step creates the largest possible candidate features list that is composed of multiple words using *Candidate_Feature_Generation* function which will be described in subsection 1 (Candidate Feature Generation);
- The second step relates to count the frequency of these features for each opinion o_i, and selects only the candidates the candidates who have support (frequency) greater than the minimum support (minsup);
- The third step, *Candidate_Feature_Filtering* function allows to generate the true candidate features composed of k-words (L_{k-1}) that will be described in subsection 2 (Candidate Feature Filtering);
- Finally, we will generate all candidate features that contains the most frequent nouns composed of one or multiple words.

1) $O = Dataset$; // dataset of reviews
2) $CF_1 = \{Unigram - itemsets\}$;
// candidate feature composed of one noun
3) $for(k = 2;\ CF_{k-1} \neq \phi; k++)\ do\ begin$
Step 1
4) $C_k = \boldsymbol{Candidate\_Feature\_Generation}(\boldsymbol{CF_{k-1}})$;
// New candidates (Fig. 3)
Step 2
5) $for\ each\ o_i \in O$
6) $C_{o_i} = subset(C_k, o_i)$;
// Candidates contained in o_i
7) $for\ each\ candidates\ c \in C_{o_i}\ do$
8) $c.count++$;
9) end
10) end
11) $CF_k = \{c \in C_k | c.couunt \geq minsup\}$
Step 3
12) $L_{k-1} = \boldsymbol{Candidate\_Feature\_Filtering}(CF_{k-1}, CF_k)$;
// (Fig. 4)
13) end
14) $CF\ List = \{L_1 \cup L_2 \cup \ldots \cup L_{k-1} \cup CF_k\}$

Fig. 2 Frequent Candidate Product Feature Extraction

1) Candidate Feature Generation

This function (Fig. 3) takes as argument a possible candidate features composed of (k-1)-words and returns a possible candidate features composed of k-words. For example, we have a candidate feature that contains the noun composed of one word (Fig. 5) be it {{battery}, {life}, {image}, {screen}}. After implementing this function for k=2, we obtain a list composed of {battery life}, {life battery}, {life image}, {image life}, {battery image} and {image battery}. Additionally, we will delete the candidates don't exist in opinions to build a new possible candidate features that composed of 2-words in order to obtain a list composed of just {battery life} (the other combinations of word nouns do not exist in opinions).

1) $insert\ into\ C_k$
2) $select\ p.item_1, p.item_2, \ldots, p.item_{k-1}, q.item_{k-1}$
3) $from\ CF_{k-1}\ p, CF_{k-1}\ q$
4) $where\ p.item_1 = q.item_1, \ldots, p.item_{k-2} = q.item_{k-2}, p.item_{k-1} \neq q.item_{k-1}$;
5) $forAll\ itemsets\ c \in C_k$
6) $if\ c \notin O\ then$
7) $delete\ c\ from\ C_k$; // delete this candidate if does not exist in all opinions
8) end
9) $return\ C_k$

Fig. 3 Candidate Feature generation

2) Candidate Feature Filtering

The main goal of this function is to produce the true candidate features composed of (k-1)-words (L_{k-1}) that do not exist in the possible candidate features composed of k-words (CF_k) and its support is greater than minsup (in the example shown in Fig. 5, the minsup =2). For example, in Fig. 5 we have the possible candidate features composed of two words (CF_2) be it {battery life} and composed of one word (CF_1) be it {{battery}, {life}, {screen}, {image}}. After applying this function, we will delete {battery} and {life} from CF_1 (these possible features belong to {battery life}) in order to generate a list that composed of {screen} and {image}.

1) $forAll\ cf_{k-1} \in CF_{k-1}\ do\ begin$
2) $forAll\ cf_k \in CF_k\ do\ begin$
3) $if(\ cf_{k-1}\ subsets\ of\ cf_k)\ then$
4) $TEMP_k = cf_k$;
5) end
6) $forAll\ o \in O\ do\ begin$
7) $if(\ cf_{k-1}\ subsets\ of\ o\ and\ cf_{k-1} \notin TEMP_k)\ then$
8) $cf_{k-1}.count++$;
9) end
10) $L_{k-1} = \{\ cf_{k-1} \in CF_{k-1} |\ cf_{k-1}.count \geq minsup\}$;
11) end
12) $return\ L_{k-1}$

Fig. 4 Candidate Feature Filtering

E. Product Feature Extraction

The main goal of this part is to extract the potential features from the CF List based on our method that combines TW and TF-IDF. The objective of our method is to give the importance to the last opinion expressed to calculate the weight of each candidate feature (cf_i), for this reason we introduce TW (4) that depends on the time when each opinion is expressed. This function (TW) is derived from logistic function [28] that has a curve like a product life cycle. As we know product features that are often shared about a product in opinions, appear from the launching phase of product life cycle that is why we introduce TW in the combination (1) to extract product features.

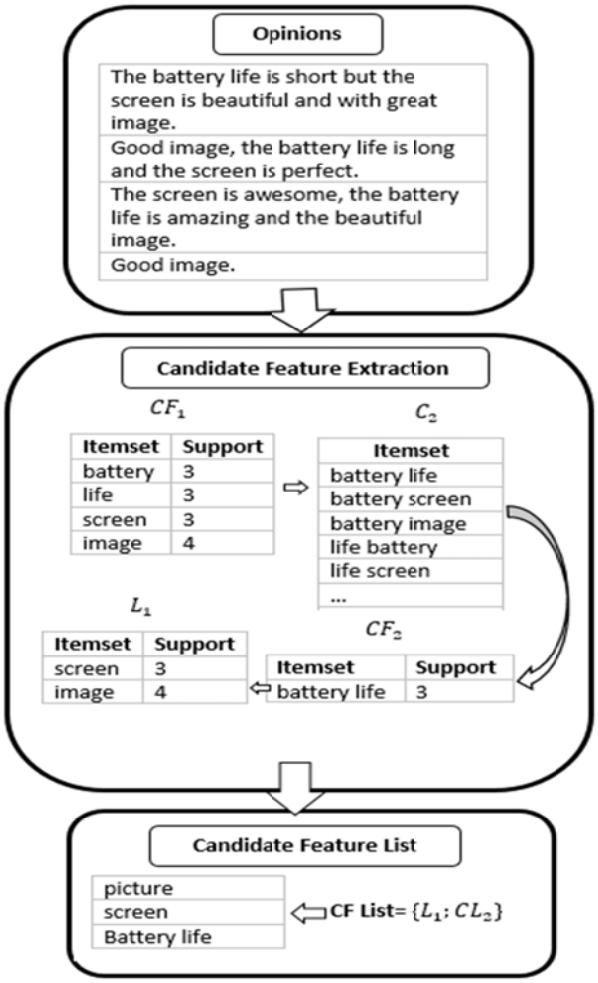

Fig. 5 Candidate Feature Extraction Example

Before calculating TW for each opinion, the first step is to quantify each date (time when each opinion expressed) in order to give it as input to TW. For this reason, we sort dates of opinions from oldest to latest. Then, we choose the step (hour, day, week, month…) that depends on the product that we are extracting its features.

$$\begin{array}{c} \\ CF\ List \end{array} \begin{array}{c} \begin{matrix} o_1, t_1 & \cdots & o_j, t_j & \cdots & o_m, t_m \end{matrix} \\ \begin{bmatrix} cf_1 & w_{11} & \cdots & w_{1j} & \cdots & w_{1m} \\ \vdots & \vdots & \ddots & \vdots & & \vdots \\ cf_i & w_{i1} & \cdots & w_{ij} & \cdots & w_{im} \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ cf_n & w_{n1} & \cdots & w_{nj} & \cdots & w_{nm} \end{bmatrix} \end{array}$$

Fig. 6 Calculation of weight between candidate feature and opinion-time pair

Finally, we will generate for each opinion o_j its value $t_j$that depends, at the same time, on the time when opinion is expressed and its seniority (depends on the step chosen). The value of TW is between 0 and 1 (if the opinion is the oldest, the value of TW close to 0, contrariwise if it is expressed recently the value of TW close to 1). e.g. we have a dataset that contains the opinions from 2016-01-01 into 2016-01-28 and we choose the week as a step. So, we generate $t_1 = 1$ from the opinions that were expressed from 2016-01-01 into 2016-01-07, $t_2 = 2$ from the opinions that were expressed from 2016-01-08 into 2016-01-14… Finally, we generate $t_4 = 4$ from the opinions that were expressed from 2016-01-22 into 2016-01-28. Then, we can calculate the weighting between each cf_i and $o_j - t_j$ pair through TW-TFIDF (figure 6) and each cf_i pair is represented as a vector in m-dimensional vector space.

$$\begin{aligned} &TW - TFIDF\ (cf_i, o_j, t_j) = w_{ij}(cf_i, o_j, t_j) \\ &= TW\ (t_j) \times TF\ (cf_i, o_j) \times IDF\ (cf_i) \end{aligned} \tag{1}$$

$$IDF\ (cf_i) = Log\left(\frac{|o|}{DF\ (cf_i)}\right) \tag{2}$$

$$TF\ (cf_i, o_j) = \frac{count\ (cf_i \in o_j)}{count\ (w \in o_j)} \tag{3}$$

$$TW\ (t_j) = \left(\frac{1}{1 + e^{-kt_j}}\right) \tag{4}$$

cf_i – a candidate feature which is present in the CF List; $DF(cf_i)$ – Number of opinions in which the candidate feature cf_i occurs; $|o|$ – Number of all opinions in our dataset; w – words present in the opinion o_j of a product; t_j –time when opinion o_j expressed; k – an adjustment value of TW.

TFIDF can calculate the frequency of these features for each opinion and TW assigns each feature a weigh that depends on the time of expressing the opinion. The changing value of k (adjustment value of TW) can vary the dispersion of weights for each feature around the time when these opinions are expressed. All depends on the type of the product and its familiarity by customers, because there are some products that take much time in this phase while other products don't. Therefore, the value of k depends on the product domain and customer knowledge. The combination (1) of TFIDF and TW extracts the most frequent features giving the priority to features expressed in the latest opinions.

For each candidate feature cf_i, the average weight is computed using (5). Then, we select the most important candidate feature using (6) in order to generate a list that contains the product features. For example: After extracting the candidate feature 'battery life' using Candidate Feature extraction method from cell phone reviews, its weight for each $o_j - t_j$ pair can be calculated using (1). Next step is to compute the weight average value of this candidate using (5). Once the value of the average is computed, the final step is to verify whether its average is higher than a threshold (α) using

(6).

$$avg\,(TW-TFIDF\,(cf_i, o_j, t_j)) = \frac{1}{|o|}\sum_{j=1}^{|o|} w_{ij} \quad (5)$$

$$cf_i \in \{\, product-\ feautre \,\}\ \ if \\ avg\big(TW-TFIDF(cf_i, o_j, t_j)\big) \geq \alpha \quad (6)$$

IV. Experiments and Results

Our system has been implemented in Java and it has generated product features that are generally common among all products of the same type. The procedure to extract these features is as follows: After collecting opinions, the first step is to select only the opinions written in English using langdetect [23]. After that, we clean these opinions from different syntactic features that may not be useful in next steps. Then, we detect and correct the spelling errors [24]. After, POS tagger tags all the words in each opinion to their appropriate part of speech tag in order to identify word nouns that are important to build a candidate features list. At last, we group nouns synonyms and filter these opinions from brand names. The next step is to extract candidate features using the method indicated in Fig. 2 to obtain a list of these candidates in order to calculate the weight for each candidate and select the most important candidates using our approach TW-TFIDF that combines TW and TFIDF.

A. Data collection

We have conducted experiments using two data sets. The first is about cell phones. We use six products. The customers' opinions for these products were collected from gsmarena.com. The details of this dataset are shown in Table I. The second dataset is about hotels. We used reviews from four hotels that were collected from TripAdvisor.com, and the details of this dataset are shown in Table II.

B. Metrics

The performances of our proposed approach are measured using the standard evaluation measures of $Precision\ (p)$, $Recall\ (r)$ and $F-score\ (f)$. Specifically, precision is the fraction of extracted features that are correct; recall is the fraction of correct features that are actually extracted among all correct features; and F-measure $f = (2pr)/(p+r)$. Higher values of p, r, and f indicate better performance. For example, our system has recognized 50 candidates as product features. Only 30 have been predicted correctly ant total number of manual product features is 60. So, the precision and recall will be calculated as: $sion = \frac{30}{50}$ and $recall = \frac{30}{60}$.

C. Tuning Parameters

The parameters of tuning used in our approach are: minimum support (minsup), adjustment value k (4) and threshold value α (6). The minsup is used in the candidate feature extraction step to identify the candidate features for each or all products. So, we changed the value of this parameter in order to generate a list that contains all candidate features. However, the value of k depends on the interval of crawling opinions (since product launch in market or not). Therefore, we changed its value to obtain good results. Finally, the threshold value α selects product features from the list that contains their candidates based on the average weight (selects the candidates that have a great average).

TABLE I
Data Description of Cell Phone Reviews

Product (P)	Product Description	Number of reviews	Number of manually extracted features	Average of words	Interval of extraction
p_1	blackberry leap	480	25	49	From 03 Mar 2015 to 23 Dec 2015
p_2	samsung galaxy j7	882	33	34	From 09 Apr 2015 to 22 Dec 2015
p_3	htc one m9	2058	34	50	From 22 Jan 2015 to 15 Dec 2015
p_4	sony xperia e4g	897	36	56	From 24 Feb 2015 to 11 Dec 2015
p_5	samsung galaxy s6 edge	2502	37	56	From 01 Mar 2015 to 09 Dec 2015
p_6	sony xperia m4 aqua	881	32	49	From 02 Mar 2015 to 19 Dec 2015

TABLE II
Data Description of Hotel Reviews

Hotel (H)	Location	Number of reviews	Number of manually extracted features	Average of words	Interval of extraction
h_1	San Francisco, California	324	32	150	From 21 nov 2006 to 7 jan 2009
h_2	Barcelona, Catalonia	244	27	218	From 4 nov 2004 to 5 jan 2009
h_3	Hong Kong	249	28	190	From 5 jun 2005 to 7 jan 2009
h_4	New York	263	46	200	From 16 feb 2006 to 6 jan 2009

D. Data Analysis and Results

Table III gives the experimental results of extracting product features. The performances are measured using the standard evaluation measures of precision (p), recall (r) and F-score (f) that we previously mentioned. The experiments have indicated that our approach using $t_j = day$ gives the values that are better than using TF-IDF to extract these features. Also, Table IV shows the experiment results to extract features from hotel reviews which indicates that our approach is independent of the domain. The parameter value k is important to obtain good results. e.g. we changed its value about products p_2, p_3 and p_6 to obtain f-measure = 0.60 (t_j = day). In addition, the large number of opinions is important to better extract these features. Good results are obtained from cell phone reviews where t_j=day because these opinions are shared on a daily basis and our dataset contains the opinions from the product once launched.

Table IV shows good results of hotel features extraction that indicate that our method gives good results compared to TFIDF, but the few numbers of opinions (1080 opinions) influenced on this extraction.

TABLE III
COMPARISON OF PRECISION, RECALL AND F-SCORE VALUE RESULT OF OUR APPROACH TW-TFIDF WITH TF-IDF TO EXTRACT PRODUCT FEATURES FROM CELL PHONE REVIEWS

	TF-IDF			TW-TFIDF								
				$t_j = day$			$t_j = week$			$t_j = month$		
P	p	r	f	p	r	f	p	r	f	p	r	f
p_1	0.56	0.52	0.54	0.54	0.52	0.53	0.54	0.52	0.53	0.54	0.52	0.53
p_2	0.46	0.57	0.51	**0.54**	**0.66**	**0.60**	0.50	0.61	0.55	0.56	0.51	0.53
p_3	0.39	0.50	0.44	**0.57**	**0.64**	**0.60**	0.59	0.55	0.57	0.47	0.53	0.50
p_4	0.47	0.53	0.50	**0.51**	**0.58**	**0.54**	0.54	0.54	0.54	0.46	0.53	0.49
p_5	0.40	0.43	0.43	**0.55**	**0.55**	**0.55**	0.50	0.57	0.53	0.53	0.54	0.53
p_6	0.45	0.59	0.51	**0.55**	**0.66**	**0.60**	0.56	0.59	0.57	0.50	0.62	0.55
Avg.	0.45	0.52	0.48	**0.54**	**0.60**	**0.57**	0.54	0.52	0.53	0.51	0.54	0.52
All product	0.42	0.50	0.46	**0.54**	**0.61**	**0.57**	0.50	0.50	0.50	0.49	0.54	0.51

TABLE IV
COMPARISON OF PRECISION, RECALL AND F-SCORE VALUE RESULT OF OUR APPROACH TW-TFIDF THAT USING TIME WITH TF-IDF TO EXTRACT PRODUCT FEATURES FROM HOTEL REVIEWS

	TF-IDF			TW-TFIDF								
				$t_j = month$			$t_j = 6\,month$			$t_j = year$		
H	p	r	f	p	r	f	p	r	f	p	r	f
h_1	0.41	0.55	0.47	0.43	0.57	0.49	0.44	0.57	0.50	0.42	0.57	0.51
h_2	0.41	0.41	0.41	0.47	0.52	0.49	0.44	0.55	0.49	0.37	0.63	0.47
h_3	0.36	0.46	0.40	0.46	0.50	0.48	0.42	0.46	0.44	0.40	0.46	0.43
h_4	0.36	0.37	0.36	0.54	0.54	0.54	0.54	0.53	0.53	0.54	0.53	0.53
Avg.	0.38	0.45	0.41	0.47	0.53	0.50	0.46	0.52	0.49	0.43	0.54	0.48
All hotel	0.38	0.45	0.41	0.42	0.48	0.45	0.43	0.48	0.45	0.45	0.48	0.46

V. CONCLUSION AND FUTURE WORK

In this paper, a product features list is automatically constructed based on the opinions about comparative products (same type). We proposed an unsupervised method that depends on the time when the opinions are expressed to select the potential features from this list without the need of opinions from other domains. The results obtained are justified: first, to introduce the time axis which is important to extract these features. Second, the approach proposed does not refer to any information of the domain. So, it is a domain independent.

In the future work, we aim at improving our method in order to extract implicit product features and generate this method for other languages.

REFERENCES

[1] Rodrigo Moraes, João Francisco Valiati, Wilson P. Gavião Neto, "Document-level sentiment classification: An empirical comparison between SVM and ANN," *Expert Systems with Applications* 40, 2013, pp 621–633.

[2] Sylvester Olubolu Orimaye et al., "Buy It – Don't Buy It: Sentiment Classification on Amazon Reviews Using Sentence Polarity Shift," *In*: *Proceedings of International Conference on Artificial Intelligence*, Kuching, Malaysia,2012, pp 386-399.

[3] Kushal Bafna and Durga Toshniwal, "Feature based Summarization of Customers' Reviews of Online Products," *Procedia Computer Science 22*,2013, pp 142-151.

[4] Hasnae Rahimi and Hanan EL Bakkali, "CIOSOS: Combined Idiomatic-Ontology Based Sentiment Orientation System for Trust Reputation in E-commerce," *International Joint Conference*,2015, pp 189-200.

[5] Ruihai Dong, Michael P. O'Mahony, Markus Schaal, Kevin McCarthy, Barry Smyth, "Combining similarity and sentiment in opinion mining for product recommendation," *Journal of Intelligent Information Systems*,2014, pp 1-28.

[6] Minqing Hu and Bing Liu, "Mining and summarizing customer reviews," *In Proceedings of 10th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'04)* USA, 2004.

[7] Saruladha. K, Banupriya. D, Nargis Banu. J, "Opinion Summary Generation for Product Reviews," *International Journal of Engineering Research & Technology (IJERT)*. Vol. 3 - Issue 3, 2014.

[8] Y. Wu, Q. Zhang, X. Huang, L. Wu, "Phrase dependency parsing for opinion mining," *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, (Singapore),2009, pp. 1533–1541.

[9] Z. Yan, ET AL., "EXPRS: An extended pagerank method for product feature extraction from online consumer reviews," Inf. Manage. Available from: http://dx.doi.org/10.1016/j.im.2015.02.002.

[10] Ravi Kumar V. and K. Raghuveer, "Dependency Driven Semantic Approach to Product Features Extraction and Summarization Using Customer Reviews," *In Proceedings of the Second International Conference on Advances in Computing and Information Technology (ACITY)*, Chennai, India - Volume 3, 2012, pp 225-238.

[11] Qadir, A.,"Detecting Opinion Sentences Specific to Product Features in Customer Reviews using Typed Dependency Relations," *In: Events in Emerging Text Types (eETTs)*, Borovets, Bulgaria, 2009, pp. 38–43.

[12] Silviu Homoceanu, Michael Loster, Christoph Lofi and Wolf-Tilo Balke, "Will I like it? – Providing Product Overviews based on Opinion Excerpts," *IEEE Conference on Commerce and Enterprise Computing (CEC),* Luxembourg, 2011.

[13] Changqin Quan and Fuji Ren, "Unsupervised product feature extraction for feature-oriented opinion determination," *Information Sciences 272*, 2014, pp 16-28.

[14] Wei Wei and Jon Atle Gulla, "Sentiment learning on product reviews via sentiment ontology tree," *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics,* Uppsala, Sweden, 2010, pp. 404–413. The Association for Computer Linguistics.

[15] Moreno, V., Fraga, A., Sanchez-Cervantes, J.L., "Feature-Based Opinion Mining through ontologies," *Expert Systems with Applications*, 2014.

[16] Lili Zhao and Chunping Li, "Ontology Based Opinion Mining for Movie Reviews," *Volume 5914 of the series Lecture Notes in Computer Science*, 2009, pp 204-214.

[17] Liu Li-zhen et al., "Generating Domain-Specific Affective Ontology from Chinese Reviews for Sentiment Analysis," *Journal of Shanghai Jiaotong University (Science)*, 2015, pp 32-37.

[18] Zhichao Li, Min Zhang, Shaoping Ma, Bo Zhou, Yu Sun, "Automatic Extraction for Product Feature Words from Comments on the Web," *5th Asia Information Retrieval Symposium*, Sapporo, Japan, 2009, pp 112-123.

[19] Hu, M., Liu, B., "Mining opinion features in customer reviews," *In: Proceedings of AAAI,* 2004, pp. 755–760.

[20] H. Nakagawa and T. Mori, "A simple but powerful automatic term extraction method," *Int. Workshop on Computational Terminology*, Morristown, NJ, USA, 2002.

[21] Ana-Maria Popescu, Oren Etzioni, "Extracting product features and opinions from reviews," *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing,* 2005, pp.339-346, Canada.

[22] D. Wang, Shenghuo Zhu and Tao Li, "SumView: A Web-based engine for summarizing product reviews and customer opinions," *Journal: Expert System with Application 40*, 2013, pp. 23-37.

[23] Language Detection with Infinity-gram. https://github.com/shuyo/language-detection.

[24] Spelling suggestions for text. http://wsf.cdyne.com/SpellChecker/check.asmx.

[25] Stanford Natural Language Processing Group. http://nlp.stanford.edu/software/tagger.shtml.

[26] S. Baccianella, A. Esuli, and F. Sebastiani, "SENTIWORD NET 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining," *7th Conf on International Language Resources and Evaluation (LREC),* Marrakech, Morocco: European Language Resources Association (ELRA), 2008.

[27] Rakesh Agrawal Ramakrishnan Srikant, "Fast Algorithms for Mining Association Rules," *Proceedings of the 20th VLDB Conference Santiago*, Chile, 1994.

[28] Weisstein, E.W. Logistic Equation. (MathWorld A Wolfram Web Resource, 2003). http://mathworld.wolfram.com/LogisticEquation.html