Appl. Math. J. Chinese Univ.
2019, 34(3): 356-378

Generating Quantitative Product Profile Using Char-Word CNNs

XU Hai-rui[1] ZHANG Wei-cheng[1] LI Ming[1,*] QIN Fei-wei[2]

Abstract. The online customer reviews provide important information for product improvement and redesign. However, many reviews are redundant with only several short sentences, which may even conflict with each other on the same aspect of a product. Thus it is usually a very challenging task to extract useful design information from the reviews and provide a clear description on the product's various aspects amongst its competitors. In order to resolve this issue, we propose an approach to build hierarchical product profiles to describe a product's kernel design aspects quantitatively. It is achieved via three main strategies: a double propagation strategy to achieve the associated features and customers' descriptions; a deep text processing network to build the aspect hierarchy; an aspect ranking approach to quantify each kernel design aspect. Experimental results validate the effectiveness of the proposed approach on online reviews.

§1 Introduction

Online shopping has become an indispensable part of people's lives, and online retailing is occupying a growing share of the retail industry. In 2017, online sales reached 5.7 trillion RMB in China, which occupies 15.5% of the whole consumer sales. Thereafter, it is expected to increase at a rate of over 1% per year. Thousands of products have been uploaded to online shopping websites. For instance, Taobao and Amazon have thousands of goods in different categories, and a mass of goods are published per minute. At the same time, the online shopping platform receives billions of reviews, including general experiences, customer experiences and shopping experiences. For manufactures, these customer reviews are very valuable for updating products and developing new products.

Received: 2018-12-01. Revised:2019-03-22.
MR Subject Classification: 35B35, 65L15, 60G40.
Keywords: customer reviews, quantitative product profile, product aspect ranking, deep learning.
Digital Object Identifier(DOI): https://doi.org/10.1007/s11766-019-3698-x.
Supported by National Natural Science Foundation of China (61872320, 61772164, 61502129), and the Key Research & Development Program of Zhejiang Province (2019C03127).
*Corresponding author.

However, extracting useful customer opinions from online reviews is quite difficult. (1) There are high volumes of reviews. It is a labor-consuming and time-costing work for engineer to check the availability of reviews one by one. (2) Every customer has his own preference, which leads to different standards. Different customers may have different opinions on one feature of the same products. For instance, a same phone may have two conflict opinions such as '*Lagging phone. My phone is choppy when I rotate it*' and '*Short response time and fluid customer experience*', etc. (3) A product may have hundreds of features. For example, mobile phones have more than two hundred features such as '*battery*', '*screen*', '*camera*', etc. Therefore identifying the most important features from hundreds of features is very important for manufacturers to understand which features are important to customers.

To solve these problems, we propose an approach for generating hierarchical quantitative product profile. Similar to the user profile, the product profile provides a high-level overview of all aspects of the product and gives a clear and correct understanding to customers to get a better decision making. A product profile consists of three parts: product features, product kernel design aspects and quantitative analysis of each product kernel design aspects. The kernel design aspect contains many product features extracted from online reviews. As shown in Fig.1, we define seven product kernel design aspects and each design aspect has many product features, such as '*tightness*' and '*flexibility*' under the '*quality*'. Furthermore, each kernel design product aspect has quantitative weights to infer the importance of the kernel design aspects. Therefore, it is very important to generate a hierarchical quantitative product profile which allows manufacturers to focus on the most important product kernel design aspects.

The proposed approach consists of three main steps: sentiment analysis, product feature classification and product kernel design aspect ranking. Our key insight is that product features and opinions extracted from customer reviews can be organized as a part of product profiles, which describe the products in seven different kernel aspects. After product feature classification using deep neural networks, we can get the overall opinions in each kernel aspect by aggregating the product opinions under the kernel aspects. In order to illustrate the product preciously, we rank the kernel aspects to infer the importance of kernel design aspects by taking into account the product sales and the overall opinions in kernel aspects.

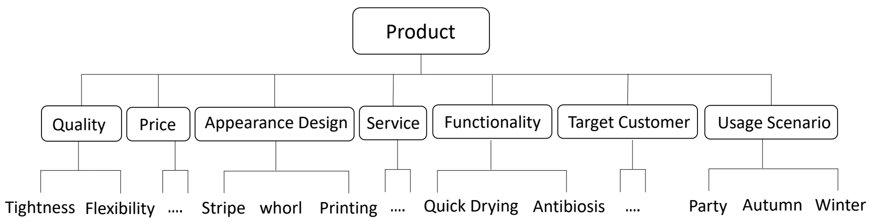

Figure 1: The illustration about the hierarchical product profile.

The main contributions of our work are summarized as follows:

- A novel approach for generating a hierarchical quantitative product profile which provides a clear description on product's various aspects.
- An unsupervised opinion mining algorithm that extract customer opinions as opinion units from online reviews by seed words.
- Distant supervision methods to extract text as labeled data from online reviews and classify the product features by the convolutional neural networks.
- Probabilistic product ranking model to infer the importance of product kernel design aspects by the associations between the amount of product sales and the opinions on specific aspects.

§2 Related Work

Our work has been inspired by recent progress in several different domains such as sentiment analysis, text classification and aspect ranking in generating product profile which we review below.

2.1 Sentiment Analysis

Extracting customer opinions and entities (product features) is one of the fundamental problems in sentiment analysis. Product features are always considered to be nouns or noun phrases and customer opinions are adjectives. The existing methods are mainly divided into two categories: supervised learning and unsupervised learning. For unsupervised learning, Hu and Liu et al. [2] select high frequency nouns as candidate product features. Wu et al. [16] and Qiu et al. [11] apply sentence dependency parser to opinion mining. They propose double propagation method to extract more opinions and features words iteratively by the seed words and the predefined rules. For supervised learning, it is mainly divided into traditional machine learning and neural networks. Li et al. [5] apply the conditional random field (CRF) as a sequence labeling problem for opinions extraction. In order to reduce the manual feature engineering of traditional machine learning, deep neural networks such as RNN and LSTM have been proposed and achieved good results [10][21]. Wang et al. [15] propose a multi-level attention mechanism extraction model based on LSTM network. Although supervised learning has a better effect than unsupervised learning, supervised learning needs a large number of expensive labeled data. And the performance of the supervised model greatly depends on the quality of the labeled data. Therefore, in order to avoid these problems, we use rule-based unsupervised learning in opinion mining.

2.2 Text Classification

The task of text classification is to classify the short text into pre-specified categories. Many traditional machine learning methods have good performance in text classification, such as naive Bayes, decision trees, SVM, etc. With the development of neural networks and the proposal of language models such as CBOW and Skip-gram [8], the high-dimensional sparse word bag is transformed into a low-dimensional dense continuous vector, which enables the application of neural networks in text classification. The proposal of RNN allows the temporal features of the text to be preserved and greatly improves the accuracy of text classification [18]. Mnih et al. [11] propose the attention mechanism, and then Yang et al. [17] propose a multi-layered attention mechanism in text classification tasks, which greatly increases the accuracy of neural networks. In addition to recurrent neural networks, convolutional neural networks are also used in text classification. Kim et al. [3] first employ convolutional neural networks to text classification tasks. Zhao et al. [22] and others add the attention mechanism to the convolutional neural networks and achieve good results. In this paper, we construct the word-related and char-related features as the input of the convolution neural networks, which greatly improves the accuracy of the model.

2.3 Aspect Ranking

Products may have hundreds of features, so it is important to rank product aspects for identifying important aspects. Wang et al. [14] develop a latent aspect rating regression model for discovering each customer's latent opinions and the relative emphasis on different aspects. Snyder et al. [13] present an algorithm that model the dependencies between assigned ranks to solve a multiple aspect ranking problem. This work mainly concentrates on each customer opinion estimation and customer rating behavior analysis, rather than on aspect ranking. Different from Wang, Zha et al. [20] propose a probabilistic aspect ranking model to infer the importance of various product aspects, which simultaneously utilizes aspect frequency and the influence of customers opinions on each aspect over the overall product opinions. Liu et al. [6] propose a method for product ranking based on sentiment analysis techniques and IFSs through online reviews. Different from their model based on customer rating of online reviews, we propose a new probabilistic model by the volume of product's sales and the customer sentiment on the product, which can directly obtain the overall importance of product aspects without aggregating each customer opinion.

§3 Overview of Approach

Our purpose is to generate a quantitative product profile based on customer reviews and online sales. As shown in Fig. 2, generating a product profile consists of three main parts: opinion mining, product feature classification, and product kernel design aspect ranking. Firstly, through opinion mining, we extract a large number of customer opinions from online reviews. And then we construct the product hierarchical structure by classifying these product features to the predefined kernel aspects. Finally, we generate our quantitative product profile by the model obtaining the importance weights of these product kernel aspects. The overall approach is further explained as follows:

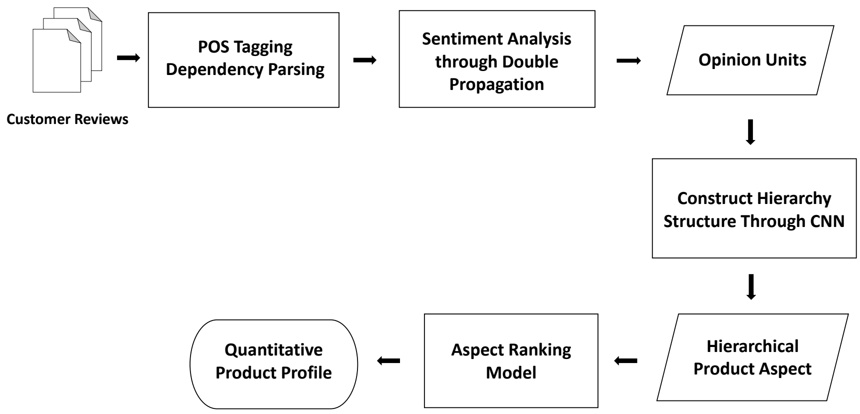

Figure 2: Overall approach for generating the quantitative product profile.

1. Sentiment analysis on customer reviews. In order to effectively express the customer opinions to the specific product feature, we define an opinion unit [2] as triple which has three elements: product feature, customer opinion of feature, and emotional polarity. In order to extract opinion units from customer reviews, we develop an unsupervised learning method which just needs seed words to start the bootstrapping process. These product features in opinion units form the base of the product profile structure.

2. Product feature classification. After obtaining thousands of opinion units from customer reviews, we build the product feature hierarchic structure by the product feature classification. Due to the lack of annotation data for training neural networks, we use distant supervision method [9] to extract the sentences containing the labeled product features from customer reviews. And then we use these annotated sentences to train our convolutional neural networks. After classifying product features to different product kernel design aspects, we can get the hierarchical structure of product profile.

3. Product kernel aspect ranking. We generate the relative importance of product kernel design aspects based on the overall customer opinions and the online sales of the specific product. In order to obtain the overall customer opinions in each design aspects, we aggregate the opinions polarity in the opinion units under different product kernel design aspects. And then we develop a probabilistic aspect ranking model to infer the importance of each kernel aspect, which considers the online sales and the overall customer opinions to specific aspect.

§4 Sentiment Analysis

In this paper, We define an opinion unit [2] as triple which has three elements: product feature, customer opinion of feature, and emotional polarity (+1 for positive polarity and -1 for negative polarity). For example, As shown in Fig.4, we can extract opinion unit '<dual camera, great, +1>' from the sentence '*the dual camera is great*'. In order to extract opinion units from customer reviews, we propose a double propagation method, which not only extracts product features but also extracts opinion words with their emotional polarity. However, a large number of noisy words are contained in the product features of opinion units. In order to filter out these noise, we develop a product feature identification approach based on n-gram model.

4.1 Opinion Units Extraction

Similar to previous works, we consider product features as nouns or noun phrases, and customer opinions are adjectives. Sentence dependency parser can extract the syntactic relations between product features and opinion words in the sentence. According to their relations, we can iteratively extract new product features and customer opinions with extracted words through double propagation, which just needs the predefined small-scale seed words to start the bootstrapping process. The overall algorithm steps are shown in Fig. 3.

Firstly, we transform the sentences of online reviews into tree structures through sentence dependency parser. As shown in Fig. 4, each node on the tree represents a word with its part of speech. Each side represents the different kind of relationship between two words.

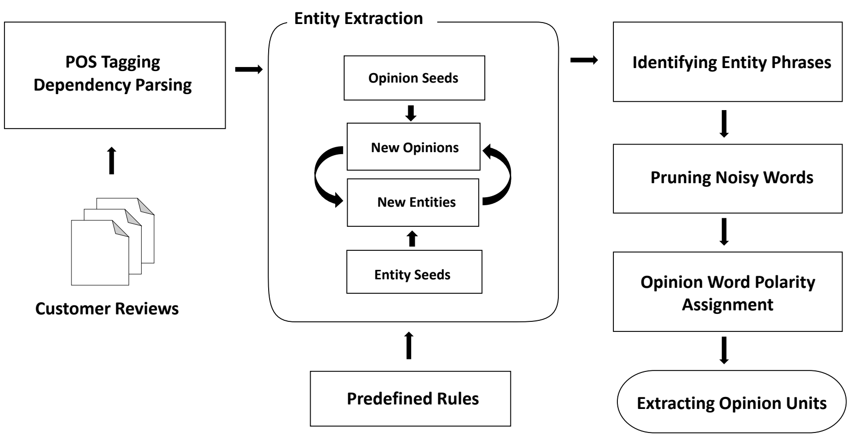

Figure 3: Opinion units extraction

Secondly, we extract product features and customer opinion words from online reviews through double propagation. The double propagation consists of four steps: (1) Extract new product features through opinion words. (2) Extract new product features through the extracted product features. (3) Extract new product features from the extracted opinion words. (4) Extract new opinion words through all opinion words (including extracted opinion words). We define several rules for target and opinion word extraction. Furthermore, we need to assign emotional polarity to the new extracted opinion words. According to our observation, we have two assumptions: (1) In a specific field, the polarity of the emotional word is consistent in different situation. (2) In a specific review, the customer emotional polarity to the product features remains the same. More details about extraction rules in double propagation and emotional polarity assignment can be found in [16]. Compared with Qiu's method, we add the product feature words as a seed lexicon, which gives more chance to extract new product features and opinion words.

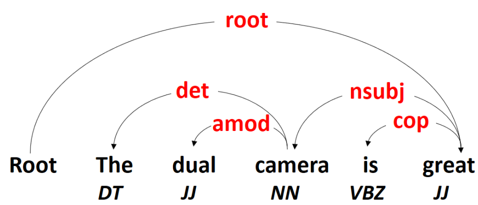

Figure 4: POS tagging and dependency parsing for an example customer product review.

Thirdly, we extract product feature phrases from the customer reviews. As we consider product features to be nouns phrases, we not only consider the relative position and syntactic relation between each other, but also consider the part of speech of each word. We merge the Q words around the target word while they meet the following conditions: (1) Q words around target word are related to each other (2) The relationship between Q words is in {mod, amod, subj, nsubj} (3) The part of speech of Q words belongs to {NN, NNS} or {JJ, JJR, JJS}. In this paper, we set Q=1. As shown in Fig.5, we get word '*dual*' through word '*camera*' and find them satisfy our rules. Thus, we extract '*dual camera*' as a single product feature.

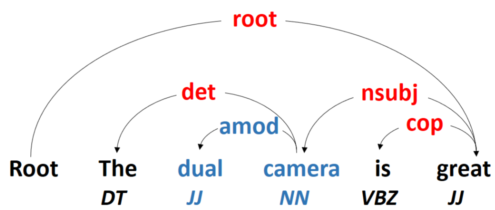

Figure 5: The word phrase 'dual camera' is extracted by our rules.

After product feature identification, we extract a large-scale product feature words set $\{F\}$

and customer opinion words set $\{O\}$. And then we loop through customer reviews to generate opinion units using $\{F\}$ and $\{O\}$. Opinion units extraction algorithm is shown in Alg.1. According to our observations, the emotional polarity of each product feature in a sentence only depends on the emotional polarity of the related customer opinions. Meanwhile, we consider the existence of negative words near the product features. The emotional polarity of the product feature would be reversed while the negative words appear in the word window. In the experiments, the word window size is set to 5. Finally, we use synonyms lexicon and similarity metric by cosine distance of word2vec to merge the feature words which have similar semantic meanings.

Algorithm 1 The opinion units extraction algorithm.

```
Input : Review Data R, Extracted Features {F} , Extracted Opinions {O}.
Output : Opinion Units.
For each pasred sentence S in R :
    For each feature word w in S :
        If w in F :
            If find nearby opinion word o in {O} :
                p = the polarity of o
                If no nearby negation/contrary word :
                    Extact opinion unit < w, o, p >
                Else :
                    Extact opinion unit < w, o, −p >
```

4.2 Product feature identification

Product feature identification can also be called product feature pruning. Although the double propagation algorithm based on sentence dependency parsing can extract a large number of product features, it also brings a lot of noise. Noisy words are usually generated by two cases:

1) The words customers used are irrelevant to the product, such as some names like Obama, Trump, or common words like happiness, pleasure.

2) Syntax errors caused by errors in sentence dependency parsing due to Chinese word segmentation. The algorithm mistakenly identifies adjectives as nouns, resulting in subsequent extraction errors. For example, good, very good.

We observe that the noisy words from the case 1 always have relatively lower frequency than the real product features. Therefore, we can set the frequency threshold to filter out the noise from the case 1. However, the noisy words from case 2 are often extracted from the similar structural sentences due to the systematic errors of sentence dependency parsing, which have the similar frequency with real product features. In order to filter out the noise from case 2, we develop a probabilistic model based on n-gram. Therefore, we manually annotate a high-frequency error lexicon, which contains the vast majority of product features that are misclassified. Then we train naive Bayes model to identify real product features, using these error lexicon as negative samples and the remaining words as positive samples.

The probability of a candidate product feature is a real product property which can be expressed as $p(w, c)$. C represents the classification condition 1 is true, 0 is not. W represents the product feature that need to be classified:

If $p(w, c = 1) > p(w, c = 0)$ means that the probability of the candidate product feature belongs to the real product feature is greater than the probability of the candidate product feature belongs to the noise. According to Bias's theorem, we can express it as the following formula:

$$p(w, c) = p(c) * p(w|c) \tag{1}$$

According to Markov chain, we can divide $p(wc)$ into three representations: unigram, bigram and trigram. They are expressed as follows:

$$p_{uni}(w|c) = \prod_{i=1}^{n} p(w_i|c) \tag{2}$$

$$p_{bi}(w|c) = \prod_{i=2}^{n} p(w_i, w_{i-1}|c) \tag{3}$$

$$p_{tri}(w|c) = \prod_{i=3}^{n} p(w_i, w_{i-1}w_{i-2}|c) \tag{4}$$

$$p(w_i|c) = \frac{C(w_i, c) + 1}{|M_c| + |V_1|} \tag{5}$$

$$p(w_i|w_{i-1}, c) = \frac{C(w_i, w_{i-1}, c) + 1}{|M_c| + |V_2|} \tag{6}$$

$$p(w_i|w_{i-1}w_{i-2}, c) = \frac{C(w_i, w_{i-1}, w_{i-2}, c) + 1}{|M_c| + |V_3|} \tag{7}$$

Here we use Laplace smoothing for $p(w_i|c)$, $p(w_i|w_{i-1}, c)$ and $p(w_i|w_{i-1}w_{i-2}, c)$ to avoid the final result of zero. Both of the above models can test whether the word is a product feature lexicon to a certain extent, but both models only consider the effect of a single word or two words on the result. Hence we introduce the weight parameters α, β, γ to fuse the three models. The final formula is as follows:

$$\begin{gathered} p(w, c) = p(c) * (\alpha * p_{uni}(w|c) + \beta * p_{bi}(w|c) + \gamma * p_{tri}(w|c)) \\ \alpha > 0, \beta > 0, \gamma > 0 \end{gathered} \tag{8}$$

§5 Quantitative Product Profile

In this section, we propose a Char-Word CNN model to transform these unstructured opinion units into hierarchical product profile. Based on our experience, we divide the product aspects into seven kernel design aspects, including usage scenario, target customer, functionality, quality, appearance design, price, service. These seven kernel aspects show the performance of different angles of products, which can effectively describe the customer's feedbacks.

Generally, opinion units have the following characteristics: (a) Each opinion unit belongs to only one kernel design aspect. (b) The product feature in the opinion units determines units' focus. These characteristics allow us to classify product features into seven kernel design aspect.

5.1 Generating Training Data

Product feature words have poor semantic information, so we can't use them directly as an input to the neural network model. Therefore, we have to get more training sets as input to the neural network model from customer reviews. In the relationship extraction task, Mintz et al. [9] use the knowledge graph to extract a large number of sentences containing the relationship entities as training samples. This method is called distant supervision. Similarly, inspired by the idea of distant supervision, we believe that these sentences in customer reviews containing the product feature words which we want to classify also provide some extra information for neural networks. Therefore, we extract these sentences as training data to the neural network model.

In addition, we find customers prefer to use long sentences to express their opinions. Long sentences often contain more than one product feature word, which can cause a great interference to the neural networks, because the long sentence containing more than one product feature word will be repeatedly labeled into different categories. In order to solve this problem, we notice that customers would use space, comma and other punctuation marks as semantic intervals. Therefore, we split long sentences into short sentences by punctuation and space.

5.2 Input Representation

The inputs of our network are Chinese sentences which will be respectively split into char tokens and word tokens. In our method, we transform char and word tokens into low-dimensional vectors by looking up the pre-trained embeddings matrix and concatenate them as the input of neural networks. In our word-part of network, we have three embeddings as input vector including word embeddings (WEs), word part-of-speech embeddings (WPEs) and position embeddings (PEs). For example, we assume that the size of the WEs $d_w = 50$, the size of the PEs $d_p = 10$, and the size of the WPEs $d_{wp} = 15$. We connect the above three embeddings of all the words in the sentence, and use a vector to represent a sentence of length n:

$$v = v_1 \oplus v_2 \oplus \cdots \oplus v_n \tag{9}$$

where $\oplus$ represents the join operation, $v_i \in \mathbb{R}^d$ $(d = d_w + d_p + d_{wp})$.And the same as word-part of input, we have two embeddings including char embeddings (CEs) and position embeddings (PEs) as the input vector of our char-part of network.

5.2.1 Word/Char Embeddings

Word Embeddings is a distributed representation of words that transform each word token to real-valued vector. Each word has a unique word vector corresponding to it. We have an embeddings matrix $V_w \in \mathbb{R}^{d_w \times |V_w|}$ where V_w is a dictionary with a fixed size and d_w is a fixed word vector size. Similar to word embeddings, char embeddings is a distributed representation of chars that maps each char to real-valued vector. We have an embeddings matrix $V_c \in \mathbb{R}^{d_c \times |V_c|}$ where V_c is a dictionary with a fixed size and d_c is a fixed char vector size.

5.2.2 Position Embeddings

Similar to Zeng [19], position distance is the distance from the word to the product feature t that needs to be classified. For example, as shown in Fig.6, *battery life* is a product feature that we need to classify, so the distance to itself is zero. The distance from '*terrible*' to '*battery life*' is 2.

Figure 6: The example for sentence position label.

Then we map the relative distance into a real-valued vector by looking up the embedding matrix $V_p \in \mathbb{R}^{d_p \times |P|}$, where P is a fixed-size distance set and d_p is a fixed vector size.

5.2.3 Word Part-of-speech Embeddings

In text classification, we find that the part-of-speech tagging of words implies a lot of semantics, and the different part-of-speech means the different impact on the classification results. For example, the meaning of nouns is much richer than the meaning of prepositions.

WPF represents the part-of-speech tag for each word in each sentence. We convert the part-of-speech tag to a real-valued vector by looking up the embedding matrix $V_{wp} \in \mathbb{R}^{d_{wp} \times |WP|}$, where $|WP|$ is a fixed-size part of speech tag set and d_{wp} is a fixed vector size.

5.3 Char-Word Hybrid CNN Model

5.3.1 Basic CNN Model

Inspired by the typical CNN text classification model proposed by Kim [3], first we transform each words in the sentence with its corresponding word vector. A sentence of length l is represented as $A \in \mathbb{R}^{l \times k}$.

$$A = [x_1, x_2, , x_i, \cdots, x_l]^T \tag{10}$$

Where $x_i \in \mathbb{R}^k$ is the i-th word 'k'-dimensional real-valued vector in a corresponding sentence. In the second step, the convolution operation is applied to a word window of size h to generate new features. The convolution operation is to multiply a convolution kernel $w \in \mathbb{R}^{h \times k}$ with the n-gram in each sentence and obtain a new feature c_i. The specific calculation formula is as follows:

$$c_i = f(w * x_{i:i+h-1} + b) \tag{11}$$

Where $b \in \mathbb{R}$ is bias term, f is non-linear function, and $x_{i:i+h-1}$ represents the concatenation of h word vectors $x_{i:i+h-1} = x_i \oplus x_{i+1} \oplus \cdots x_{i+h-1}$.

In the third step, the maximum pooling operation finds the maximum value on each c_i. Finally, fully connected layers and a sotfmax layer are used to output the final probability.

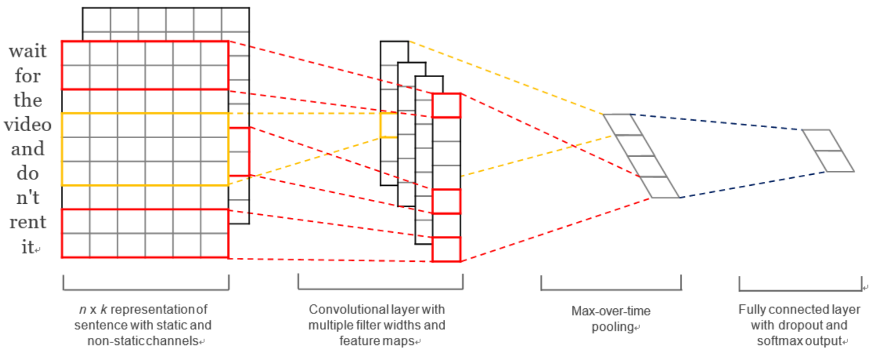

Figure 7: Basic CNN model architecture for an example sentence.

5.3.2 Char-Word CNN Model

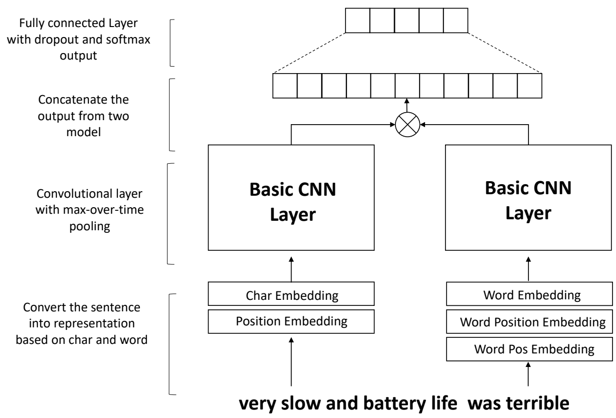

Figure 8: Char-Word CNN model architecture for an example sentence.

We propose a Char-Word CNN model based on words' and chars' features. As the structure of network shown in Fig. 8, we use two convolutional layer with max-over-time pooling, which respectively extract char-related and word-related features. Char-related embeddings include CEs and PEs. Similarly, word-related embeddings include WEs, WPEs and PEs. After the convolution operation, we concatenate the outputs of the two CNN layers as the input of the last layer [7]. The last layer is the full connection layers and the softmax layer. We insert dropout [1][4] modules in the fully-connected layers to regularize.

§6 Product Aspect Ranking Method

In this section, based on product sales and customer opinions, we propose a probabilistic product aspect weight regression algorithm to learn the importance of aspects from customers' perspective from custoners' perspective. The approach is mainly based on previous study[20] which is extended here via further consideration of the products' histrory sales. We observe from real life that customers' real thoughts are not show up well from the overall rating of products, because the online merchant tends to encourage customers to rate higher scores, even fake rates. However, the amount of product sales can reflect the customer preference for the product. Therefore, we have the following assumptions: (a) customer opinions on these aspects greatly influence the amount of sales. (b) The overall opinion to specific aspect in the product is an aggregation of the opinions given to specific aspects in the review of product. (c) Different aspects of various products have different contributions in the aggregation.

To model such aggregation, we formulate that the overall opinions O in each product is generated based on the weighted sum of the opinions on the specific aspects. After the sentiment analysis, we can get the word frequency matrix W which is normalized for each product aspect and the word sentiment polarities β in each review. We can calculate the overall opinions for the product aspect as:

$$O_i = \frac{1}{n}\sum_{j=1}^{n}\sum_{w\in A_i, w\in R_j}\beta_{jw}W_{iw} \tag{12}$$

Where O_i is the overall opinions for the *i-th* product aspects, n is the amount of product reviews, R_j is the *j-th* product reviews , A_i is the set of product feature words on the *i-th* aspect , $\beta_{jw} \in \{1, -1\}$ is the word sentiment polarity of word w in the *j-th* reviews and $W_{iw} \in [0, 1]$ represents the word-frequency weight of the word w in the *i-th* aspect.

Then, we can get the overall opinions based on the weighted sum of α_i and O_i, as $\sum_{i=1}^{k}\alpha_i O_i$ or in matrix form as $\alpha_p^T O_p$. The weight on the product aspect α_i means the importance of the aspect for the product. The larger α_i is more important and vice versa. The amount of product sales is assumed to satisfy the Gaussian distribution, with mean $\sum_{i=1}^{k}\alpha_i O_i$ and variance δ^2 as:

$$S_p \sim N(\sum_{i=1}^{k}\alpha_i O_i, \delta^2) \tag{13}$$

According to our knowledge, the kernel aspect of products in the same market which consumers are concerned about should be roughly same. For example, in the electric kettle market, customers' first concern is the quality rather than the appearance. This illustrates the dependence of the weight distribution between products aspects in the same market. However, the weight of different products aspects in the same market also has randomness due to diversity of reviewer's preference. Customers may choose to purchase based on price, appearance, service and other factors. Furthermore, to capture the dependence and randomness among different aspects, we employ a multivariate Gaussian distribution as the prior for aspect weights as:

$$\alpha_p \sim N(\mu, \Sigma) \tag{14}$$

Where μ, Σ represent the mean and variance parameters.

Combining Eq (13) and (14), the probability of the amount of product sales S_p in product p is given as:

$$\begin{aligned} P(S_p|d) &= P(S_p|\mu, \Sigma, O_i) \\ &= \int p(S_p|\alpha_p^T O_p, \delta^2)p(\alpha_p|\mu, \Sigma)d\alpha_p \end{aligned} \tag{15}$$

Where O_p are the overall opinions on the product calculated through Eq. (13) and $\{\delta^2, \mu, \Sigma\}$ is the corpus-level parameters, which can be estimated from reviews corpus and product sales. Therefore $\{\delta^2, \mu, \Sigma\}$ is independent from α_p which is the weight for a product p. Zha[20] proposed the EM-style algorithms to optimize their model. In this paper, the main difference between our model and Zha's model is we consider the influence of product online sales instead of product ratings on product aspect weights. Therefore, as shown in Alg.2 ,we also apply the EM-style optimization algorithms because the optimized parameters is very similar.

Given $\{\mu, \Sigma, \delta^2\}$, optimize α_p:

With the given parameters $\{\delta^2, \mu, \Sigma\}$, we use the maximum a posteriori (MAP) estimation to optimize the value of α_p. The object function of MAP estimation for product p is defined as:

$$\hat{\alpha}_p = argmax[-\frac{(S_p - \alpha_p^T O_p)^2}{2\sigma^2} - \frac{1}{2}(\alpha_p - \mu)^T \Sigma^{-1}(\alpha_p - \mu)] \tag{16}$$

s.t.

$$\sum_{i=1}^{k} \alpha_{pi} = 1, \qquad 0 \leq \alpha_{pi} \leq 1 \quad for \quad i = 1, 2, 3, \cdots, k \tag{17}$$

We apply the conjugate-gradient-interior-point method optimize α_p to solve the above-mentioned constant nonlinear optimization problem and get following formula the derivatives with respect to α_p:

$$\frac{\partial L(\alpha_p)}{\partial \alpha_p} = -\frac{(\alpha_p^T O_p - S_p)O_p}{\delta^2} - \Sigma^{-1}(\alpha_p - \mu) \tag{18}$$

Given α_p, optimize $\{\mu, \Sigma, \delta^2\}$:

Given α_p, we optimize the parameters $\{\delta^2, \mu, \Sigma\}$ using the maximum-likelihood (ML) estimation over the products corpus P. The log-likelihood function on the whole set of products is:

$$\hat{\theta} = \arg\max_{\theta} \sum_{p \in P} \log p(\alpha_p | \mu, \Sigma, \delta^2) \tag{19}$$

We take the derivative of $L(P)$ with respect to each parameter in $\{\delta^2, \mu, \Sigma\}$, and let it equals to zero which lead to following updating formulas:

$$\mu_{t+1} = \frac{1}{|D|} \sum_{p \in D} \alpha_p \tag{20}$$

$$\Sigma_{t+1} = \frac{1}{|D|} \sum_{p \in D} (\alpha_p - \mu_{t+1})(\alpha_p - \mu_{t+1})^T \tag{21}$$

$$\delta^2 = \frac{1}{|D|} \sum_{p \in D} (S_p - \alpha_p^T O_p)^2 \tag{22}$$

We repeat the above two optimization steps until the model converges. Finally, we compute overall weight on the special aspect by following formulas:

$$\alpha_i = \frac{\sum_{p \in P} \alpha_{pi} * S_p}{\sum_{p \in P} S_p} \tag{23}$$

Where S_p is the amount of product sales and α_{pi} is the *i-th* aspect of the product. The amount of products sales S_p is added as the weight for α_p because we think the product has more influence on the overall opinions with the larger of its amount of product sales.

Algorithm 2 EM-style algorithm for aspect weight learning

1: *Input* : Review Data R, Extracted Features $\{F\}$, Extracted Opinions $\{O\}$.
2: *Output* :Overall Weight $\alpha_i|_{i=1}^{k}$ for the k aspects.
3: While not converged do :
4: Update $\{\alpha_p\}_{p=1}^{n}$ according to Eq(19).
5: Update $\{\mu, \Sigma, \delta^2\}$ according to Eq(22)(23)(24).
6: Compute Overall Weight $\{\alpha_i\}_{i=1}^{k}$

§7 Evaluations

In this section, we will conduct extensive experiments to evaluate the effectiveness of our proposed approach for constructing quantitative product profile, including product feature extraction and identification, product feature classification, and product aspect ranking method.

7.1 Experimental Datasets

Table. 1 shows the details of our product review corpus, which are crawled from Taobao, the largest online shopping platform in China. We have crawled 249 products under blender machine market, 745 products under the electric kettle market, and also collected their sales data in Taobao during this period. In addition, we have two annotators to annotate the product features in each review and also label which kernel aspect product features belong to as well.

Table 1: Evaluation corpus statistics.

Category ID	Category Name	Product Amount	Average Number Of Reviews Per Product	Average Number Of Words Per Product
50003695	Electric Kettle	743	3083	2636193
50012097	Blender Machine	249	2287	2839732

In this paper, F_1-score, accuracy and recall rate are used as evaluation metrics in product feature identification and classification. F_1-score is a combination of precision and recall rate, as $F_1 = 2 * precision * recall/(precision + recall)$. We use NDCG@k [12] to evaluate the product aspect ranking results. Given a ranking list of product aspects, NDCG@k can be calculated as:

$$NDCG@k = \frac{1}{Z}\sum_{i=1}^{k}\frac{2^{t(i)}-1}{\log(1+i)} \tag{24}$$

where $t(i)$ is the importance degree of the product aspect at position i, and Z is the normalization term derived from the *top-k* product aspects of a perfect ranking. The rank of the product aspects is highly abstract and difficult to judge, so we ask the two annotators to rank the product aspects separately after reading all the customer reviews. Each product aspect has two rank numbers from two annotators. And then we sort the product aspect again by comparing the average of two rank numbers in each product aspect, which we regard as the ground truth to evaluate our product aspect ranking approach.

In the product feature classification experiments, each product feature has many sentences extracted from online reviews by the distant supervision which leads to the problem we may get the inconsistent classification by different sentences. In order to get the final class of the product features, we design a method based on product features classification frequency. For example, the product feature f_i has $Text_{f_i} = [l_1, l_2, \cdots, l_n]$ which is extracted from customer reviews by the distant supervision. And the classifications of the neural networks corresponding to these inputs are $Y'_{f_i} = [y'_1, y'_2, \cdots, y'_n]$. We choose the most frequent classification in Y'_{f_i} as the predicted class of product features.

7.2 Evaluations Of Product Aspect Identification

In this paper, we employ the HanLP [1] tool to do the POS tagging and sentence parser. Fig. 10 shows the examples of extracted feature phrases from our double-propagation approach in the customer reviews of electric kettle market.

[1] https://github.com/hankcs/HanLP

professional qualities	**low cost**	**thermal flask**
personal opinion	**price index**	**vacuum flask**
product price	**cost performance**	**after-sale service**
product package	**price level**	**commodity quality**
product quality	**service life**	**safety measures**
product safety	**easy operation**	**outer packing box**
product descriptions	**product specification**	**response speed**
product design	**outer packing box**	**response time**
technological level	**volume of product**	**safety factor**
household items	**customer service**	**work efficiency**

Figure 9: Examples of extracted feature phrases from the double-propagation approach.

In the experiment of product features identification, we compare the proposed product features identification algorithm with the following three methods: (a) Frequency-based method (Freq), which identifies product features according to their frequency. The minimum frequency of identifying product features is set to 100. (b) Unigram-based method (Unigram), which uses the unigram model to identify the rest product features after filtering by frequency-based method. (c) Bigram-based method (Bigram), which uses the bigram model to identify the rest product features after filtering by frequency-based method. (d) Trigram-based method (Trigram), which uses the trigram model to identify the rest product features after filtering by frequency-based method.

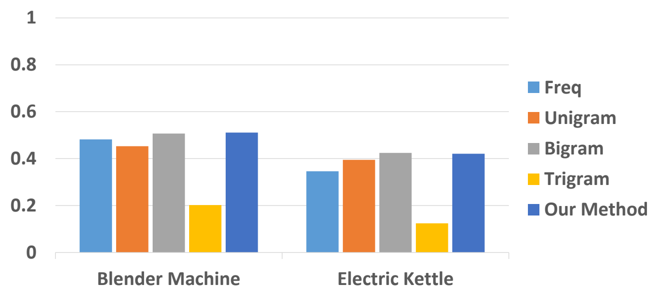

Figure 10: F1-score in product aspect identification.

In order to evaluate our product feature identification methods, we collect a large number of noisy words (incorrect product feature words) and product feature words as a training data set from dictionary and manual annotation which include a total of 14451 product features words and 13344 noisy words. In this paper, we set $\alpha = 0.5, \beta = 0.3, \gamma = 0.2$.

From Fig.11, Our method achieves the best results in the dataset of blender machine market, 5.1% higher than the Freq method, 12% higher than the Unigram method, and 0.8% higher than the Bigram method. Our method achieves an approximate result with Bigram on the dataset of electric kettle. As Fig.12 shown, Our method achieved the best results on both datasets, which are respectively outperforms Freq method by over 38.2% and 41%. Overall, our method achieved the best results on both precision and F1, and the Freq method achieves the highest performance on recall. Trigram method gets the worst on F1, precision and recall. In fact, we always concern more about the accuracy of identifying product features, because identifying product features can greatly reduce the manual operations cost accurately. Hence this proves the validity and practicality of our method in identifying product features.

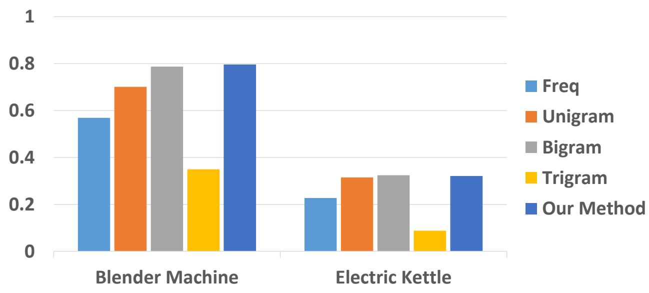

Figure 11: Precision in product aspect identification.

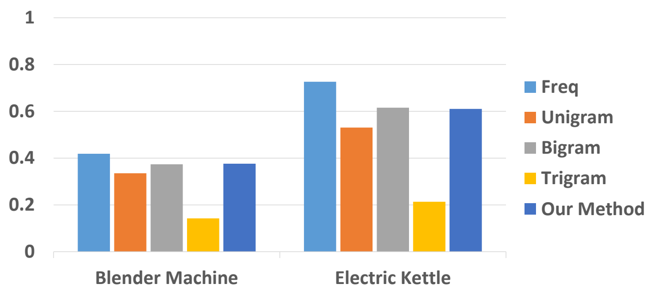

Figure 12: Recall in product aspect identification.

7.3 Evaluations of Aspect Classification on Product Profile

In order to evaluate the effectiveness on product feature classification, as shown in Table 2, we compare our method with four models, all of which use the CNN model proposed by Kim [3] with various chosen embedding as the input of CNN. The parameters of these models are the same. Following Kim [3],the number of convolution kernels in the convolutional layer is 3, and each convolution kernel is 50. By the way, We set their windows are 2, 3, and 4 respectively. Other parameters we choose are: dropout rate of 0.5, mini-batch size of 50 and learning rate of 0.001.

Table 2: Different input representation for aspect classification.

Char Embedding	Char Position Embedding	Word Embedding	Word Position Embedding	Word Pos Embedding	Approach Name
√					Char CNN
√	√				Char-Plus CNN
		√			Word CNN
		√	√	√	Word-Plus CNN
√	√	√	√	√	Our method

Results of comparison between our model and other methods are listed in Table 3. Char-Plus CNN approach significantly outperforms Char CNN in two datasets by over 39.3% and 11.5% respectively. It shows that the CPEs can effectively improve the performance of the model. Furthermore, Word-Plus CNN improves the performance over Word CNN on the two data sets by over 25.4% and 2.5% respectively, which indicates that the WPEs have also greatly helped the neural network to extract more effective features from text. Above all, the performance of our method is better than other methods in the two data sets. Hence, these results suggest that our approach can boost the performance of product feature classification.

Table 3: Performance of product aspect classification.

Accurancy	Electric Kettle	Blender Machine
Char CNN	0.3351	0.6717
Char-Plus CNN	0.4671	0.7496
Word CNN	0.4367	0.6880
Word-Plus CNN	0.5476	0.7058
Our Method	**0.5739**	**0.8316**

7.4 Evaluations of Product Aspect Ranking

Table 4: Performance of aspect ranking in terms of NDCG@5 and NDCG@7.

	Electric Kettle		Blender Machine	
Method	NDCG@5	NDCG@7	NDCG@5	NDCG@7
Frequency-Based	0.6746	0.8746	0.6831	0.7085
Our Aspect Rank	**0.7929**	**0.8862**	**0.7717**	**0.9074**

In this experiment, we compare our method with frequency-based sorting method, which ranks the aspects according to aspect frequency. Table.4 shows the comparison results in terms of NDCG@5, NDCG@7 respectively. Our method achieved better results on both data sets than the frequency-based method in terms of NDCG@5 by over 11.82% and 8.86%. At NDCG@7, our method is also superior to the frequency-based method by over 1.16% and 19.89% respectively. On average, the proposed aspect ranking approach significantly outperforms frequency-based. The frequency-based approach only considers the frequency of aspects in online reviews, and neglects the influence of the customer opinions on the specific product aspect. Our approach takes into account the overall customer opinions and the amount sales of the product, which greatly help us capture the information from customer opinions.

§8 Conclusions

In this paper, we propose a novel approach for generating quantitative product profile which provides a clear description on product's multiple aspects. The approach is consists of three main steps. First, the double propagation strategy extracts customer opinions as opinion units from online reviews. Second, we propose a hybrid char-word convolutional neural network to classify and build the aspect hierarchy. In order to cover the lack of labeled data, we extract sentences containing labeled feature words from the idea of distant supervision. Finally, the probabilistic product ranking model infers the importance of product kernel design aspects based on the associations between product sales and customer opinions. From the result of experiments, we can see that the hybrid CNN model preciously classifies the product features and constructs the aspect hierarchy. And through the aspects ranking method, the quantitative product profile expresses well the overall customer opinions on online reviews.

References

[1] G E Hinton, N Srivastava, A Krizhevsky, I Sutskever, R R Salakhutdinov. *Improving neural networks by preventing co-adaptation of feature detectors*, arXiv preprint arXiv:1207.0580, 2012.

[2] Hu M, Liu B. *Mining and summarizing customer reviews*, Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August, DBLP, 2004:168-177.

[3] Kim Y. *Convolutional Neural Networks for Sentence Classification*, Eprint Arxiv, 2014.

[4] Qin F, Gao N, Peng Y, et al. *Fine-grained leukocyte classification with deep residual learning for microscopic images*, Computer methods and programs in biomedicine, 2018, 162: 243-252.

[5] Li F, Han C, Huang M, et al. *Structure-aware Review Mining and Summarization*,International Conference on Computational Linguistics, Association for Computational Linguistics, 2010:653-661.

[6] Liu Y, Bi J W, Fan Z P. *Ranking products through online reviews: A method based on sentiment analysis technique and intuitionistic fuzzy set theory.* Information Fusion, 2017, 36:149-161.

[7] Qin F W, Bai J, Yuan W Q. *Research on intelligent fault diagnosis of mechanical equipment based on sparse deep neural networks*, Journal of Vibroengineering, 2017, 19(4):2439-2445.

[8] Mikolov T, Sutskever I, Chen K, et al. *Distributed Representations of Words and Phrases and their Compositionality*, Advances in Neural Information Processing Systems, 2013, 26:3111-3119.

[9] Mike Mintz, Steven Bills, Rion Snow, Dan Jurafsky. *Distant supervision for relation extraction without labeled data*, Joint Conference of the Meeting of the ACL and the International Joint Conference on Natural Language Processing of the AFNLP: Volume 2, Association for Computational Linguistics, 2009:1003-1011.

[10] Mnih V, Heess N, Graves A, et al. *Recurrent models of visual attention*, 2014, 3:2204-2212.

[11] Qiu G, Liu B, Bu J, et al. *Opinion word expansion and target extraction through double propagation*, Computational Linguistics, 2011, 37(1):9-27.

[12] Kalervo Jarvelin, Jaana Kekinen. *Cumulated gain-based evaluation of IR techniques*, ACM, 2002.

[13] Snyder B, Barzilay R. *Multiple Aspect Ranking using the Good Grief Algorithm*, Human Language Technologies 2007: the Conference of the North American Chapter of the Association for Computational Linguistics. 2007:300–307.

[14] Wang H, Lu Y, Zhai C. *Latent aspect rating analysis on review text data:a rating regression approach*, ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2010:783-792.

[15] Wang W, Pan S J, Dahlmeier D, et al. *Coupled Multi-Layer Attentions for Co-Extraction of Aspect and Opinion Terms*, AAAI, 2017: 3316-3322.

[16] Wu Y, Zhang Q, Huang X, et al. *Phrase dependency parsing for opinion mining*, Conference on Empirical Methods in Natural Language Processing: Volume 3, Association for Computational Linguistics, 2009:1533-1541.

[17] Yang Z, Yang D, Dyer C, et al. *Hierarchical Attention Networks for Document Classification*, Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2017:1480-1489.

[18] Zaremba W, Sutskever I, Vinyals O. *Recurrent Neural Network Regularization*, Eprint Arxiv, 2014.

[19] Zeng D, Liu K, Lai S, et al. *Relation classification via convolutional deep neural network*, 2014.

[20] Zha Z J, Yu J, Tang J, et al. *Product Aspect Ranking and Its Applications*, IEEE Transactions on Knowledge and Data Engineering, 2014, 26(5):1211-1224.

[21] Zhang M, Zhang Y, Vo D T. *Gated Neural Networks for Targeted Sentiment Analysis*, AAAI, 2016: 3087-3093.

[22] Zhao Z, Wu Y. *Attention-Based Convolutional Neural Networks for Sentence Classification*, INTERSPEECH, 2016:705-709.

[1] Department of State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou 310027, China. Email: eming.li@gmail.com

[2] School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310027, China.