A Study of Opinion Classification on Review Data

OU Wei

Japan Advanced Institute of Science and Technology

Doctoral Dissertation

A Study of Opinion Classification on Review Data

OU Wei

Supervisor: Associate Professor Van-Nam Huynh

School of Knowledge Science
Japan Advanced Institute of Science and Technology

December, 2018

Abstract

Nowadays, along with the rapid growth of the e-commerce economy, consumers have become more and more dependent upon review data to make decisions on shopping websites. However, reading reviews is a very time-consuming, even frustrating process when the number of reviews is overwhelmingly large. Both academia and industry have been working to devise algorithms that can automatically extract knowledge of products or services from the review data to improve users' review-reading experience. The knowledge extraction is usually treated as a sentence-level opinion classification problem, that is to detect what attributes of products or services consumers have discussed about, and how they felt of the attributes in the review sentences. This dissertation mainly focuses on the following 2 fundamental problems related to the classification task: sentence representations, the limited availability of training data. Also, based on the results of the opinion classification process, this dissertation proposes a novel machine learning based approach to identify the aspects that are generally attractive to consumers. The discovered attractive attributes allow users to quick capture the selling points of a product or service. A brief introduction to the originality of this dissertation is presented as follows.

1) Sentence representations. Word embeddings models, as an effective way to represent text, have been widely used in various text classification tasks. Since word embeddings are only optimised to represent individual words, one has to define ways to aggregate word embeddings to represent sentences. A very effective, easy-to-compute aggregation function is averaging, though it obviously leads to loss of information. Recently, researchers have applied complex, but also computationally expensive neural network structures, such as convolutional neural network (CNN) and recursive neural network (RNN) units, to aggregate word embeddings. This dissertation proposes a novel weighted average approach, named 'Abstract Keywords', as an alternative to the existing aggregation operators. The proposed approach assumes there exist some extremely important abstract keywords that can be derived in the training process, and assigns words different weights according to their semantic similarities to the abstract words. Each sentence is represented by the weighed average of the embeddings of all words in the sentence.

Experiment results show that the proposed approach is computationally efficient, and outperforms the simple averaging approach.

2) Limited availability of labelled training data. As an important aspect of review mining, sentence-level sentiment classification has received much attention from both academia and industry. Many recently developed methods, especially the ones based on deep learning models, have centred around the task. Generally speaking, training sentence-level sentiment classifiers requires training datasets of labelled sentences, that are usually every expensive to obtain. It is possible to use the less expensive labelled review documents to train sentence-level sentiment classifiers, by treating each document as a long sentence, and the label of the document as the label for the long sentence. However, this way is obviously questionable because there may exist sentences in a document whose sentiments are very different from the sentiment of the document. Therefore, the sentiments of individual sentences can be easily misrepresented by the document-level labels in the training process. To address the problem, we propose a novel approach, named 'Averaged-logits', that also uses labelled documents to train sentence-level sentiment classifiers, but makes a difference by assuming different sentences in a document have different sentiments, and the 'average' of the sentence-level sentiments is used to determine the document-level sentiment. In the experiment, we collected two review datasets: one contains 50,000 hotel reviews crawled from TripAdvisor, the other 50,000 reviews from Amazon. The proposed approach was evaluated on the two datasets. The results show that, the proposed approach outperforms the existing approach treating each document as a long sentence, by margins of 3%-8% on sentence-level sentiment classification .

3) Attractive attribute classifiers. Researchers have proposed statistical regression models that analyse on-line review data to identify attractive attributes of a product or service. This dissertation has the same aim, but with an approach based on machine learning models instead of statistical models. The proposed approach first extracts attribute-level sentiments from the review text by natural language processing techniques, then derives features that reflect the non-linear relations between attribute performance and customer satisfaction based on the sentiments. The non-linear features are fed to the Support Vector Machine (SVM) model to train predictive attractive attribute classifiers. The proposed approach is evaluated on a hotel review dataset crawled from TripAdvisor. The experiment results indicate that the classifiers reach a precision of 79.3% and outperform the existing statistical models by a margin of over 10%.

Keywords: Review data; Sentiment analysis; Sentence representation; Weakly-supervised learning; Kano model analysis

Acknowledgments

Foremost, I would like to express my sincere gratitude to my supervisor Prof. Van Nam Huynh from Japan Advanced Institute of Science and Technology (JAIST), for his assistance from all aspects. Thanks to his constant encouragement, patient guidance, unlimited knowledge base, I was able to keep moving forward and negotiated one obstacle after another. He means much more than a supervisor to me, he has offered me much more than teaching me about research.

I would like to express my gratitude to Chinese Scholarship Council (CSC) for my scholarship. With CSC's generous support, I was able to focus on my study without too much monetary worry. I would like to express my gratitude to JAIST for providing us with the super convenient living and research facilities.

I thank my fellow lab mates in Huynh's lab, in particular Tang, for their support and encouragement. The time I spent with you, including our spring trip to the sea, intense discussion at the weekly seminar, was extremely valuable, entertaining, educational. The memory will accompany me to embrace every beautiful day in the rest of my life.

I also thank the country, Japan, for her magnificent people and culture. 99%, if not 100%, of the things I have experienced in Japan are extremely positive and unforgettable.

Last but not the least, I would like to thank my family members for their love, understanding, support. Every time I feel tired, I go home, the harbour shields storm from me.

Thank you,

OU Wei

September 2018

Table of Contents

List of Figures

List of Tables

Chapter 1

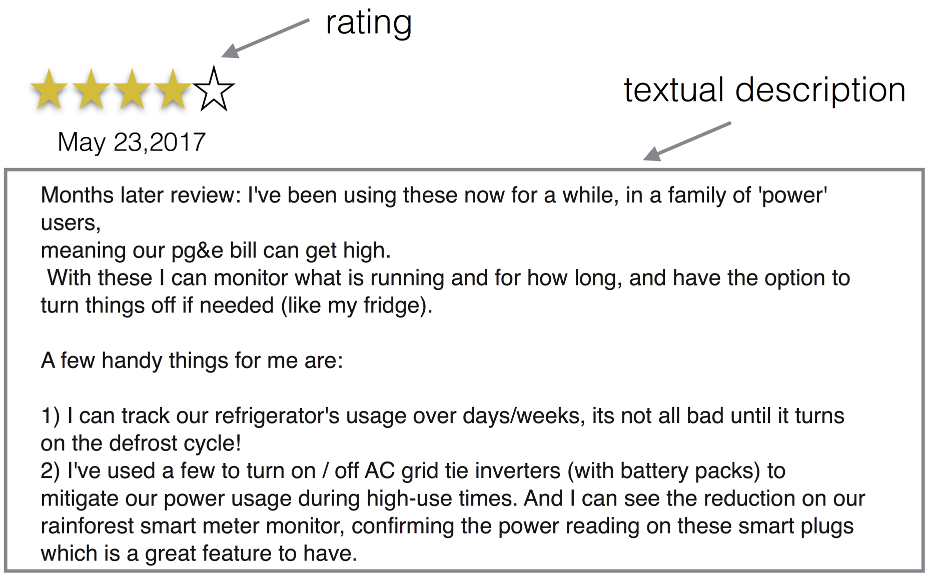
Introduction

This chapter introduces the research background, motivation, and the structure of this thesis.

1.1 Research background and motivation

In recent years, thanks to the popularisation of mobile devices, the popularity of on-line shopping has been raised to an unprecedented level. Various on-line shopping sites, such as Amazon, Alibaba, Rakuten, etc., have reported record-high growth in on-line sales volume. The sales growth prompts those companies to constantly increase the variety of their products or services, and expand the sizes and scales of their websites. Today, for each product or service category, they will be hundreds, even thousands of choices available on a major shopping site. For users, the more choices they have, on one hand, the higher the chances in making satisfying deals; on the other hand, the more difficult they will feel to navigate through the ocean of webpages.

Fortunately, on-line shopping sites host a natural wealth that can help users effectively locate the products or services fitting their needs: review data. A review usually consists of a piece of textual description that details a user's evaluation on a product or service, and a numeric rating that summarises the user's overall feelings on it. A typical review is shown in Figure 1.1 . By reading reviews, people can gain a certain degree of knowledge of a target product or service to reduce the risk of a regretful purchase. Nowadays, on-line shoppers have become more and more dependent upon the review data for making


rating
textual description
May 23,2017

Months later review: I've been using these now for a while, in a family of 'power' users,
meaning our pg&e bill can get high.
With these I can monitor what is running and for how long, and have the option to turn things off if needed (like my fridge).

A few handy things for me are:

1) I can track our refrigerator's usage over days/weeks, its not all bad until it turns on the defrost cycle!
2) I've used a few to turn on / off AC grid tie inverters (with battery packs) to mitigate our power usage during high-use times. And I can see the reduction on our rainforest smart meter monitor, confirming the power reading on these smart plugs which is a great feature to have.

Figure 1.1: A toy example of reviews

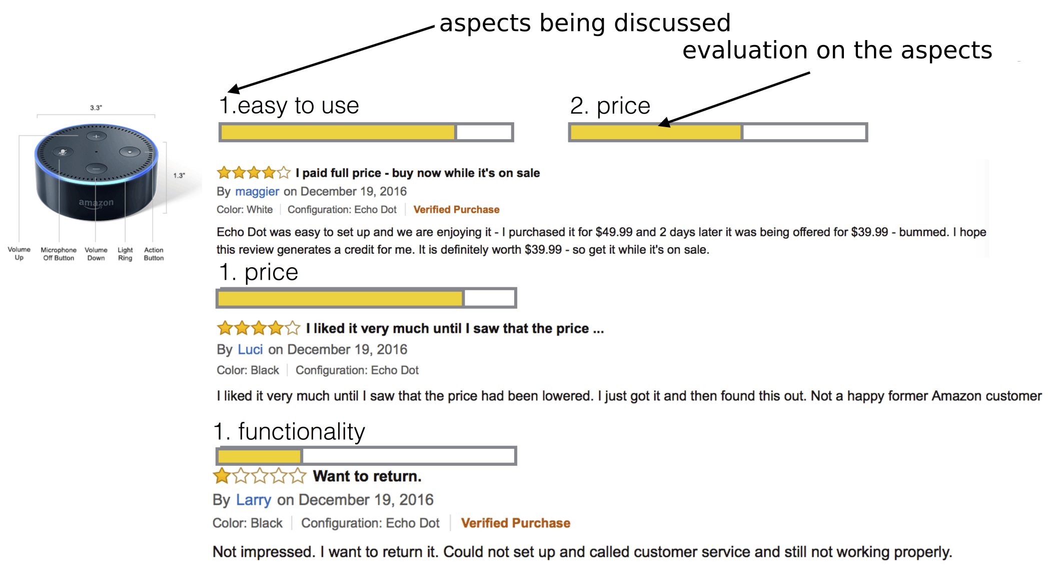
decisions on shopping websites.

However, reading reviews is very a time-consuming, confusing process when the number of reviews is overwhelmingly large. To ease the review-reading process of users, researchers have been working to devise powerful computer algorithms that can automatically extract knowledge from the review data. One of the topics related to the efforts, is review summarisation, that aims to summarise what product or service aspects users talk about, and how they feel of them in all the reviews for each product or service [1–4]. A pictorial demonstration of review summarisation is shown in Figure 1.2 .

Review summarisation is usually treated as a text classification problem: understanding what aspects are discussed in reviews is treated as an aspect (or topic) classification problem; understanding users' feelings on each aspect is a sentiment classification problem. For conventional documents, such as news articles, book chapters, etc., the two classification tasks are well-studied topics with a long research history [5]. However, for review documents that are usually short and full of informal use of language, the classification tasks are particularly challenging.

The challenges mainly lie in the following 3 aspects:

- Document representations. Traditional bag-of-words based document representation models, such as one-hot encoding, term frequency-inverse document frequency(tf-idf), [6] etc., do not work well on review data because of the short length and noisy



I paid full price - buy now while it's on sale
By maggier on December 19, 2016
Color: White | Configuration: Echo Dot | Verified Purchase
Echo Dot was easy to set up and we are enjoying it - I purchased it for $49.99 and 2 days later it was being offered for $39.99 - bummed. I hope this review generates a credit for me. It is definitely worth $39.99 - so get it while it's on sale.

I liked it very much until I saw that the price ...
By Luci on December 19, 2016
Color: Black | Configuration: Echo Dot
I liked it very much until I saw that the price had been lowered. I just got it and then found this out. Not a happy former Amazon customer

Want to return.
By Larry on December 19, 2016
Color: Black | Configuration: Echo Dot | Verified Purchase
Not impressed. I want to return it. Could not set up and called customer service and still not working properly.

Figure 1.2: A pictorial demonstration of review summarisation

nature of the data. Also, in opinion classification, semantic information usually plays critical roles. Unfortunately, the traditional representation models cannot encode the information.

- Limited availability of labelled data. Review summarisation involves sentence-level aspect and sentiment classification. Currently, methods for the classification tasks are usually based on supervised machine learning models. To use the supervised learning models, datasets of review sentences annotated with ground-truth sentiment and aspect labels have to be supplied as the training data. The data annotation process involves intense human labor even linguistic expertise, therefore, the availability of the training data is usually very limited.

- Encoding the non-linearity of the data. Review data is full of comparative and shifting opinions, therefore, the boundaries separating different opinion classes should be highly non-linear. Traditional learning models, such as SVM, Naive Bayes, Logistic regression, etc., are not strong enough to capture the non-linearity [7].

1.2 Research scope and general methodologies

This dissertation mainly focuses on the aforementioned problems in the classification tasks. This dissertation explores a number of recently developed techniques for possible solutions to the problems. In this dissertation, text embedding models [8,9] are used as an alternative to the traditional bag-of-words models to address the document representation problem; a semi-supervised learning model is proposed to reduce the intense manual labor for data annotation; a number of deep learning structures are adopted as the building blocks of the opinion classifiers used through out this dissertation to better encode the non-linearity of the review data

Furthermore, inspired by the recent progresses in precision marketing [10–12], this dissertation aims to build machine learning classifiers that can identify the 'specialties', or attractive aspects of a product or service based on the discovered opinions. The attractive aspects can be used as a supplementary indicator to the review summaries to help consumers quick capture the selling points of a product or service.

1.3 Structure of the dissertation

The remaining of this dissertation is organised as follows.

- Chapter 2 presents a brief introduction to a number of building blocks of the opinion classifiers, such as text embedding models to represent review documents, the deep learning models to encode the non-linear relations between the textual content and the ground truth labels.

- Chapter 3 presents a novel weighted average approach to represent review sentences based on word embedding models. The difficulty of using word embedding models to represent the textual content of reviews lies in that, word embedding models are to represent individual words, not sentences or documents. To represent a review sentence, one has to combine the representations of individual words in the sentence. Researchers have used the average of the embedding vectors of individual words in a sentence as the representation of the sentence. Obviously, this approach leads to loss of information. In recent years, the deep learning models that can encode the spatial or sequential information of textual documents, such as convolutional neural network (CNN) and recurrent neural network (RNN), have been used as the operators to combine word embeddings. The disadvantage of these approaches is that the CNN and RNN models are usually very computationally expensive. The proposed approach assigns different words different weights according to their semantic meaning, and combine the word embedding vectors in a sentence by their weighted average. Evaluation results show that, the proposed approach is competitive to the CNN and RNN models in performance, but much computationally cheaper.

- Chapter 4 proposes a weakly-supervised structure for sentence-level sentiment classification based on the neural network models. Conventionally, to train a sentence-level sentiment classifier, labelled sentences have to be supplied as the training set. However, as previously mentioned, the labelled datasets are usually every expensive to obtain. The proposed weakly-supervised model uses ratings, instead of the manually annotated sentence-level sentiment labels, to train the sentence-level sentiment classifiers. Since the ratings are prevalently available in review data, no manual labor for data annotation is involved in the proposed approach.

- Chapter 5 proposes a machine-learning based approach to identify the attractive attributes of a product or services. According to the Kano theory [13], all the aspects of a product or service can be divided into 3 main categories: must-be attributes, one-dimensional attributes, attractive attributes. Must-be attributes are the minimum requirements customers expect, while attractive attributes usually serve as the 'bonus' to boost customer satisfaction. There are existing research works that use statistical models to identify the attractive attributes. However, so far, no previous work has used machine learning techniques to train predictive classifiers to identify the attractive attributes. The proposed model is to fill in the gap. A critical problem in training such a machine learning classifier is deriving features to encode the dynamics in the relations between customer sentiments on each individual attribute and the overall sentiments on the whole product or service. In the proposed approach, a novel neural network structure is proposed to model the relations.

- Chapter 6 summarises the contributions of this dissertation to the community of Knowledge Science, and provides a conclusion and future research directions.

Chapter 2

Background knowledge in the opinion classification

Generally speaking, most methods for sentiment and aspect classification are based on supervised machine learning models. The process to train such classifiers usually consists of the following steps: 1) preparing and pre-processing the labelled data; 2) numerically representing the data; 3) selecting the learning models; 4) fitting the models. This chapter presents a brief introduction to the background knowledge related to the 4 steps. Furthermore, at the end of this chapter, a brief introduction to the Kano model theory is presented as the prerequisite knowledge for chapter 5.

2.1 Preparing and processing the data

Supervised machine learning models [14] require labelled datasets for the training process. A labelled data sample usually consists of 2 parts: the feature vector (or matrix) and ground truth label. This dissertation mainly focuses on sentiment and aspect classification on review sentences, therefore, the ground truth labels are the aspects each sentence discusses about or its sentiment orientation. Obviously, the ground truth labels are not inherently available, and usually obtained by employing human experts to manually read the content of each sentence. Since a dataset may contain up to millions of sentences, laborious human efforts can be involved in the data preparation process.

Each English textual document may contain many decorative words, such as 'the', ‘at’, ‘be’, that have no or little semantic meaning. Therefore, according to the traditional natural language processing practices, those words have to be removed from the documents in the pre-processing [15]. Furthermore, there exist many compound words that consist of multiple words but have to be treated as a single word, such as ‘New York’, ‘front desk’, ‘king bed’, etc. It is imperative to identify the compound words in the pre-processing. A widely used approach to identify the compound words is checking whether there exists statistically significant difference between the probability distribution of a group of words being treated as a single compound word and the joint probability distribution of the words being treated as independent [16].

It is also possible to feed the data directly into the learning models without any pre-processing efforts to train the classifiers. However, a premise has to be met to skip the pre-processing step: the training size has to be large enough [17].

2.2 Representing the data

In traditional natural language processing practices, documents are usually treated as ‘bags of words’ [6]. The ‘words’ here are actually the n-grams, that are contiguous sequences of n words from a given document. Though theoretically n can be an arbitrary integer, in practice it is usually limited under 4. There is no order among the ‘words’ in the ‘bag’

Based on the bag-of-words model, the simplest way to represent a document is the one-hot encoding [6]. In this representation model, each dimension of a vector corresponds to each unique word in the vocabulary, and takes a binary value to code the occurrence or absence of the corresponding word in a given document.

Another widely used bag-of-words based representation model is the term frequency-inverse document frequency (tf-idf) model [6]. In the tf-idf vector of a given document, each dimension holds the weight of each unique word from the vocabulary in the document. The weight of each word in the document is proportional to the product of the term frequency of the word in the document and the inverse of the document frequency of the word in the entire corpus. The underlying argument of tf-idf is that the words frequently occur in many documents are usually the stop words, therefore, the weights of those words have to be suppressed by factoring in their inverse document frequencies.

Topic models [18] have also been used to derive the representation vectors of textual documents. In a representation vector generated by topic models, each dimension represents the membership of the document in each topic. The membership is derived based on a generative model that defines each topic as a probability distribution over all the unique words in the vocabulary, and each document as a probability distribution over all the topics.

All the three models have gained huge successes in various classification tasks on conventional textual documents. However, on short document classification, their performance is usually limited. When the corpus is large and the average document length is very short, the representations generated by tf-idf and one-hot encoding are very high-dimensional and sparse that can easily cause severe over-fitting problems; the representations generated by topic models are usually very noisy that can cause misrepresentation of the data [19].

In recent years, word embedding models [8] have been used as a powerful alternative to the traditional representation models in short text classification. Word embedding models generate low-dimensional and dense vectors to represent words, that are much easier to be handled by machine learning models. Each dimension of a word embedding vector represents the membership of the word in each abstract sense, therefore, it allows for computing the semantic similarity between any pair of words in the vocabulary. Currently, the most representative word embedding models are the word-to-vector (word2vec) [9] and global vectors (Glove) [20].

Word2ve is essentially a language model built upon shallow neural networks that predicts what a word may appear given a textual context, or what a textual context may appear next to a given word. The architecture to tackle the former task is called 'continuous bag-of-words', and the architecture for the latter task 'skip-gram'. A pictorial demonstration of the two architectures are shown in Figure 2.1 [9] . The representation of each word is derived by fitting the shallow neural network on the training corpus.

Penninton et al. [20] argued that both CBOW and Skip-gram fail to adequately incorporate the global statistics of word occurrences. They observed that the co-occurring frequency of a pair of words usually suggests the semantic similarity between them. The authors proposed the GloVec model that assumes the co-occurring frequency of a word

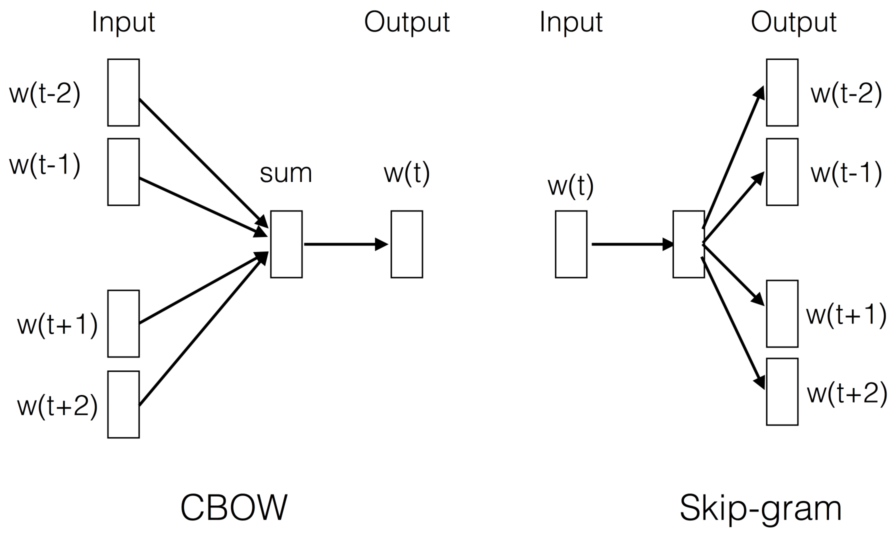


Figure 2.1: The structures of CBOW and Skip-gram

pair is somehow linearly related to their distance in semantic spaces, and treats the inference of the embedding vectors as a least-square regression problem.

2.3 Machine learning models

Traditional machine learning models, such as SVM, Logistic regression, Naive Bayes, etc., can be used to train the aspect and sentiment classifiers. In recent years, the neural network models have been proven more effective than the traditional learning models in a wide range of short text classification tasks [21]. This section presents a brief introduction to the 3 most frequently used neural network architectures: multi-layer perceptron (MLP), convolutional neural network (CNN), and recurrent neural network (RNN) [22].

A multi-layer perceptron consists of an input layer, an output layer, and at least a hidden layer. Assuming an input vector is x, the activations of the first hidden layer are computed as:

$$A^{[1]} = g(W^{[1]}x + b^{[1]}) \tag{2.1}$$

Where $W^{[1]}$ is the linear weight matrix, $b^{[1]}$ is the bias term for the first hidden layer, g is the activation function. Similarly, the activations of the nth hidden layer is computed as:

$$A^{[n]} = g(W^{[n]}A^{n-1} + b^{[n]}) \tag{2.2}$$

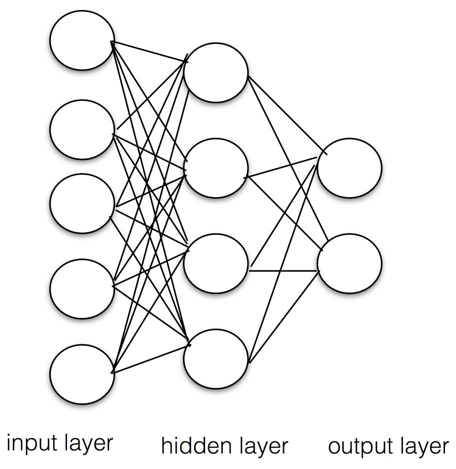


Figure 2.2: The structure of MLP

Where $W^{[n]}$ and $b^{[n]}$ are the linear weight matrix and bias term for the nth hidden layer, respectively, $A^{[n-1]}$ is the activations of the previous hidden layer.

The activations in output layer are computed as :

$$A^{[o]} = s(W^{[o]}A^{[o-1]} + b^{[o]}) \tag{2.3}$$

Where s is usually the softmax function, $A^{[o-1]}$ is the activations of hidden layer proceeding the output layer, $A^{[o]}$ can be interpreted as the probability distribution of the input sample over all the target classes. The cost of the sample is measured by the cross entropy between $A^{[o]}$ and the ground truth label y of the sample:

$$J = -\sum_{c=1}^{C} y_c * \log A_c^{[o]} \tag{2.4}$$

The architecture of MLP is shown in Figure 2.2

CNN models use convolution operators instead of the linear mapping in MLP to compute the activations. Assuming the input of a CNN model is x, x can be a vector or a matrix, the activations resulting from convolution operation j in the first hidden layer is computed as:

$$A^{[1]} = g(W_j^{[1]} \circledast x + b_j^{[1]}) \tag{2.5}$$

Where $\circledast$ is the convolution operator, $W_j^{[1]}$ and $b_j^{[1]}$ are the jth convolution filter and bias term, respectively. In each hidden layer, there can be an arbitrary number of convolution filters, and the activations resulting from all convolution operations are stacked to form a 3-d activation matrixes.

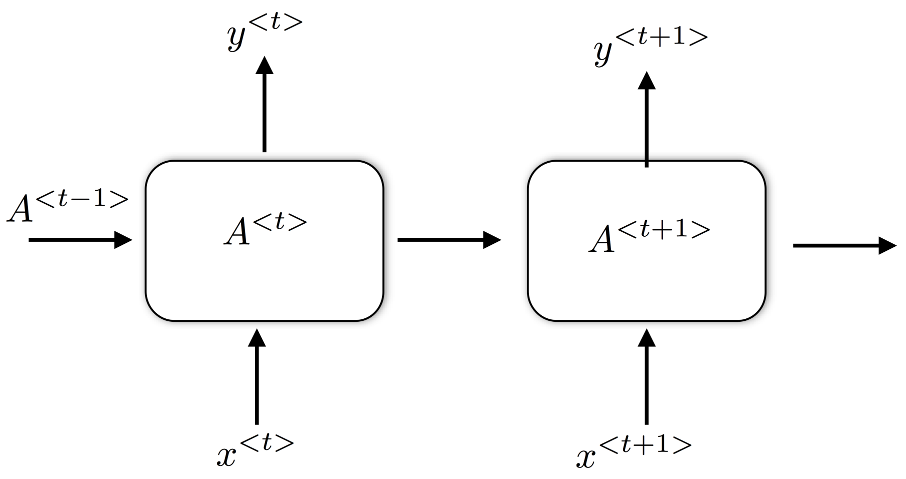


Figure 2.3: The structure of RNN

One more difference of CNN from MLP is that CNN always adopts the pooling operations to reduce the sizes of the hidden activations. Two types of pooling operations are frequently used in various applications: average pooling that replace all the activation values in each neighbourhood of a predefined size with the their average, and max pooling that replaces the activation values with their maximum.

RNN is to handle sequential data. Each input consists of a varying number of time steps. The activations [1]at each time step depend upon the activations of the previous time step. Assuming the activations of time step t are $A^{<t>}$, then the activations of time step $t+1$ are computed as:

$$A^{<t+1>} = g(W_x x^{<t+1>} + W_a A^{<t>} + b) \tag{2.6}$$

Where W_x is the linear weight matrix for the input at each time step, W_a is the linear weight matrix for the activations of the previous step, b is the bias term. The architecture of RNN is shown in Figure 2.3

In practice, the RNN model is rarely used because RNN is usually ineffective to encode long time dependency among all the time steps in each training sample [23]. Two variants of the RNN model: the long short time memory (LSTM) [24] and gated recurrent network (GRU) [25], have been widely used as the alternatives to the plain RNN model. The general structures of GRU and LSTM are similar as RNN except for that they use more complex mapping functions to compute the activations of each time step.

[1]In RNN, the activations' here are actually the 'hidden state' values. The term is still kept to make the description of RNN consistent with that of MLP and CNN.

In GRU, the activations computed by equation (2.6) are treated as the 'candidate activations' $\tilde{A}^{<t+1>}$, and the actual activations are the weighted combination of the activations of the previous step and the candidate activations. The combination weights of the two types of activations are decided by the 'update' gate Γ_u :

$$\Gamma_u = \sigma(W_{ux}x^{<t+1>} + W_{ua}A^{<t>} + b_u) \tag{2.7}$$

Where σ is the sigmoid function, W_{ux} and W_{ua} are the linear weight matrices for the input and the previous activations to compute the update weight, respectively. The actual activations of time step $t^{<t+1>}$ are computed as:

$$A^{<t+1>} = \Gamma_u A^{<t>} + (1 - \Gamma_u)\tilde{A}^{<t+1>} \tag{2.8}$$

In LSTM, each unit contains two types of values: activation values and 'memory cell' values. The candidate values for the memory cell at step $t + 1$ are computed as :

$$\tilde{c}^{<t+1>} = g(W_{ma}A^{<t>} + W_{mx}x^{<t+1>} + b_m) \tag{2.9}$$

Where W_{ma} and W_{mx} are the linear weight matrices for the activations of the previous time step and the input, respectively, b_m is the bias term. The actual memory cell values for the time step is a weighted combination of the memory cell values $c^{<t>}$ of time step t and the candidate memory cell values $\tilde{c}^{<t+1>}$. The weights for $c^{<t>}$ and $\tilde{c}^{<t+1>}$ are governed by a 'forget gate' Γ_f and an 'update gate' Γ_u , whose values are computed by two independent equations similar as (2.7), respectively. The actual memory cell values are computed as follows:

$$c^{<t+1>} = \Gamma_f c^{<t>} + \Gamma_u \tilde{c}^{<t+1>} \tag{2.10}$$

LSTM uses an 'output gate' value Γ_o, that is also computed in a similar way as the forget and update gate values, to compute the activations of time step $t + 1$ based on the memory cell values of that time step:

$$A^{<t+1>} = \Gamma_o * g(c^{<t+1>}) \tag{2.11}$$

2.4 Fitting the models

The training process of a learning model is essentially a numerical optimisation problem with a goal of minimising the costs of the learning model. Various convex optimisation techniques [26], such as gradient decent, the Newton method, conjugated gradient, etc., can be used for the training processes of the traditional learning models whose cost functions are convex.

However, in the context of deep learning, the training process is more difficult because the deep structures result in very high risk of gradient explosion or gradient vanishing [27]. To stabilise the training process, the following special treatments are usually necessary. Firstly, the weight matrices have to initialised with small non-zero fractions to prevent their gradients from quickly exploding or vanishing [28]. Secondly, in each iteration of the training process, it is beneficial to normalise the activations of each hidden layer [29]. Thirdly, special optimisation techniques, such as gradient decent with momentum **??**, RMSProp [30], Adam [31], etc., are usually much more effective than the aforementioned optimisation techniques. Details of the initialisation process, activation normalisation, and the special optimisation algorithms are introduced as follows.

2.4.1 Weight initialisation

In practice, the weight matrices cannot be initialised with 0 as it can cause the 'symmetric gradient' problem: the derivatives of the loss function with respect to all the weights are all the same, thus, all the weight matrices will have the same values in the subsequent iterations.

The weight matrices have to be initialised with small values that are close but not equal to 0. There are existing the following frequently used and effective methods for the weight initialisation: Xavier initialisation [32] and He initialisation [33].

Assuming the number of nodes in hidden layer l is $n^{[l]}$, then in the Xavier initialisation, the weight matrix $w^{[l]}$ will be initialised with values that are sampled from a normal distribution with a mean of 0 and a variance of $\frac{1}{n^{[l]}}$:

$$w^{[l]} \sim N(0, \sqrt{\frac{1}{n^{[l]}}}) \tag{2.12}$$

He initialisation is similar as Xavier initialisation, but is specifically for the tanh activation function. In He initialisation, the weight matrix $w^{[l]}$ for the tanh activation functions is initialised with values that are sampled from a normal distribution with the mean of 0 and the variance of $\frac{2}{n^{[l]}}$:

$$w^{[l]} \sim N(0, \sqrt{\frac{2}{n^{[l]}}}) \tag{2.13}$$

2.4.2 Activation normalisation

In the training process, the activation values of the hidden layers may vary significantly as the input data changes. Drastic changes in the hidden activation values may cause the training process to be very unstable. In practice, batch normalisation is usually imposed on activation values to alleviate the problem.

Given a dataset containing M training examples, assuming the state values of a hidden layer is Z. In the normalisation process, Z is first normalised as follows:

$$Z_{norm} = \frac{Z - \mu}{\sigma} \tag{2.14}$$

where μ is the mean of the state values of all the M examples, σ is the standard deviation of the state values for all the M examples. Then Z_{norm} is linearly transformed as follows:

$$\tilde{Z} = \gamma * Z_{norm} + \beta \tag{2.15}$$

where γ and β are unknown linear coefficients, that will be learned in the training process. $\tilde{Z}$ is used to replace the original state values and compute the activation values of the hidden layer.

2.4.3 Optimisation algorithms

In the generic gradient decent algorithm, the gradients of the cost function with respect to the trainable variables may oscillate wildly around the shortest decent path. In other words, the generic gradient decent algorithm always detours around the optimum, and have difficulties to identify the short decent path. A pictorial demonstration of the behaviour of gradient decent is shown in Figure 2.4.

The gradient oscillation can significantly slow down the training process, even cause the algorithm to miss the optimum. To accelerate and stabilise the training process, the following optimisation algorithms are usually used: gradient decent with momentum, RMSProp, Adam.

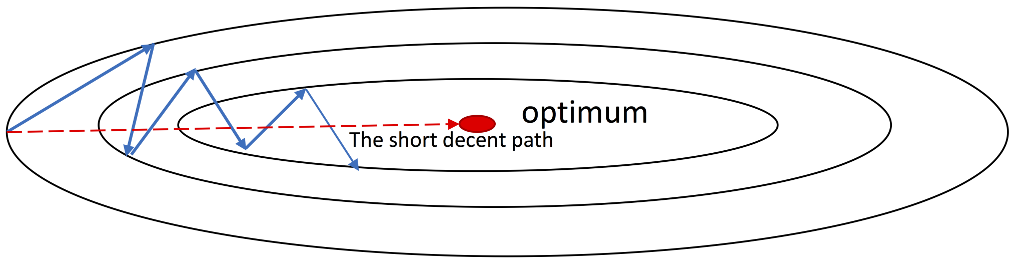


Figure 2.4: The oscillation behaviour of gradient decent

'Momentum' in the gradient decent with momentum algorithm is essentially the smoothed gradients. Smoothing the gradients is the core idea to alleviate the gradient oscillation problem. Assuming the gradient of the weight matrix in the first iteration of a gradient decent process is $dw^{<0>}$, the gradients of the second and the third iteration are $dw^{<1>}$,$dw^{<2>}$, respectively, then the smoothed versions of $dw^{<0>}$, $dw^{<1>}$ and $dw^{<2>}$ are computed as follows:

$$\begin{aligned} V_{dw^{<1>}} &= (1-\beta) * dw^{<0>} \\ V_{dw^{<2>}} &= \beta * V_{dw^{<1>}} + (1-\beta) * dw^{<1>} \\ V_{dw^{<3>}} &= \beta * V_{dw^{<2>}} + (1-\beta) * dw^{<2>} \end{aligned} \tag{2.16}$$

where β is the a hyper-parameter that controls the smoothing level. Similarly, given the smoothed gradient of the ith iteration $V_{dw^{<i>}}$, and the original gradient of iteration $i+1$ $dw^{<i+1>}$, the smoothed gradient of iteration $i+1$ is computed as:

$$V_{dw^{<i+1>}} = \beta * V_{dw^{<i>}} + (1-\beta) * dw^{<i+1>} \tag{2.17}$$

In gradient decent with momentum, at each iteration i, assuming the learning rate is α, then the weight matrix w is updated as follows:

$$w = w - \alpha * V_{dw^{<i>}} \tag{2.18}$$

In the RMSDrop algorithm, let S denote the momentum, and it can be computed as follows:

$$S_{dw^{<i+1>}} = \beta * S_{dw^{<i>}} + (1-\beta) * (dw^{<i+1>})^2 \tag{2.19}$$

The weight matrix w in an iteration i is updated as follows:

$$w = w - \alpha * \frac{dw^{<i>}}{\sqrt{S_{dw^{<i>}}}} \tag{2.20}$$

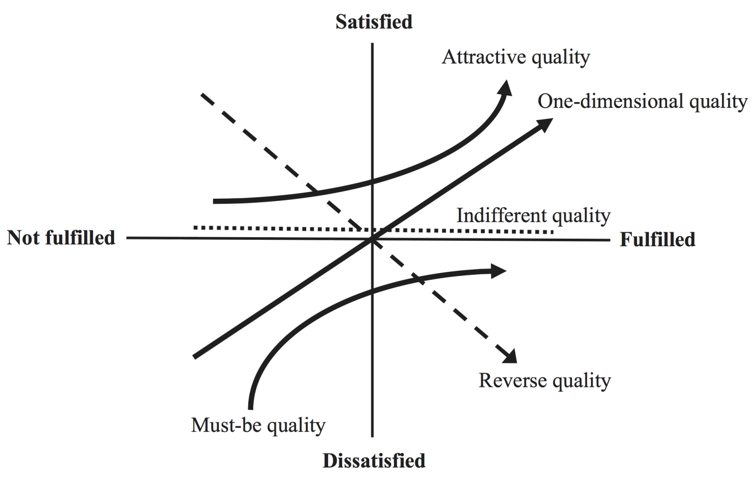


Figure 2.5: The relations between attribute performance and customer satisfaction

Adam algorithm combines the momentum in both algorithms. The weight matrix w in an iteration i can be updated as follows:

$$w = w - \alpha * \frac{V_{dw^{<i>}}}{\sqrt{S_{dw^{<i>}}}} \tag{2.21}$$

2.5 Kano model theory

The relations between the performance of an attributes and the overall satisfaction level can reflect the nature of the attribute. Based on the relations, Kano theory [13] divides all attributes of a product or service into the following 5 categories: must-be, one-dimensional, attractive, indifference, and reverse attributes. The definitions of the 5 attribute categories are as follows:

- Must be: these are the requirements that the customers expect and are taken for granted. When their performance is sufficient, customers are not necessarily satisfied; but when their performance is insufficient, customers are very dissatisfied. For example, for a cellphone, signal quality is a must-be attribute.

- One-dimensional: they result in satisfaction when their performance is sufficient, and dissatisfaction when their performance is insufficient. For example, the screen resolution of a cell phone is an one-dimensional attribute: customers are satisfied if the resolution is high and dissatisfied if the resolution is poor.

- Attractive: they result in satisfaction if their performance is sufficient, but not necessarily lead to dissatisfaction if their performance is insufficient. For example, the voice assistant of a smart phone is an attractive attribute: users would be satisfied if the system works well but not necessarily dissatisfied if the system performs poorly.

- Indifference: the performance of those attributes has no or very limited impacts on customer satisfaction. For example, some rarely used features of a cellphone can be counted as indifference' : users just ignore them and their performance poses no impact on overall customer satisfaction.

- Reverse quality: they result in dissatisfaction when those attributes are overly implemented. For example, for a phone with a target group of the people who are not very tech-savvy, a state-of-the-art touch screen may result in dissatisfaction.

The asymmetric relations between attribute-level performance and customer satisfaction is pictorially demonstrated in Figure 2.5 [13]. Categorising the attributes of a product or service into those categories (especially into the first 3 categories) helps people understand the driving forces of customer satisfaction, and optimally allocate limited resources in the product or service development process to maximise customer satisfaction.

The Kano questionnaire is commonly used to categorise attributes into those categories. The questionnaire consists of functional and dysfunctional questions. Functional questions aim to get respondents' responses when the performance of an attribute is sufficient; the dysfunctional questions are to collect respondents' responses when the performance is insufficient. Based on the responses, one can make the categorisation decision based on the Kano matrix shown in Table 2.1.

In the table, 'M' stands for must-be, 'O' stands for one-dimensional, 'A' stands for attractive, 'I' stands for indifferent, 'R' stands for reverse. If an respondent's answer regarding an attribute falls in a cell, then the corresponding category shown in the cell should be the category the attribute belongs to.

Table 2.1: The Kano matrix

		Dysfunctional question How do you feel if the performance of the attribute is insufficient?				
		satisfied	it should be that way	I am neutral	I can live with it	I dislike it that way
Functional question How do you feel if the performance of the attribute is sufficient ?	satisfied		A	A	A	O
	It should be that way	R	I	I	I	M
	I am neutral	R	I	I	I	M
	I am live with it	R	I	I	I	M
	Dissatisfied	R	R	R	R	

Chapter 3

Representing sentences by a weighted average of word embeddings

This chapter presents a novel weighted average approach, named ‘Abstract Keywords’, to combine word embedding vectors to represent sentences in the training process of the opinion classifiers.

3.1 Introduction

In short text classification, the traditional text encoding models, such as one-hot encoding and TF-IDF [6], would inevitably result in very sparse, high-dimensional representations, that are generally hard for existing learning models to handle. Furthermore, traditional text representation models do not encode semantical information and rely only on word overlaps to measure document similarities, therefore, cannot capture the similarity between a pair of documents that are semantically related but have no words in common.

Word embedding models [9, 20, 34, 35], that learn low-dimensional and dense feature vectors for each individual word to represent their semantic meaning, have been proven more effective in a wide range of short text classification tasks. However, word embedding models have an obvious downside: to represent a document in the classification task, one must combine embedding vectors of all words in the documment into a single vector by predefined composition operators. Currently, the easiest and most widely used composition operator is element-wise averaging [36, 37], though it is obviously problematic since it assign words equal weights that can easily dilute salient signals and augment noisy signals.

One can also use convolutional layers in CNN [38, 39], or the recurrent units in RNN [40, 41], as the composition operators. Compared with the averaging approach, on one hand, CNN and RNN are much more computationally expensive; on the other hand, they encode much richer structural information of the training documents. Theoretically, the CNN and RNN based approaches would have better performance than the averaging composition operator. However, surprisingly, Iyyer et al. [37] found that the performance of the averaging composition operator can rival that of the RNN or CNN based composition operators in short text classification. One possible reason, as the authors stated, is that the diluted salient signals in the averaging operator will be increasingly amplified as the learning models getting deeper.

In short text classification, the averaging operator is still a strong baseline. To surpass it, many researchers have been focusing on devising more complex structures based on the frameworks of CNN or RNN. Though exciting improvements have been reported in recently published papers [42–47], the more complex model structures usually lead to massive increases in computation costs and require more labelled data for the training process.

In this research, we aim to build a weighted averaging operator, that assigns different words different weights according to their semantical meaning. We assume in each particular classification task, there exist some abstract keywords whose embeddings are extremely salient. For the words in each document, the more similar they are to the keywords, the higher the weights they will be assigned in the composition process. Since the abstract words are unknown beforehand, they are integrated into the structure of the learning model under study and learned along with the existing model parameters in the training process. Experiment results show that, this approach is much more computationally cheaper than the CNN and RNN based approaches, and has better performance than the averaging approach. The remaining of this chapter is organised as follows. Section 2 presents a brief introduction to related work. Section 3 details the proposed approach.

Section 4 shows the evaluation results and Section 5 presents the conclusion and future research directions.

3.2 Related work

In this section we give a brief introduction to the compositionality of word embeddings based on deep learning models. Also, since this research is related to text embedding models, a number of representative works in this filed are introduced though they are not the focus of this research .

3.2.1 Composition of word embeddings based on deep learning models

Iyyer et al. [37] proposed to use the average of word embeddings of a document as the compositional representation of the document. The authors feed the averaged representation to a deep multilayer perceptron model, called DAN, and find that its performance rivals other models using more complex structures, such as CNN, LSTM, etc.

Kim [38] proposed to apply the convolution operator to model the semantic composition of word embeddings. The author first stacks all the word embedding vectors in each document into a matrix, then applies convolution filters of varying sizes on the embedding matrix to get the convolution responses of the matrix. The most salient features of the responses, that are selected by applying the pair-wise max pooling on the convolution results, are concatenated as the representation of the document. A pictorial demonstrator of the process is shown in Figure 3.1 [38].

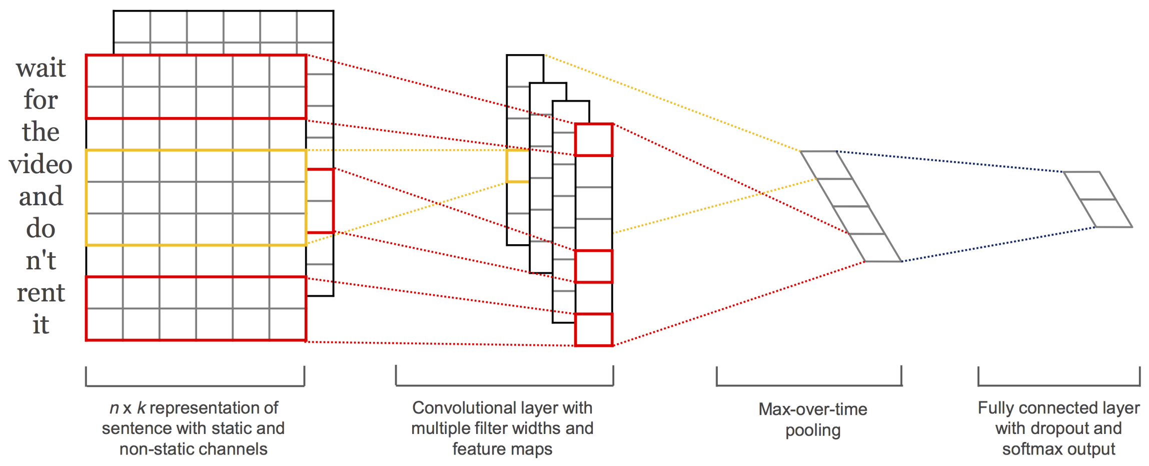

wait
for
the
video
and
do
n't
rent
it
n x k representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Figure 3.1: The structure of the CNN model for combining word embeddings

Tai et al. [40] proposed to use the many-to-one structured LSTM model to combine word embeddings. In this structure, each word position of a sentence is treated as a time step, and the embedding of the word at each position is fed into the RNN unit of the corresponding time step. The hidden state h_t at time step t is a function of the embedding vector for the time step and the previous hidden state h_{t-1}. The hidden state at the last time step is used as the semantic representation of the document. A pictorial demonstration of the composition process is shown in Figure 3.2.

Wang et al. [48] proposed a hybrid RNN-LSTM model for the composition process. In the model, the authors first feed the input data to a CNN model, then pass the output of the CNN model to a LSTM model. The output of the LSTM sequence is used as the compositional representations. The authors claimed that this approach combines the expressiveness of the both models, therefore, is more effective than the approaches using RNN or CNN alone.

3.2.2 Text embedding models

Besides the word embedding models, there exist a number of sentence embedding models. Sentence-embedding models also use word embeddings as the building blocks, however, they take into account the semantical composition of individual words during their training process. In other words, the word embeddings generated by sentence embedding models are already optimised for the composition process based on their predefined composition operators, before being fed into a classifier.

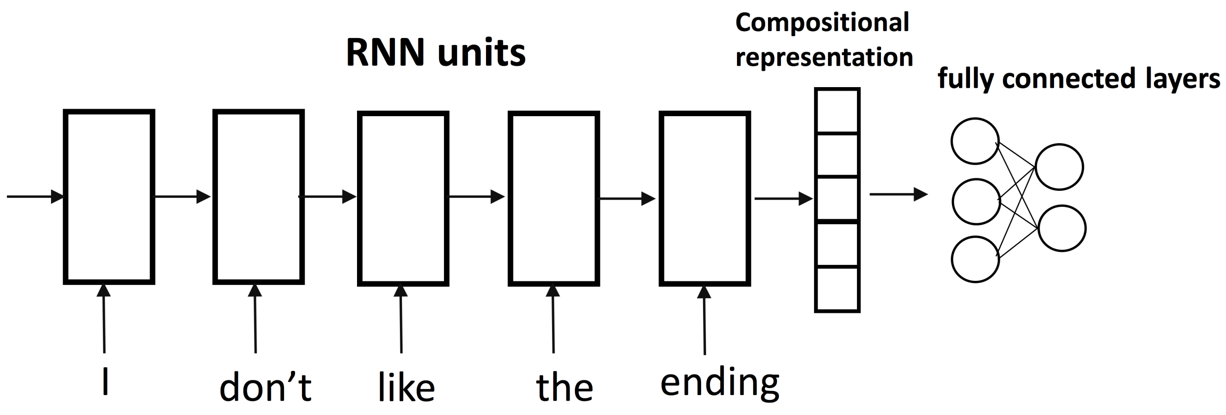


Figure 3.2: The structure of the RNN model for combining word embeddings

One of the most widely used sentence embedding models is the paragraph to vector model (Para2vec) [49], that is an extension of the Word2vec model. Similar as Word2vec, Para2vec is also built upon the language model that uses a piece of textual context to predict a word may appear inside or next to the context. The difference between them lies in that, Word2vec uses only words in a sentence as the context, whereas Para2vec uses both words and the indexing identities of sentences as the contextual information. By fitting the shallow neural network based language model, the embedding representation of each sentence and each word can be learned at the same time.

Kenter et al. proposed the Siamese CBOW model [50] to derive sentence embeddings. This model is build upon a language model that predicts whether a pair of sentences should be adjacent to one another in the training set. In the model, each sentence is represented by the average of the word embeddings in the sentence, and the possibility of a pair of sentences being adjacent is decided by their cosine similarity. By fitting the language model, word embeddings will be optimised for being averaged to represent sentences.

Kiros et al. proposed the Skip-thought model [51] that uses encoder-decoder structure to encode the composition process. The encoder of the model uses a LSTM model to aggregate the embeddings of all words in a sentence, and the decoder uses another LSTM model to predict the previous and next sentences of the input sentence conditioning on the output of the encoder. Compared with the Siamese CBOW model, this model encodes word order of documents therefore has better performance in applications that are sensitive to the sequential information.

3.3 The Abstract Keywords approach

Assuming there exists an extremely salient abstract keyword w_a for a particular classification task, let v_a denote the embedding vector of the abstract keyword. As mentioned previously, we assume the more similar the words in a document are to the abstract keyword w_a, the higher the weights they will be assigned in the composition process. In this paper the similarity between a pair of word embeddings v_i and v_j is measured by their dot product.

Assuming there are n_m words in a document, the weights of the words in the document for the composition process are computed by the softmax function based on their semantic similarities to the abstract keyword:

$$w_i = \frac{e^{v_i^\top v_a}}{\sum_{i=1}^{n_m} e^{v_i^\top v_a}} \tag{3.1}$$

The compositional embedding vector of the document is computed as follows:

$$\bar{v}_m = \sum_{i=1}^{n_m} w_i * v_i \tag{3.2}$$

The compositional embedding vectors are fed into the learning models. Let $h(x)$ denote the prediction function of a deep MLP model, then the negative log-likelihood loss is computed as follows:

$$L(\theta, v_a) = -\sum_{c=1}^{C} 1\{y_m = c\} \log(h(\bar{v}_m; \theta, v_a)) \tag{3.3}$$

θ is the parameters of the MLP model. As the equation suggests, the embedding of the abstract keyword is integrated into the loss function and can be derived along with the model parameters θ in the training process. In the test phase, the word embedding vectors of an unseen sentence are combined by the same weighted averaging shown in equation (3.2).

It is also possible to introduce multiple abstract keywords to make the model more non-linear. In this case, we compute a compositional embedding with respect to each

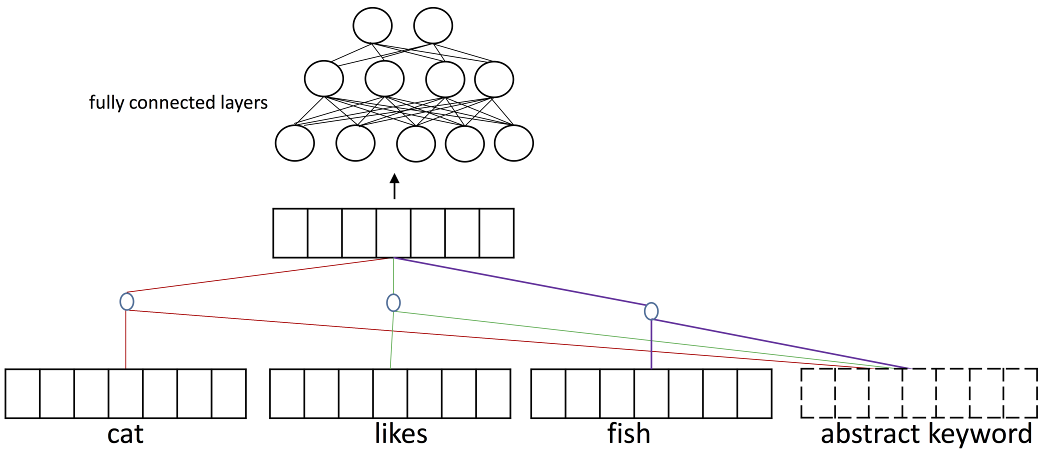


Figure 3.3: A toy example of the word embedding aggregation in Abstract Keyword

abstract keyword, and combine the multiple compositional vectors by concatenating or averaging as the input of the MLP. However, for computation efficiency, we limit the number of abstract keywords within 2 in this research.

A pictorial demonstration of the composition process in Abstract Keywords is shown in Figure 3.3.

3.4 Evaluation

In this section we evaluate the performance of the proposed weighted average approach. DAN [37], CNN [38], RNN (including GRU and LSTM) [40] and the CNN-RNN hybrid model [48] are used as the baselines. The following datasets that are frequently used in existing papers are used in the experiment.

- MR: Movie reviews with one sentence per review. Classification involves detecting positive/negative reviews [52].
- SST-1: Stanford Sentiment Treebankan extension of MR but with train/dev/test splits provided and fine-grained labels (very positive, positive, neutral, negative, very negative) [53].
- SST-2: Same as SST-1 but with neutral reviews removed and binary labels.
- Subjectivity dataset where the task is to classify a sentence as being subjective or objective [54].

- TREC: TREC question datasettask involves classifying a question into 6 question types (whether the question is about person, location, numeric information, etc.) [55].
- MPQA: Opinion polarity detection subtask of the MPQA dataset [56].

Statistics of the datasets are shown in Table 3.1. The Google's word2vec model [1] pre-trained on the Google News corpus is used across for all the models. The dimensionality of each word embedding vector is 300. For all the baseline models, we use the same architectures reported in the original papers introducing those models. For the proposed approach, we choose the architecture shown in Table 3.2 by grid search on the SST-1 dataset and use it across on all the datasets.

Table 3.1: Statistics of the datasets used Chapter 3

Data	Number of target class	Average sentence length	Data Size	Vocabulary size
SST-1	5	18	14065	17836
SST-2	2	19	11434	16185
Subj	2	23	10000	21323
TREC	6	10	6452	9592
MR	2	20	10662	18765
MPQA	2	3	10606	6246

Also, as mentioned in Section 3.3, it is possible to use multiple abstract words. To demonstrate the impact of the number of abstract words, we train two models, one with only one abstract word, and the other with two. The 10-fold cross validation is used on each task and the classification performance of all the models is evaluated in terms of average accuracy. The comparison results are shown in Table 3.3.

We first compare the performance of the average baseline approach against that of other baseline models. As the results indicate, the performance of the DAN model is

[1]https://code.google.com/archive/p/word2vec/

Table 3.2: The architecture of Abstract Keywords

layers	size	regularisation strength	drop-out rate
input	300	-	-
abstract word	300	-	-
1st layer	500	0.001	0.15
2nd layer	150	0.001	0.15
3rd layer	60	0.001	0.15
output layer	task specific	-	-

not always lower than that of other baseline models, though it uses the simplest model structure. On Subj and SST-1, the DAN model even outperforms other baseline models.

We then compare the proposed approach against the average baseline approach. The proposed approach outperforms the averaging approach on all the datasets except for SUBJ. The proposed approach enjoys an average advantage margin of 1% over the averag approach when using 1 keyword, and an average advantage of 2.1% when using two keywords.

Lastly, we compare the proposed approach against other baseline models. When using 1 keyword, the performance of the proposed approach is on par with that of the LSTM, GRU and CNN models; when using 2 keywords, the average accuracy of the proposed approach is very close to that of the CNN-LSTM hybrid model.

Furthermore, to demonstrate the computation efficiency of the proposed approach, the average training time (from the start of the training to the point when the performance on the validation data starts to decrease) for each model is also recorded. The computation environment is as follows: MacBook Pro, 2.7 GHz Intel Core i5, 8 GB 1867 MHz DDR3. The time comparison is shown in Table 3.4. As the results indicate, on each dataset, the training process of the proposed approach with one abstract keyword takes 35% less time by average than the most computationally efficient model among all the baselines except for DAN, and around 20% less when using two.

We also check the semantic meaning of the abstract words. We show the top 10

Table 3.3: Performance of Abstract Keyword and the baseline models

Model	SST-1	SST-2	Subj	TREC	MR	MPQA
DAN	0.466	0.821	0.924	0.896	0.782	0.895
RNN(LSTM)	0.463	0.839	0.902	0.910	0.796	0.906
RNN (GRU)	0.459	0.835	0.90	0.912	0.820	0.880
CNN	0.450	0.865	0.906	0.914	0.810	0.873
CNN-LSTM hybrid	0.464	0.856	0.910	0.919	0.812	0.921
Ours-1 abstract word	0.475	0.835	0.912	0.900	0.815	0.90
Ours-2 abstract word	0.487	0.850	0.919	0.910	0.827	0.912

Table 3.4: Time (seconds) for the training process of Abstract Keywords and the baseline models

Model	SST-1	SST-2	Subj	TREC	MR	MPQA
DAN	40	45	47	33	68	32
RNN(LSTM)	330	305	310	162	400	266
RNN (GRU)	311	285	309	169	392	271
CNN	235	270	272	140	334	280
CNN-LSTM hybrid	190	210	255	120	266	154
Ours-1 abstract word	107	130	146	100	185	125
Ours-2 abstract word	136	160	180	132	210	150

semantically nearest words, and the top 10 most distant words of the learned abstract word (from the model using only one abstract keyword) in Table 3.5. We observe that the nearest words are the ones that are semantically related to the target class labels, while the majority of the most distant words are stop words that have no or little semantic meaning. This further justifies our approach of weighting words according to their similarities to the abstract words.

Table 3.5: The most semantically nearest words of the abstract word

TREC	SST-1	SUBJ	MPQA
prize guaranteed	pitiful	enjoyably	ineffective
half price	dreadful	talky	excused
ipod	lousy	overlong	barred
pub	unappealing	formulaic	upset
chance win	dismal	likably	harm
bonus caller	sloppy	unsatisfying	detrimental
auction	defecates	joyless	nothing
services	incapable	unengaging	incapable
wine	miserably	mundanity	doubtful
replying	undressed	tiresomely	runin

Table 3.6: The most semantically distant words of the abstract word

TREC	SST-1	SUBJ	MPQA
situation	the	now	bring
vomit	this	is	our
changed	here	the	is
congratulations	at	in	has
wonder	our	may	get
correct	its	will	at
kinda	local	below	come
happened	we	within	follow
joke	now	above	put
txting	today	first	world

3.5 Conclusion

In this chapter we propose a weighted average approach to aggregate word vectors to represent short text. The proposed approach assumes there exist some very important abstract keywords, and the weight of each word in each document is determined by the semantic similarities between the word and the abstract keywords. The abstract keywords are integrated into the cost function of the learning model under study, and learned along with other model parameters during the training process. Experiment results show that the performance of the proposed approach is better than that of the basic averaging approach, and on par with that of the aggregation operators based on complex neural networks, such as CNN and RNN. Furthermore, compared with the CNN, RNN and their hybrid model, the training process of the proposed approach is computationally effective. In this research we limit the number of abstract keywords to 2. In the future, we plan to extend the proposed approach by using more abstract keywords to have broader understanding of the proposed composition process.

Chapter 4

Averaged-logits: a Weakly-supervised Approach to Use Labelled Documents to Train Sentence-level Sentiment Classifiers

4.1 Introduction

Supervised machine learning methods have been widely applied in sentiment analysis [57–63]. To train such classifiers, one needs to provide training data labelled at the granularity the same as that of the sentiment classification task at hand. For example, it requires labelled documents to train document-level sentiment classifiers, and labelled sentences for sentence-level sentiment classifiers. To label the data, one needs to manually read the textual content of each training example, and decides which sentiment class each sample is associated with. Therefore, the labelling process usually takes laborious manual efforts. Also, the finer the granularity, the more difficulties to make the labelling decisions and more examples to be labelled, therefore, more manual efforts are needed to prepare the data. In practice, datasets of labelled sentences are much more expensive than datasets of labelled documents to obtain. The difficulty of the sentence labelling process can be reflected by the fact that only few datasets containing sentence-level sentiment labels are publicly available on the internet. The unavailability of labelled sentence datasets limits the feasibility of training effective sentence-level sentiment classifiers, especially in today's era when many methods are built upon deep learning models that usually require very large datasets for the training processes.

One possible way to ease the problem is using labelled documents that are less expensive, sometimes even free (such as on-line reviews with ratings) to obtain, to train sentence-level sentiment classifiers. The easiest way to do that is treating each document as a single long sentence, the label of the document as the label for the long sentence, and apply the generic methods for sentence-level sentiment classification to train the classifiers.

However, there exists a problem in training sentence-level sentiment classifiers with labelled documents: the sentiments of individual sentences may be very different from the sentiments of their containing documents. For example, there may exist very negative sentences in a generally positive document, or positive sentences in a generally negative document. By using document-level sentiments as the supervision signal, the true sentiment orientations of individual sentences can be easily misrepresented, that would inevitably result in harmful impacts to the classifiers. Furthermore, sequential deep learning models, such as GRU and LSTM, cannot handle long word sequences. To train classifiers with these models on long documents, one has to either shorten the documents by throwing away words or use the truncated back-propagation technique in the training process. The former way leads to loss of information, the latter way may significantly increase the training time.

This paper proposes a novel end-to-end approach, called 'Averaged-logits', that uses labelled documents to train sentence-level sentiment classifiers. Unlike the previously mentioned methods, the proposed approach keeps the sentiments of individual sentences, and allows for using the sequential models without truncating the documents or the need of the truncated back-propagation method. In the proposed approach, individual sentences are fed as the inputs, and the averages of the resulting logit vectors of sentences from the same reviews are used to compute the sentiment distributions of the containing documents. The cross entropies between the resulting sentiment distributions and the ground-truth document-level labels are used as the cost function. In the experiment, we collected two review datasets: one contains 50,000 hotel reviews crawled from TripAdvisor, the other 50,000 electronic product reviews crawled from Amazon. We used the review data to train classifiers based on the proposed approach and the results indicate that, the Averaged-logits classifiers outperform the existing approach that treats documents as long sentences by margins of $3\% - 8\%$. At the meantime, we observed that, by using a large enough training dataset, the performance of the Averaged-logits classifiers can be on par with that of generic classifiers trained with labelled sentences.

The rest of this paper is organised as follows: Section 4.2 presents a brief introduction to the existing work; Section 4.3 introduces the proposed approach; Section 4.4 presents the evaluation results and Section 4.5 provides the conclusion and future research direction.

4.2 Related work

This section first presents a brief introduction to existing methods that use labelled sentences to train sentence-level sentiment classifiers, then present a few rare methods that exploit document-level sentiment labels to train sentence-level sentiment classifiers.

4.2.1 Sentence-level sentiment classification based on labelled sentences

A critical phase in generic text classification problem is extracting effective input features to represent the raw text. This is especially true for sentence-level sentiment classification, as sentences are usually very short and noisy. Based on how the features are extracted, the existing methods are divided into two categories: 1) lexicon-based methods whose feature extraction process involves the use of external opinion lexicons ; 2) machine-learning based methods that exploit the statistical patterns or contextual structures of sentences to define the features.

Lexicon based methods

Lexicon based models require lexicons consisting of opinion words or phrases. Those methods assume that opinions words and phrases are the dominating indicator for sentiment classification, therefore, the input features of those methods are usually derived based on the presence or absence of the opinion words in a text fragment. A big array of models, ranging from rule based algorithms [68–70], to unsupervised [71–73] and supervised machine learning models [74–78], have been proposed to build sentiment classifiers with the help of sentiment lexicons.

A problem with the lexicon based methods is that the opinion lexicons are manually built, therefore, the scope and size of a lexicon is usually very limited. There exists a high possibility that the words of an opinionated sentence has no overlap with the opinion lexicon. Researchers have proposed to automatically expand the lexicons by iteratively searching WordNet or other dictionaries for synonyms and antonyms of the opinion words as the addition to the lexicons [4, 79–81]. The shortcoming of this approach is that the added words from the external dictionaries can be unrelated to the training corpus, therefore, cannot reflect the domain knowledge and corpus-specific statistics. To address the problem, Liu et al. [70, 82] proposed to search the training corpus, instead of external dictionaries, for words that are semantically or syntactically related to the opinion words. However, it is usually not easy to determine the sentiment orientation of the newly added words by the semantic or syntactical connection, therefore, external linguistic knowledge and intense labor are still needed to help make the decision.

Machine learning based methods

Compared with lexicon based methods, machine learning based methods do not involve opinion lexicons in the feature extraction process. Generic text features, such part-of-speech tags, tf-idf weights, term frequencies, etc., are very popular in those methods [57–63]. Inevitably, those features are much more noisy than the features generated from opinion lexicons. To compensate for the downside, those methods usually use complex model structures or large datasets. This is especially true in recent years as many newly developed methods are built upon deep learning models and big data. Socher et al. [53] proposed the recursive neural network, in which the authors first use the tree structure grammars to parse an input sentence into sub-phrases, then combine the embedding vectors of words in each sub-phrase through a set of pre-defined combination functions as the compositional representation for the sub-phrase. The sub-phrase representations are passed to a MLP model to predict the sentiment of each phrase. By fitting the MLP, not only the sentiment classifier, but also the phrase representations, can be learned. Tai et al. [40] proposed the Tree-LSTM, in which the authors apply the LSTM model on a sentence, and combine the hidden and memory cell states of words in each sub-phrase as the representation of the sub-phrase. Kim [38] stacks the embedding representations of words in each sentence and applies the convolutional neural network on the stacked feature matrix to train a sentiment classifier. Wang et al. [48] combine the expressiveness of LSTM and CNN by proposing a hybrid model, that first applies 1-dimensional convolutional filters on the embedding feature matrixes of the input sentences, then passes the resulting feature maps to RNN units to get deep sentence representations for sentiment classification.

4.2.2 Exploiting document-level sentiment labels for sentence-level sentiment classification

Besides the methods mentioned in the Introduction section, there also exist the following methods that exploit document-level sentiment labels to train sentence-level sentiment classifiers. In those models, document-level sentiments are usually used as a form of weak supervision, that has to work with other knowledge, such as opinion lexicons, human linguistic expertise, sentence-level sentiment labels, etc., to train sentence-level classifiers. Qu et al. [64] proposed a multi-expert model that makes use of local syntactic patterns of sentences, and opinion lexicons to build a set of rule-based base predictors, and predict the sentiment labels based on the votes of the base predictors. Yang et al. [65] proposed a CRF model that uses sentence-level sentiment labels as the main supervision signal, and use overall ratings as a form of posterior regularisation to keep sentence-level sentiments and document-level sentiments consistent. Tackstrom et al. [66] proposed a Hidden CRF model, that treats the sentence-level sentiments of a review as latent variables, and the overall ratings are observable variables conditioning on the latent variables. Opinion lexicons are used in the method to define the feature functions for the CRF model. Wu et al. [67] proposed the SSWS model that uses two levels of features: document-level sentiments and word-level sentiments, along with a predefined set of linguistic rules, to train a linear sentence-level sentiment classifier.

As previously mentioned, those methods also require intense human labor and linguistic expertise to build the opinion lexicons and linguistic rules. We differ our research

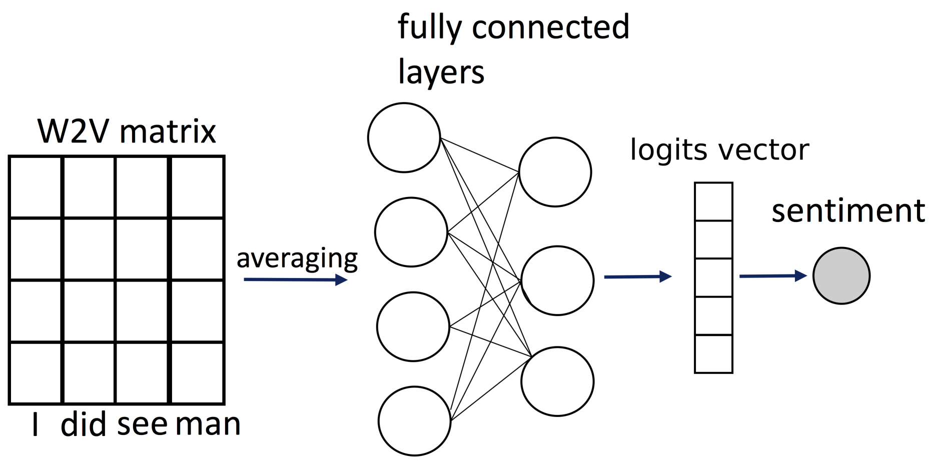


Figure 4.1: The application of MLP in sentiment classification

from those existing work by that we focus on generalised, end-to-end approach to train the sentence-level sentiment classifiers, to avoid the laborious efforts to build the lexicons or the external linguistic expertise .

4.3 The Averaged-logits model

In this paper, 4 types of neural network models: MLP, LSTM, GRU, and CNN, can be used in the proposed approach to train sentence-level sentiment classifiers under the supervision of document-level labels. This section will first introduce the structures of the 4 models in sentiment classification, then present the framework of the proposed approach. Word-to-vector (Word2vec) embedding vectors are used across on all the 4 models as the word representations.

4.3.1 Multiple layer perceptron

To use MLP to train the sentence-level sentiment classifier, one first has to combine the multiple word vectors associated with a sentence into a single vector as the sentence representation. The most commonly used method to combine word vectors is element-wise averaging.

Assuming the average of the word vectors of a sentence is $\bar{x}$, the activation function

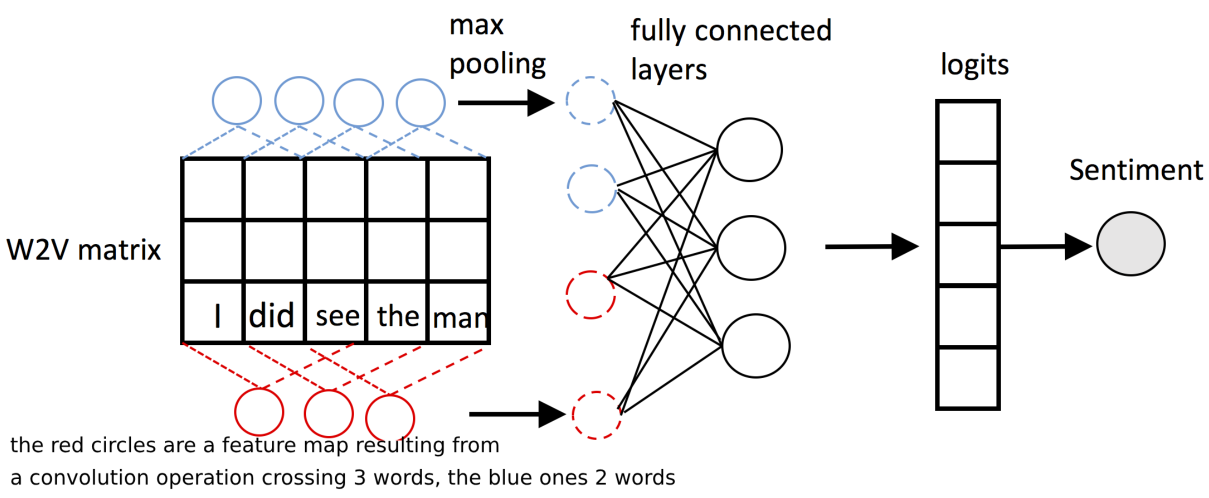

max pooling
fully connected layers
logits
Sentiment
W2V matrix
I
did
see
the
man
the red circles are a feature map resulting from a convolution operation crossing 3 words, the blue ones 2 words

Figure 4.2: The application of CNN in sentiment classification.

of a MLP model is denoted as g, then the activation of the first hidden layer is :

$$A^{[1]} = g(W^{[1]}\bar{x} + b^{[1]}) \tag{4.1}$$

where $W^{[1]}$, $b^{[1]}$ are the linear weights and bias for the first layer, respectively. Similarly, the activation in the nth hidden layer is computed as

$$A^{[n]} = g(W^{[n]}A^{[n-1]} + b^{[n]}) \tag{4.2}$$

In the output layer $l^{[N]}$, the activation of layer $l^{[N-1]}$ is linearly mapped to a vector called 'logits' as follows:

$$Z^{[N]} = W^{[N]}A^{[N-1]} + b^{[N]} \tag{4.3}$$

$Z^{[N]}$ is used to used to compute the probability distribution of the input over all the target classes based on the Softmax function:

$$p(y_i|\bar{x}) = \frac{\exp Z_i^{[N]}}{\sum_{j=1}^{C} \exp Z_j^{[N]}} \tag{4.4}$$

The structure of MLP in sentiment classification is shown in Figure 4.1.

4.3.2 Convolutional neural network

In a CNN model, the word vectors associated with a sentence are stacked to form a matrix X as the input. A set of convolution operations are applied on the input matrix to combine the multiple word vectors into a compositional sentence representation.

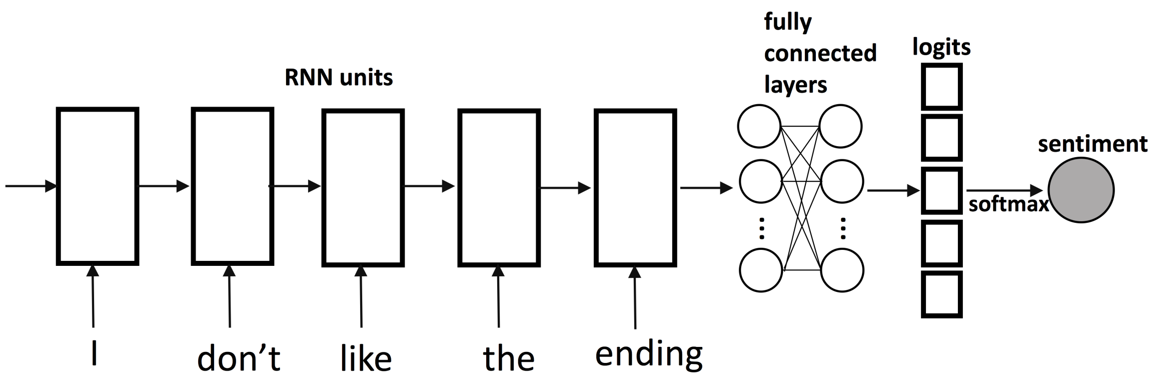


Figure 4.3: The application of RNN in sentiment classification

Assuming K convolution kernels h are used in a CNN model, the size of the kth kernel is $d * n$, where d is the dimensionality of the word vectors, and n is the number of words each sliding window of the kernel crosses. At a window crossing from the m^{th} word to the $(m+n)^{th}$ word, the feature results from the convolution operation between the kernel and the input matrix is:

$$s_{km} = g([x_m : x_{m+n}] \circ h_k + b_k) \tag{4.5}$$

where $\circ$ is Hadamard product between the n word vectors and the convolution kernel, g is the activation function, b_k is the bias term for the kernel. The features of all the sliding windows of the kernel are concatenated as a feature map s_k. The max pooling is then applied on s_k to take the maximum value $\hat{s}_k = \max s_k$ as the final feature corresponding to h_k.

The features of all convolution kernels are concatenated as the compositional representation $\hat{s}$ of the input sentence, that is further fed into some dense layers. The resulting logit vector of the subsequent dense layers is used to computed the probability distribution over the target classes. The structure of CNN is shown in Figure 4.2 (In the figure, the features marked by the blue colour result from convolution operations crossing two words at each step, the red one 3 words.).

4.3.3 LSTM and GRU

LSTM and GRU are two variants of the recurrent neural network (RNN). The many-to-one structured RNN is usually used for sentiment classification. In this structure, each word position of a sentence is treated as a time step, and the word vector at each position is fed into the RNN unit of the corresponding time step. The hidden state of the last RNN unit is used as the sentence representation. In practice, the plain RNN model is rarely used because of its ineffectiveness in the word order of long sentences. Instead, GRU and LSTM, are frequently used to train the classifiers.

Assuming the hidden state of time step $t-1$ is $h^{<t-1>}$ in a GRU model, then the hidden state of time step t is computed as follows:

$$r^{<t>} = \sigma(W_r x^{<t>} + U_r h^{<t-1>}) \tag{4.6}$$

$$z^{<t>} = \sigma(W_z x^{<t>} + U_z h^{<t-1>}) \tag{4.7}$$

$$\tilde{h}^{<t>} = tanh(W x^{(t)} + U(r_t \circ h^{<t-1>})) \tag{4.8}$$

$$h^{<t>} = (1 - z^{<t>})h^{<t-1>} + z^{<t>}\tilde{h}^{<t>} \tag{4.9}$$

Where $x^{<t>}$ is the word vector at time step t, σ is the sigmoid function. $\tilde{h}^{<t>}$ is the candidate value for $h^{<t>}$, $r^{<t>}$ is to govern how much the existing hidden state should be related to the previous state, $z^{<t>}$ how much information from previous time step should be kept at the existing step.

In the LSTM model, at a time step t, besides the hidden state $h^{<t>}$, there also exists the 'memory cell state' $c^{<t>}$. That two states can be computed as follows:

$$i^{<t>} = \sigma(W_i x_t + U_i h^{<t-1>}) \tag{4.10}$$

$$f^{<t>} = \sigma(W_f x^{<t>} + U_f h^{<t-1>}) \tag{4.11}$$

$$o^{<t>} = \sigma(W_o x^{<t>} + U_o h^{<t-1>}) \tag{4.12}$$

$$g^{<t>} = tanh(W_g x^{<t>} + U_g h^{<t-1>}) \tag{4.13}$$

$$c^{<t>} = f^{<t>} \circ c^{<t-1>} + i^{<t>} \circ g^{<t>} \tag{4.14}$$

$$h^{<t>} = o^{<t>} \circ tanh(c^{<t>}) \tag{4.15}$$

where $g^{<t>}$ is the candidate value for $c^{<t>}$, $i^{<t>}$ and $f^{<t>}$ govern how much information from the previous memory cell state and the candidate memory cell state should be kept, respectively, $o^{<t>}$ controls how the hidden state $h^{<t>}$ is related to the memory cell state.

The hidden state at the last time step is further fed into some dense layers, and the logit vector in the output layer is used to compute probability distribution over all the target classes. The structure of the RNN model is shown in Figure 4.3.

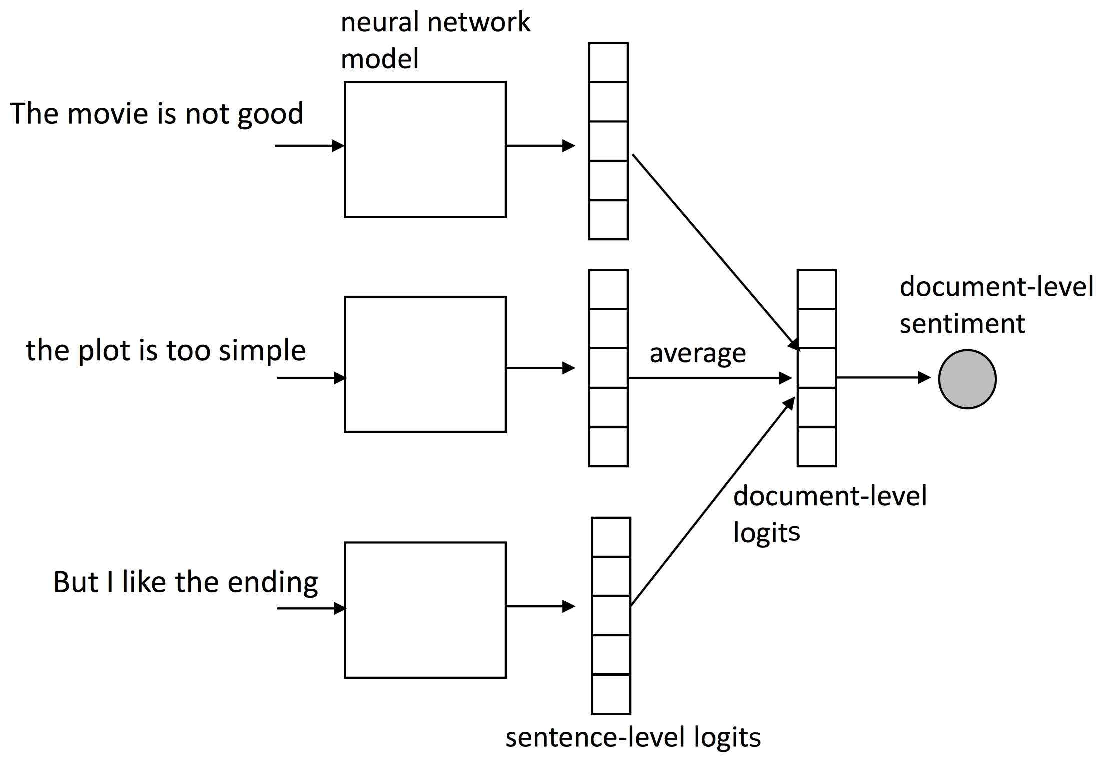


Figure 4.4: The structure of the Averaged-logits model

4.3.4 The structure of the Averaged-logits model

Assuming a neural network model is used to train the classifier. Given a document d consisting of N sentences, each sentence is fed into the neural network. As previously mentioned, each sentence will result in a logit vector in the output layer of the neural network model, that is usually used to computed the probability distribution of the sentence over all the sentiment classes. If the sentence-level ground-truth labels are available, the losses of the model can be computed by minimising the cross entropy between the labels and the probability distributions.

Since we wish to avoid the laborious sentence labelling efforts, the sentence-level labels are not available. Instead, we assume only document-level labels are available and use them as the supervision signal to train the sentence-level classifier. To achieve that, a way to associate the sentence-level logit vectors with document-level labels have to be defined. In the proposed approach, we first average over the logit vectors of sentences from the same review into a single logit vector, then compute the losses based on the averages and document-level labels. The approach can be formulated as follows.

Assuming the resulting logit vector of a sentence n of a review d in a neural network model is $Z_d^{(n)}$

$$\bar{Z}_d = \sum_{n=1}^{N} Z_d^{(n)} \tag{4.16}$$

where $\bar{Z}_d$ is the average of the sentence-level logit vectors for all the N sentence in document d, $\bar{Z}_d$ is used to compute the document's probability distribution $p(\ddot{y})$ over the target classes:

$$p(\ddot{y} = i|d) = \frac{\exp \bar{z}_{di}}{\sum_{j=1}^{C} \exp \bar{z}_{dj}} \tag{4.17}$$

Then the loss on the document is computed as follows:

$$l_d = -\sum_{i} 1\left\{\ddot{y} = j\right\} \log p(\ddot{y} = j) \tag{4.18}$$

A pictorial demonstration of the proposed approach is shown in Figure 4.4. The training process minimises the losses in (4.18) to derive the unknown parameters of the sentence-level models by the back-propagation technique.

4.4 Evaluation

This section presents the evaluation results. We collected two datasets, one contains 50,000 electronic product reviews from Amazon, the other 50,000 hotel reviews from Tripadvisor. Each review consists a piece of text, and an overall rating on a 5-point scale that can serve as the document-level sentiment label. Statistics of the two datasets are shown in Table 4.1

Table 4.1: Statistics of the review datasets

	Num of reviews	Avg num of sentences	Avg num of words	Num of sentiment classes
TripAdvisor	50,000	9	176	5
Amazon	50,000	12	126	5

We manually read part of the reviews, and labelled 7000 sentences for each dataset. Each sentiment label takes 3 possible categorical values: positive, negative, neutral. Statistics of the labelled sentence sets are shown in Table 4.2. We use the labelled review sentences to train classifiers based on the SVM, MLP, CNN, GRU and LSTM models. To train the classifiers, 5,000 labelled sentences are randomly chosen from each dataset as the training set, and the remaining as the test set.

Table 4.2: Statistics of the review sentence sets

	Training size	Test size	Avg num of words	Num of classes
TripAdvisor	5000	2000	22	3
Amazon	5000	2000	17	3

At the meantime, we use the reviews to train sentence-level sentiment classifiers based on the proposed approach and the existing approach treating documents as long sentences. The performance of those classifiers are also evaluated on the test set of labelled sentences. It is noteworthy that, since the ratings are on a 5-star scale, the predictions of the classifiers trained under the supervision of the ratings would also be values on that scale. Therefore, the predicted labels have to be converted into the same 'negative-neutral-positive' format used by the labelled sentences in the evaluation process. In this paper, the following conversion scheme is used: 'negative' when predictions are less than 2 stars, 'neutral' when equal to 2 stars, 'positive' otherwise. The performance comparison among all the classifiers will be shown in 4.4.3.

Naturally, the performance of the classifiers trained with the proposed approach should be lower than that of the classifiers trained with sentence-level labels when their training sizes are close [1]. However, by increasing the training sizes of the proposed approach simply by adding more raw reviews, the performance gap is expected to narrow down. Details of the effects of training sizes on the performance of the classifiers will be shown in 4.4.4.

It is also possible to mix raw reviews with labelled sentences to train the classifiers with the proposed approach. In this case, each labelled sentence is treated as a review consisting of only one sentence, the label of the sentence is treated as the document-level sentiment label. Once again, the ratings of the reviews are on a 5-point scale, therefore, have to be converted into the negative-neutral-positive format to be mixed with the labelled

[1]When labelled sentences are used to train the classifiers, each sentence is counted as a training sample; when reviews are used as the training data, each review is counted as a training sample

sentences. The aforementioned scheme is also used for the label conversion process. The performance of the classifiers trained on the mixed data will be shown in 4.4.5.

4.4.1 Pre-processing and Hyper-parameter Setting

In the experiment, other than sentence tokenisation and removing the words that appear for less than 5 times in the datasets, no additional pre-processing treatment is performed. Google's W2V model [2] pre-trained on the Google news corpus is used to represent words. The dimensionality of the W2V vectors is 300. In the training process, the W2V vectors are set as non-trainable. The Keras package with Tensorflow backend [3] is used to implement both the baseline models and the proposed approach.

4.4.2 Convergence of Averaged-logits

In the experiment we observed that the training process of Averaged-logits converges quickly and steadily. The training and validation accuracies of the proposed model on each dataset over epochs are visualised in Figure 4.5 (It is noteworthy the validation data is a set of around 500 reviews, that are randomly drawn from the training data). As shown in the figure, the validation accuracies of all the classifiers based on Averaged-logits begin to stabilise at approximately the 30th epoch. Also, the classifiers trained on the Amazon dataset have a more obvious overfitting problem. One possible reason is that the average length of the Amazon reviews is shorter than that of the TripAdvisor reviews, therefore, the dataset provides less information for Averaged-logitsl to fit on.

4.4.3 Comparison of Averaged-logits with the baselines

As mentioned previously, we train three types of classifiers: generic classifiers trained with the datasets of 5,000 review sentences; generic classifiers trained with the review datasets; Averaged-logits classifiers trained with the review datasets. To make a fair comparison, 5,000 reviews are only drawn from each review dataset for the latter two types of classifiers to make their training sizes equal to that of the firs type of classifiers. The following models: SVM, MLP, CNN, GRU, LSTM are used to train the first and

[2]https://code.google.com/archive/p/word2vec/

[3]https://www.keras.io/

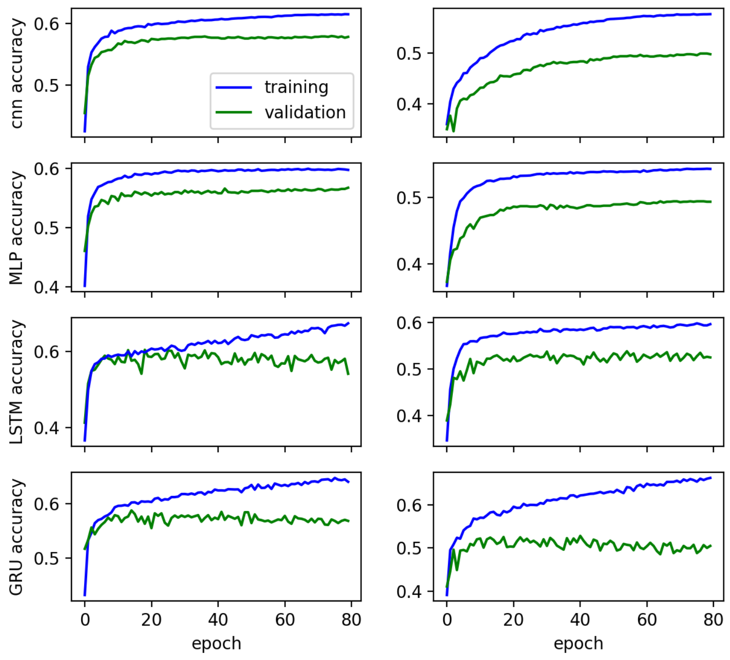


Figure 4.5: The training and validation accuracies of Averaged-logits on each dataset.

second types of classifiers; MLP, CNN, GRU, LSTM are used to train the Averaged-logits classifiers. The hyper-parameters of each model is decided by the grid-search technique on each dataset. Details of the hyper-parameters are shown in the accompanying code and therefore being omitted in the paper.

The performance comparison in terms of accuracy among all the models is shown in Table 4.3. We first compare the performance of the first type of classifiers trained using labelled sentences, against the performance of the Averaged-logits classifiers trained using reviews. As the results indicate, the performance of the Average-logits classifiers, is close to that of the SVM classifier, and around 6-10% lower than that of other classifiers in the first type.

We then compare the performance of the Average-logits classifiers against the second type of classifiers trained by treating documents as long sentences. As the results show that, Averaged-logits classifiers enjoys good margins of 6%-8% over the second type of classifiers on the Amazon dataset, and 3-5% on the TripAdvisor dataset. The advantage of the Averaged-logits model is more obvious on the Amazon dataset. One possible reason is that there are more cases where the sentiments of individual sentences are opposite to that of the reviews in the Amazon dataset.

Table 4.3: Performance of the three types of classifiers: generic classifiers trained with review sentences; generic classifiers trained with the review datasets; Averaged-logits classifiers trained with the review datasets

Model	Structure	Training data type	Features	TripAdvisor	Amazon
SVM	generic	labelled sentences	ParaVec	0.71	0.66
	generic	reviews	ParaVec	0.58	0.49
MLP	generic	labelled sentences	W2V mean	0.75	0.70
	Avg-logits	reviews	W2V mean	0.64	0.55
	generic	reviews	W2V mean	0.68	0.63
CNN	generic	labelled sentences	W2V	0.81	0.75
	Avg-logits	reviews	W2V	0.68	0.62
	generic	reviews	W2V	0.73	0.69
LSTM	generic	labelled sentences	W2V	0.77	0.73
	Avg-logits	reviews	W2V	0.66	0.58
	generic	reviews	W2V	0.69	0.65
GRU	generic	labelled sentence	W2V	0.80	0.72
	Avg-logits	reviews	W2V	0.66	0.60
	generic	reviews	W2V	0.71	0.66

4.4.4 Impacts of training sizes

To improve the performance of the classifiers, one can use larger training datasets to re-train the classifiers. However, for the first type of classifiers, increasing the training size comes at the heavy cost of labelling many more review sentences, therefore, is out of the question in this experiment. The training sizes of the second type of classifiers and the Average-logits classifiers can be increased simply by adding more reviews into the original training sets. In the experiment, the raw reviews are divided into 5 portions. We iteratively increase the training sizes of the classifiers by adding one portion of the reviews at a time into the training set. The performance of the classifies with varying

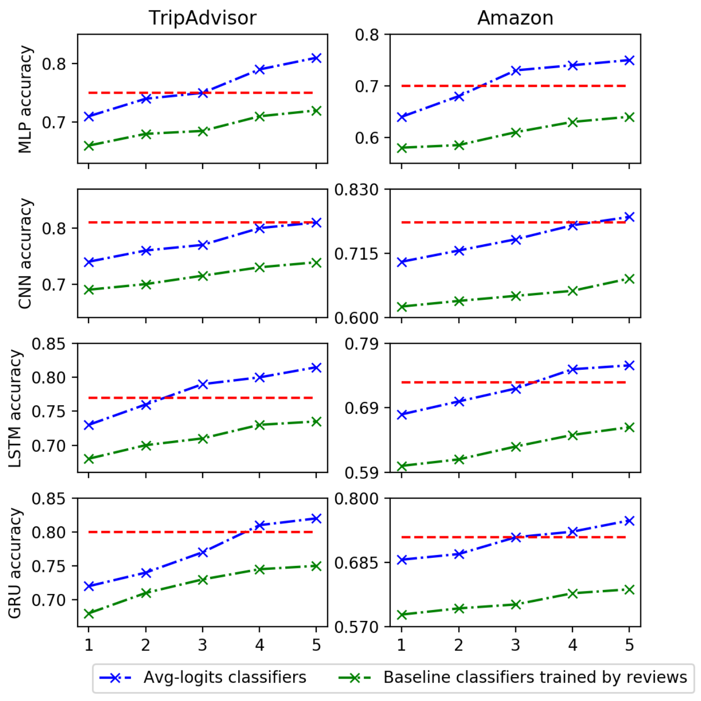


Figure 4.6: Performance of the proposed approach and the existing approach trained by varying percentages of the original review dataset.

sizes of training data is shown in Figure 4.6 (The red dash line is the performance of the first type of classifiers trained with the 5,000 review sentences).

The figure indicates, as the training size increases, the performance of all the classifiers increases. However, the performance of the Averaged-logits classifiers increases more obviously and quickly. On both datasets, when 3 or 4 portions of reviews are added into the training sets, the performance of all the Averaged-logits classifiers is close to that of the first type of classifiers trained using the 5,000 sentences; whereas the performance of the second type of classifiers is still obviously lower than that of the first type even all the 5 portions of reviews are added into the training datasets.

4.4.5 Mixing reviews and labelled sentences as the training data

As previously mentioned, it is also possible to mix labelled sentences and reviews to train the classifiers. In the experiment, we also divide each review sentence dataset into 5 portions (1000 sentences in each portion), and mix them with the review datasets. We mix one portion of the reviews and one portion of review sentences as the training set at

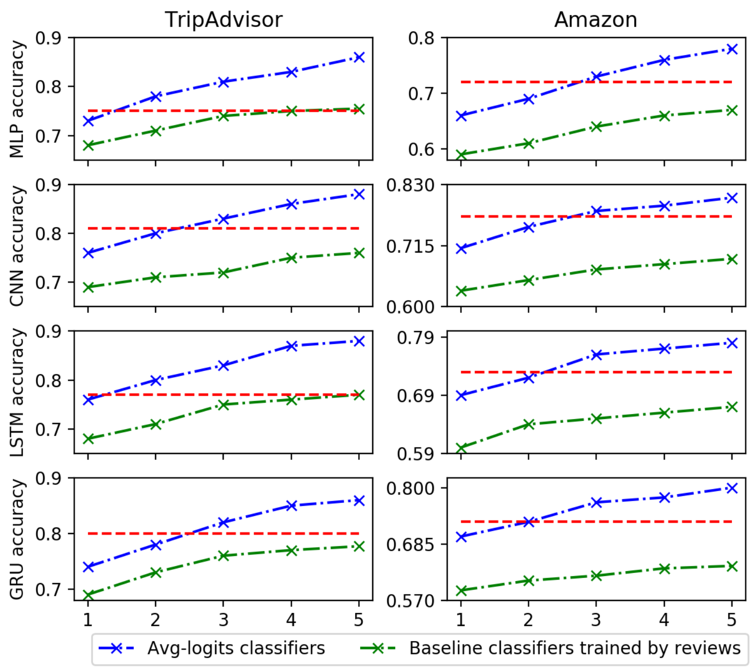


Figure 4.7: Performance of the classifiers trained with the mixed data

the beginning, and proceed by iteratively adding one portion of the both types of data at each time into the training sets, to train the classifiers. The performance is shown in Figure 4.7.

We compare the performance of the classifiers trained with the mixed data against that of the classifiers trained only with the reviews. The results indicate that, the addition of all the review sentences helps improve the performance of the Averaged-logits classifiers by a margin of 5% on the Amazon dataset, 7.5% on the TripAdvisor dataset; whereas improves the performance of the generic classifiers by 3% on the Amazon dataset, and 4.5% on the TripAdvisor dataset. The improvements on the Averaged-logits classifiers are more obvious. As shown in the figure, the review sentences result in little change for the generic classifiers after 3 portions of reviews are used. Also, it is noteworthy that the performance of the generic classifiers trained with the mixed data is still below that of the generic classifiers trained only with the review sentences. One possible reason is that, each unit in the input of the generic models is a review, that contains much more words therefore has dominantly stronger impacts in the training process. This makes it difficult for the models to capture the information of the added review sentences.

The performance of the Averaged-logits classifiers trained with the mixed data surpasses that of the classifiers trained with review sentences more quickly and strongly than do the Averged-logits classifiers trained with only the reviews. This might be because the mixed data allows the classifiers to capture both the sentence-level and document-level statistical patterns.

4.5 Conclusion

In review mining, a majority of existing methods for sentence-level sentiment classification are based on supervised machine learning models, that require a large number of labelled sentences for the training process. In practice, the labelled data usually takes intense manual efforts to obtain. To avoid or reduce the manual efforts, one can use reviews, that are much less expensive even free to obtain, to train sentence-level sentiment classifiers. In this approach, each review document is treated as a long sentence, the rating of the review is treated as the sentiment label of the long sentence. However, there exists an obvious problem with this approach: it loses the sentiment signals of individual sentences in the training process.

In this paper, we propose a novel approach, called 'Averaged-logits', to address the problem. The proposed approach assumes each sentence in a review has its own sentiment label, and the average of the sentence-level sentiments should be close to the rating of the review. The evaluation results show that, the proposed approach outperforms the existing approach by good margins of 3%-8%. In the future, we plan to train the classifiers based on the proposed approach on a much larger training set to have better understanding of the true potential of the proposed approach.

Chapter 5

An machine learning approach to identify the attractive aspects

This chapter proposes a machine learning based approach to identify the attractive aspects of hotels based on review data. The main challenge in building such a classifier is extracting discriminative features to represent each aspect. The proposed approach extracts two types of features, named 'empirical' and 'interactive effect' features, that reflect the asymmetric relations between attribute-level performance and overall customer satisfaction based on the results of the opinion classification process, to represent each aspect. The two types features are fed into the SVM model to train the classifiers classifiers. Experiment results show that the proposed approach outperforms the existing statistical model based approaches by a margin of 10%.

5.1 Introduction

The Kano model [13] has been widely used by researchers and marketing practitioners as a good tool to identify the drivers of customer satisfaction. According to the model, all the attributes of a product or service can be divided into 3 main categories: must-be attributes, one-dimensional attributes, and attractive attributes. Must-be attributes are the minimum requirements that customers expect. Extreme dissatisfaction is caused if their performance is insufficient while satisfaction is not necessarily caused if their performance is sufficient. One-dimensional attributes are also expected by customers but they differ from the must-be aspects in affecting customer satisfaction in a linear way: satisfaction is caused if their performance is sufficient and dissatisfaction is caused if their performance is insufficient. Attractive attributes are usually not expected by customers and their effects are opposite to that of must-be attributes: satisfaction is caused if their performance is sufficient but dissatisfaction is not necessarily caused if their performance is insufficient.

Categorising product attributes into those 3 categories helps people understand better the drivers of customer satisfaction [83]. In particular, pinpointing the attractive attributes is of strong interest to marketing practitioners because attractive attributes can distinguish a product or service from its competitors. In a mature market where players are homogenous, attractive attributes are usually vital to boost customer satisfaction and expand customer bases. In this research we focus only on the attractive attributes and aim to build a model to classify an attribute of a product or service as whether belonging to the attractive category.

The propensity of an attribute belonging to the attractive category can be decided by using regression techniques to estimate the effects of its performance on customer satisfaction. The first phase in the classification task is to collect customer opinions about their responses to various levels of attribute performance. In the early days of marketing research, researchers usually relied on quantitive questionnaire-based surveys to get the opinion data [84–87]. Since questionnaire-based surveys use a limited set of predefined questions and can investigate only a trivial fraction of a population, it provides a relatively narrow spectrum of information, especially in today's era of big data. As e-commerce grew rapidly in recent years, on-line reviews have been used to extract the opinions. Because of the accessibility and flexibility customers enjoy on on-line review platforms, the data can cover much broader demographic groups and deliver richer information than the questionnaire-based survey data.

However, there is a problem when trying to use review data for the classification task. Review data usually consists of numerical ratings that indicate overall customer satisfaction levels and free-form text that details customers' evaluation on each attribute. Obviously, the qualitative textual description cannot be processed by regression tools. To address the problem, researchers have proposed approaches [88–91] that first use natural language processing (NLP) techniques to extract customer sentiments associated with each attribute as the quantitive proxy for the qualitative description, then apply statistical regression models to derive the effects of the attribute-level sentiments on the overall ratings as an approximation for the effects of attribute performance on customer satisfaction.

A major weakness of those works lie in their ineffectiveness in encoding the non-linearity of the effects. Also, those works treat the items to be analysed as isolated and their regression processes are performed on the local data of individual items. The global statistical patterns across the attributes of different items are overlooked. Those limitations lead to severe loss of information. Furthermore, the attribute-level sentiment analysis process usually generates heavy noise as the NLP techniques are still far from being mature. Ignoring the global statistics may also cause those methods to be sensitive to the noise.

In this study, the review data is also used to collect the customer opinions. However, this study differs its research goal from the existing ones by using machine learning models instead of statistical models to train classifiers to determine the Kano categories of attributes. To train such classifiers, numerical features that reflect the aforementioned effects have to be derived from the review data to represent attributes. A very natural form of attribute representation is the empirical probabilities of ratings conditioning on various sentiments associated with each attribute. To compute the representation, the effects of the sentiments have to be assumed as independent. However, in a realistic situation, the sentiments should closely interact with each other in determining the overall ratings; ignoring these interactions may lead to misrepresentation of attributes. Modelling the interactions is difficult as it is essentially a regression problem with a very high degree of nonlinearity. In this research a neural network model is proposed to encode the non-linearity. In the neural network, the input is the attribute-level sentiments of a review and the output is the possibility of each possible rating for the review. The weights of a particular hidden layer are used as the attribute representation incorporating the impacts of the interactions. For the convenience of expression, in the rest of this chapter, the former type of representation will be called ‘empirical effect’, and the latter type will be called ‘interactive effect’.

SVM classifiers are trained with the derived attribute representations. Since the majority of the existing statistical models focus on hotel review data [88–91], this study also aligns the target on hotel review data for a fair comparative evaluation. The experiment results show that the proposed classifiers outperform the statistical models by a considerable margin of over 10%. This is partly because the classifiers make use of the global statistics and can better encode the non-linearity of the training data. The contributions of this chapter are as follows. Firstly, we are the first to use machine learning techniques to train predictive classifiers to identify attractive attributes from on-line review data. Secondly, we propose to use neural network techniques to derive the non-linear interactions among different levels of attribute performance in determining overall customer satisfaction.

A flowchart of the proposed approach is shown in Figure 5.1. The rest of this chapter is organised as follows. Section 2 reviews the related literature; Section 3 introduces the proposed attribute representation models; Section 4 presents details of the attribute-level sentiment analysis; Section 5 analyses the data and summarises the results; Section 6 provides conclusions and future research directions.

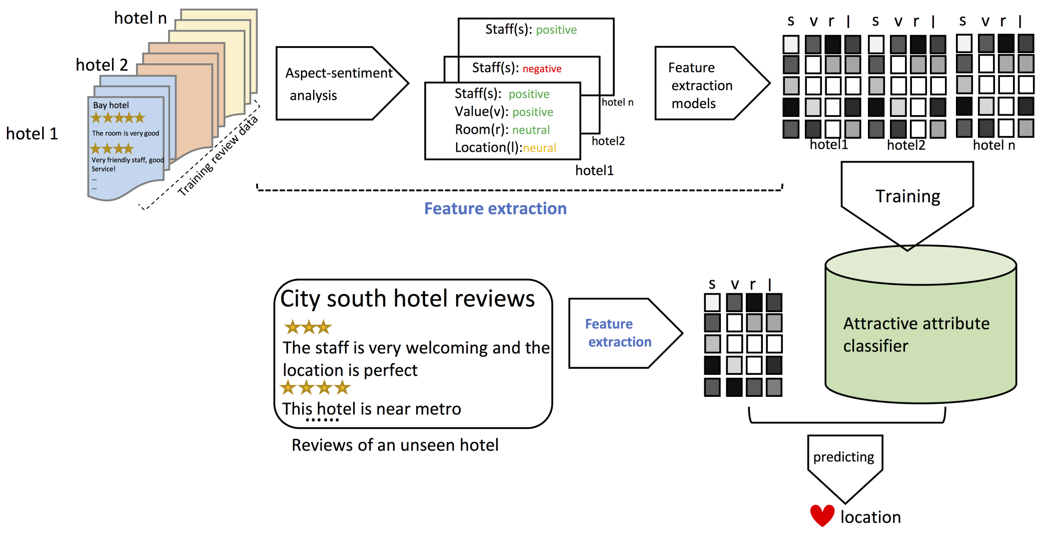


Figure 5.1: The workflow of the proposed attractive attribute identification process

5.2 Related work

In the proposed approach, the premise to identify the attractive attributes is devising natural language processing techniques to obtain structured customer opinions from the unstructured review text. In this research, the opinion extraction process is treated as an attribute-level sentiment analysis problem [92], that is to detect the attributes mentioned in a review and the sentiment orientations on them. This is the first major subproblem that needs to be discussed. Based on the structured customer opinions, associations between attribute performance and customer satisfaction have to be derived to determine the Kano category of each attribute. This is the second subproblem that needs to be investigated. In this section some representative works related to the two problems are presented.

5.2.1 Attribute-level sentiment analysis

As the name suggests, attribute-level sentiment analysis consists of two sub-classification problems: sentiment classification and attribute classification. The two tasks can be performed either separately or jointly. Various off-the-shelf supervised classification models can be used if they are treated as two separate tasks. A key problem in the classification tasks is deriving features to represent the text. The traditional bag-of-words based feature models, such as one-hot-encoding, tf-idf, etc., have been widely used to represent the review text [57, 61, 93–95]. However, the bag-of-words based features are usually very noisy because reviews are very short and the frequencies of informative words in a review are too low to reflect their semantic meaning. To reduce the noise, lexicons of attribute or sentiment keywords can be used and the representation of a review is defined only by the keywords present in the review [1–4, 96–98]. In recent years, text embedding models that project textual data into a semantic space to derive their semantic representations have been proven to be effective in the two classification tasks [9, 20, 36, 50, 99]. The elements of an embedding feature vector represent the membership of a word or a text fragment in a limited set of abstract senses. Compared with the bag-of-words based features, the embedding representations are much semantically richer, lower-dimensional, and therefore, easier to be processed by machine learning models.

In the joint approach, the aspect labels and the sentiment labels are included in the same cost functions and the syntactic or semantical dependencies between the two labels serve as the constraints for the cost functions. Optimising such cost functions will generate the two types of labels simultaneously. Methods based on conditional random field [2,100, 101] and topic modeling [72, 102–105] have been proposed for this approach. Compared with the separate approach, the joint approach is naturally more computationally efficient. Furthermore, some of those methods [72, 102, 104] are unsupervised and do not require manual labor to annotate the training data. However, the performance of the methods in the joint approach is usually worse than the separate approach because they rely on strong assumptions about the correlations between the attribute and sentiment labels that are not necessarily realistic in the real world.

5.2.2 Statistical Kano classification models

With the extracted customer opinions, methods based on penalty-reward contrast analysis(PRCA) [106] can be used to classify each attribute into 1 of the 3 categories. PRCA based methods are focused on estimating the impacts of each attribute on customer satisfaction. The impacts of an attribute consist of the reward impact and the penalty impact. The former represents the impact when the performance of the attribute is sufficient, and the latter indicates the impacts when performance is insufficient. If the reward impact of an attribute is stronger than the penalty impact, then the attribute is very likely to be an attractive attribute. Tontini et al. [90] first perform attribute-level sentiment analysis on review data, then apply the linear regression model to derive the relations between the sentiments and ratings as an approximation of the impacts. Zhang et al. [91] extract keywords and their sentiment weights for each attribute from the review data to quantise the attribute performance. By conditioning on the derived attribute performance, the authors apply the logistic regression model to derive the reward and penalty impacts.

There are also existing models based on the critical incident technique (CIT) [107]. A critical incident refers to the incidents that cause significant contributions or damages to customer satisfaction. If an attribute frequently causes positive incidents in favourable reviews but rarely causes negative incidents in unfavourable reviews, then the attribute is very likely to be attractive. Lu et al. [89] identify the incidents in each attribute based on the attribute-level sentiments and employ the Krusal-Wallis test to determine the relation between the incidents and the overall ratings. Xu et al. [88] first divide all reviews into a positive group and a negative group, then represent all the words in each review group by a matrix of TF-IDF weights. The authors employ the latent semantic analysis (LSA) model to decompose the matrices to uncover the keywords and critical incidents for each attribute. The correlation between the incidents and each review group is determined by aggregating the weights of the corresponding keywords.

5.3 Attribute representations

As mentioned in Section 1, two types of representations, the empirical effects and the interactive effects, are derived from the review data to represent attributes to train the classifiers. Assuming there is a set of M review documents for an item, the mth review document is denoted as $d^{(m)}$. The evaluation of the item involves a set of K aspects. There are I different scales of sentiments; the larger a sentiment index i the more positive the sentiment. There are J scales of overall ratings; the larger a rating j the higher the overall customer satisfaction. The notation used throughout this section is shown in Table 5.1.

Table 5.1: Notation used throughout Section 3

m	review index
k	attribute index
i	sentiment index
j	rating index
$d^{(m)}$	the mth review document
$s_k^{(m)}$	the sentiment set on attribute k in $d^{(m)}$
$r^{(m)}$	the rating of $d^{(m)}$
u_k	the empirical effect features of k
v_k	the interactive effect features of k

5.3.1 Empirical effect

Let r denote the ratings, r_l and r_h are a rating falling in the lower half of the rating range and a rating in the higher half of the rating range, respectively. Let s_k denote the sentiments on attribute k, $s_{k_-}$and s_{k_+} are a negative and a positive sentiment on the attribute, respectively.

According to the definition, for must-be attributes, the probability of a negative sentiment leading to a low rating must be greater than the probability of a positive sentiment leading to a high rating, namely, $p(r_l|s_{k_-}) > p(r_h|s_{k_+})$; for attractive attributes, $p(r_h|s_{k_+}) > p(r_l|s_{k_-})$; for one-dimensional attributes, $p(r_l|s_{k_-}) \approx p(r_h|s_{k_+})$. In this research, the conditional probabilities associated with an attribute are concatenated to form a feature vector to represent the attribute:

$$u_k = [p(r = j|s_k = i)]_{i\in[1,I],j\in[1,J]} \tag{5.1}$$

$$p(r = j|s_k = i) = \frac{\sum_{m=1}^{M} 1\{i \in s_k^{(m)}, r^{(m)} = j\}}{\sum_{m=1}^{M} 1\{i \in s_k^{(m)}\}} \tag{5.2}$$

where $p(r = j|s_k = i)$ denotes the probability of rating scale j given sentiment index i associated with aspect k across all M reviews for the item, $s_k^{(m)}$ is the set of sentiments on attribute k in review $d^{(m)}$, and $r^{(m)}$ is the overall rating of the review.

According to the previously presented analysis, the empirical effects of attributes in the same Kano category would share some common patterns. However, there is an obvious weakness in the representation: it assumes aspects are independent of each other and does not reflect the interactions among attributes in deciding the overall ratings. The interactions can also provide information related to the nature of an attribute. For instance, when a basic need is poorly fulfilled, it is very likely that the probability $p(r_l|s_{k_-})$ would be pushed high and the conditional probabilities on other attributes would be lowered significantly. This is because customers may simply ignore the performance of other attributes if a must-be attribute fails to meet their expectations. In this chapter we model the interactions with the neural network model and show the details in the following subsection.

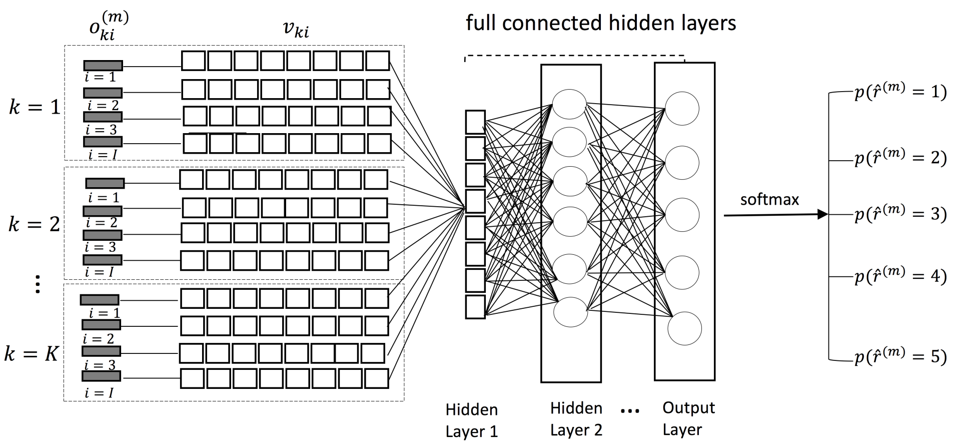


Figure 5.2: The structure of the neural network for extracting the interactive effect features

5.3.2 Interactive effect

This subsection assumes that the effects of attribute-level sentiments are dependent and connected in determining the overall ratings. Theoretically, one can hard encode the rules of how the effects interact with each other and use probabilistic models to derive the effects based on those rules. However, given the high variety and complexity of the effects, it is infeasible to construct such a model. Instead, this research uses the neural network model as a mapper that aggregates the effects of the attribute-level sentiments appearing in a review and maps them to the rating of the review. Since the output of the neural network, namely the ratings, is observable in the data, the back-propagation algorithm [108] can be used to derive the mapping function of the neural network and the interactive effects.

In the proposed neural network, the input is a vector indicating the presence or absence of each attribute-level sentiment: $O^{(m)} = [1\{i \in s_k^{(m)}\}]_{k\in[1,K],i\in[1,I]}$. Let $V = [v_{ki}]_{k\in[1,K],i\in[1,I]}$ denote the matrix of interactive effects of all attribute-level sentiments. The activations of the first layer are the weighted average of the effects of the attribute-level sentiments present in review $d^{(m)}$:

$$A^{[1]} = W^{[1]\top}(O^{(m)} \otimes V) + b^{[1]} \tag{5.3}$$

where $W^{[1]}$ is the vector of weights for the attribute-level sentiments, $b^{[1]}$ is the bias term, $O^{(m)} \otimes V$ is the element-wise product of the indicator vector and the interactive representation matrix. By applying the element-wise product operation, only the attribute-level sentiments present in the review play roles in determining the overall ratings.

The activations in the first hidden layer are fed into subsequent layers in which neurons are fully connected to model the interactions among the attribute-level sentiments in deciding the overall ratings. Assuming there are L hidden layers in the neural network, the activations of a hidden layer $l \in (1, L]$ can be computed as follows:

$$A^{[l]} = g(W^{[l]} A^{[l-1]} + b^{[l]}) \tag{5.4}$$

where g is the tanh activation function of the neural network, $W^{[l]}$ is the matrix of weights of the connections between layer $l-1$ and layer l, $b^{[l]}$ is the bias term. In the output layer, the softmax function σ is used to compute the probability of each possible rating for the review:

$$A^{[o]} = \sigma(W^{[o]} A^{[L]} + b^{[o]}) \tag{5.5}$$

where $A^{[o]} = [p(\hat{r}^{(m)} = 1), p(\hat{r}^{(m)} = 2), ...p(\hat{r}^{(m)} = J)]$, $p(\hat{r}^{(m)} = j)$ is the probability to predict a rating of j for the review. The negative log likelihood is used as the loss function of the proposed model:

$$C^{(m)} = -\sum_{j \in [1,J]} 1\left\{r^{(m)} = j\right\} * \log p(\hat{r}^{(m)} = j) \tag{5.6}$$

The backward propagation method [108] is used to derive the unknown interactive effects and the hidden layer weights and biases. The structure of the neural network is shown in Figure 5.2. The hyper-parameters of the structure, including the dimensionality of the interactive effect features, the number of hidden layers, and the number of activations in each hidden layer, are also unknown beforehand and decided by the grid search technique [?] in the training process. Details of the hyper-parameter search process will be shown in Section 5.

After the training process, the learned interactive effects of all the sentiments associated with an attribute k are concatenated to represent the attribute: $v_k = [v_{ki}]_{i \in [1,I]}$. Furthermore, we concatenate the interactive effects to the aforementioned empirical effects to form a more informative representation for the attribute.

5.4 Extracting attribute-level sentiments

This section introduces the proposed approach for attribute-level sentiment analysis. The attribute classification and sentiment classification are treated as two separate problems. In other words, two separate classifiers are trained for the attribute classification and the sentiment classification. In the proposed approach, a review is decomposed into sentences and the classifications are performed on each individual sentence. Assuming a sentence of a review $d^{(m)}$ is classified as discussing about attribute k with sentiment i, then the sentiment i is added to the sentiment set for attribute k to make $i \in s_k^{(m)}$ true. In this research, the off-the-shelf sentiment classifier in the StanfordNLP package [109] is used for the sentiment classification and the following method is used for the attribute classification.

Word-to-vector (W2V) [9], a representative text embedding model, is used to represent the review text because it reflects the semantic information and allow for computing the similarity between any pair of words in the vocabulary. Assuming there exists a set of keywords ϕ_k for each aspect k, the similarity between a review sentence and an given aspect k is computed as follows:

$$sim\left(d^{(m,n)}, k\right) = \max_{e_1 \in d^{(m,n)}, e_2 \in \phi_k} Cos(f_{e_1}, f_{e_2}) \tag{5.7}$$

where $d^{(m,n)}$denotes the nth sentence in review $d^{(m)}$, $e_1 \in d^{(m,n)}$ denotes each word in the review sentence, $e_2 \in \phi_k$ denotes each keyword for aspect k, f denotes the word-to-vector features of a word, Cos denotes the cosine similarity between a pair of feature vectors. The equation indicates that we compute the similarity between each word of a sentence and each keyword of an aspect and use the maximum similarity as the similarity between the sentence and the aspect.

If the similarity between a sentence and an aspect exceeds a predefined threshold γ, then the sentence is deemed to be about the aspect:

$$z^{(m,n)} = \arg_k sim(d^{(m,n)}, k) > \gamma \tag{5.8}$$

where $z^{(m,n)}$ is the set of aspects the sentence may discuss about.

5.5 Evaluation

A dataset containing around 60,000 reviews of 400 hotels in New York city is collected from TripAdvisor [3]. In this dataset, each hotel has at least 100 reviews. Each review has an integer rating in the range $[1, 5]$ and a piece of review text. The distributions of customer opinions may vary significantly by room rate, therefore, all the hotels are divided into 4 categories according to their room rates: cheap, budget, medium, and luxury. Statistics of each category of hotels are shown in Table 5.2.

Table 5.2: Statistics of each hotel category

Hotel type	Price range(USD)	Number of hotels	Number of reviews	Average rating	Standard deviation of ratings
Cheap	0-100	86	18567	3.23	1.75
Budget	100-200	114	20463	3.05	1.64
Medium	200-400	120	26742	3.39	1.67
Luxury	above 400	85	15064	3.71	1.26

Table 5.3: The Kano matrix

		Dysfunctional question How do you feel if the performance of the attribute is insufficient?		
		I am neutral	I can live with it that way	I dislike it that way
Functional question How do you feel if the performance of the attribute is sufficient ?	I like it that way	A	A	
	It must be that way			
	I am neutral			

To get the ground truth labels for the training process, all the reviews of each hotel are evenly divided into 15 portions. 15 volunteers with background in hotel management are employed to read the reviews, each volunteer is assigned one portion of the reviews. Based on their understanding from the reading, each volunteer is asked to fill in the Kano matrix shown in Table 5.3 for each attribute discussed in the reviews. According to the Kano theory [13], if the answer of a volunteer for an attribute falls in the cells marked with 'A' then the attribute is attractive. If more than 7 of the 15 volunteers give the attractive response then the attribute is labelled as attractive, otherwise as non-attractive.

In the attribute-level sentiment analysis, as previously mentioned in Section 4, the sentiment classifier provided by StanfordNLP [109] is used to determine the sentiment of each review sentence. In this classifier each sentiment is indicated by an integer in the range $[1, 4]$, such that close to 4 indicates a strong positive sentiment while close to 1 indicates a strong negative sentiment. The proposed method described in Section 4 is used to detect the attributes in each review sentence. To use the method, the word-to-vectors features have to be learned from the review text. Since the aspects of a sentence are mostly characterised by the nouns, verbs, adjective, and adverbs, the part-of-speech tagger [109] is used to select those words from each sentence as the input of the word-to-vector model. The number of dimensions of the representation is set to be 100.

A set of attributes shown in Table 5.4 that are frequently discussed in the existing hospitality research papers [88–91] are used as the possible labels the proposed method would assign to a review sentence. Also, 10 keywords for each attribute are obtained by consulting a volunteer expert in the hospitality industry. We set the threshold γ in Equation (8) to 0.6, on which the best performance is observed.

Table 5.4: The rating attributes of the reviews

General hotel attributes	lobby, accessibility, parking, shuttle services, check-in/out, staff, price, reservation, nearby leisure facility, taxi-calling
General room attributes	cleanliness, quietness, room size, furniture, bedding, AC, view, bathroom, decoration, internet, pet-friendliness
Room appliances	fridge, microwave, coffee-maker, computer
Food related attributes	breakfasts, affiliated restaurants, complimentary food and drinks
Additional facilities	conference rooms, gym, pool

5.5.1 Aspect identification

The performance of the proposed attribute detection method is compared with the following methods: the supervised Naive Bayesian [110] and SVM model [94] trained with the bag-of-words based tf-idf features, the lexicon based Boot-strapping method [3], and the SVM model trained with the word-to-vector features [99]. The comparison results are shown in Table 5.5. The results indicate that the proposed method outperforms the best among the first 3 methods by a margin of 5% in terms of the F1 score. The SVM classifier trained with word-to-vector features outperforms the proposed method, but the advantage comes at the price of tedious manual labor for labelling the training data. Also, since the threshold γ in Equation (8) allows the proposed method to assign aspect labels to an instance only when it is 'extremely' confident about the decision, the proposed method usually assigns no label, while other models usually assign wrong labels in the false negative cases. This may result in less noise in analysing the correlations between attribute-level sentiments and the overall ratings, especially when the training data comes in very large volume.

We also show the number of review sentences related to each aspect across the whole dataset in Figure 5.3. The results indicate that staff is the most frequently discussed attribute in the dataset. This is consistent with the fact that professional ethics and competence of staff members are crucial elements to increase customer satisfaction in the hospitality industry.

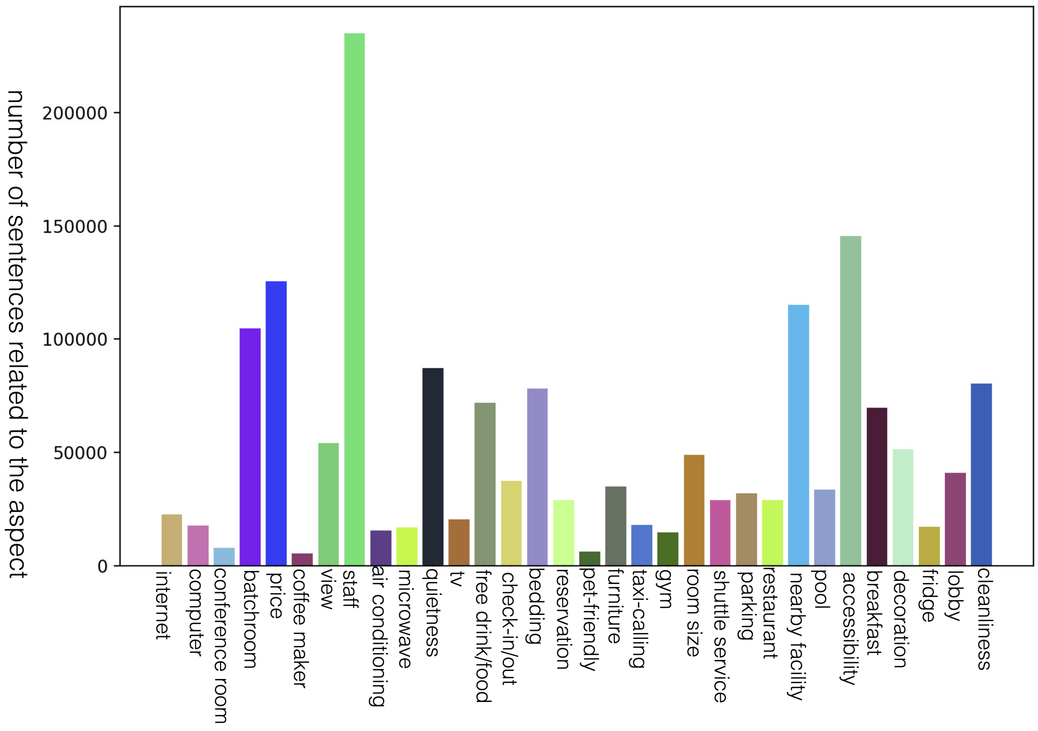


Figure 5.3: Number of sentences related to each aspect

Table 5.5: Performance of the attribute classification models

	SVM +tf-idf	Naive Bayesian +tf-idf	Lexicon-based boots-strapping	SVM+W2V	Our model
Average precision	0.80	0.72	0.82	0.88	0.85
Average recall	0.66	0.62	0.60	0.79	0.70
Average F1 score	0.72	0.67	0.69	0.83	0.77

5.5.2 Empirical effect

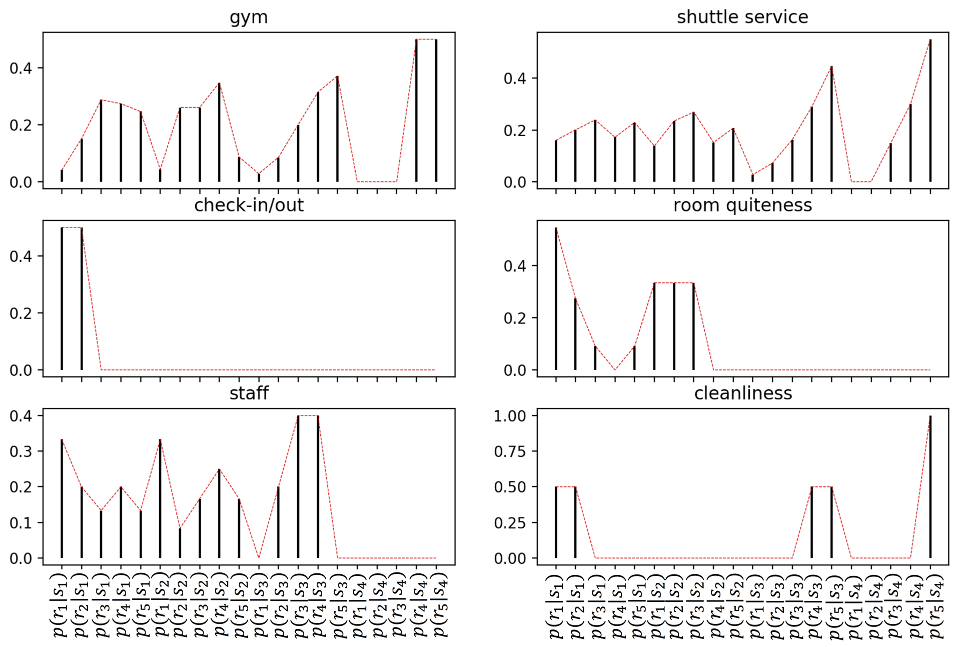


Figure 5.4: A snippet of the empirical effect features

The tick $p(r_j|s_i)$ represents the conditional probability of rating j given sentiment i associated with an aspect.

As previously mentioned, the dimensionality of the empirical effects of an aspect is $I * J = 4 * 5 = 20$. Each dimension represents the conditional probability of an overall rating given a sentiment index detected on the aspect. We compute the features for each aspect of each hotel and show a snippet of them in Figure 5.4. The features in the first two rows are for two attractive aspects of a hotel: gym and shuttle service. The waves of the two features have relatively low values when the sentiments are not positive and have sharp peaks at $p(r = 4|s = 4)$ and $p(r = 5|s = 4)$. This indicates that customers are prone to give high ratings when the performance of those aspects is sufficient, but are insensitive to them when their performance is insufficient. The two features in the middle row are for two must-be aspects of the same hotel: check-in/out and room quietness. The waves are opposite to that of the attractive aspects. This indicates that customers can be easily dissatisfied when the performance of the aspects is insufficient but less possible to be satisfied when the performance is sufficient. The features in the third row are for two one-dimensional attributes: staff and cleanliness. Those waves have peaks at both ends, reflecting the linear relation between the performance of those aspects and the overall customer satisfaction.

5.5.3 Interactive effect

In this research, the Tensorflow package [111] is used to implement the proposed neural network to derive the interactive effects. As previously mentioned in Section 3, there are several hyper parameters of the neural network that need to be determined by grid search: the dimensionality of the interactive effect features, the number of hidden layers, and the number of activations in each hidden layer. In the grid search, the dimensionality of the interactive effect features is empirically set in the interval $[5, 20]$ with a step size of 2; the number of layers in the interval $[2, 10]$ with a step size of 1; the activation size in the interval $[10, 30]$ with a step size of 2;. The grid search tries all possible parameter combinations and finds the neural network that has the best performance when the dimensionality of the features is 9, the number of hidden layers is 3, the first hidden layer size is 9 (the same as the dimensionality of the features), the second hidden layer size is 20, and the third hidden layer size is 12.

The interactive effect features of sentiments associated with each attribute are visualised by the Hinton diagram [112]. There are 4 rows in the diagram, each row for a sentiment indicator. The white blocks in the diagram represent positive values, and the black blocks negative values. The areas of those blocks indicate the magnitude of the

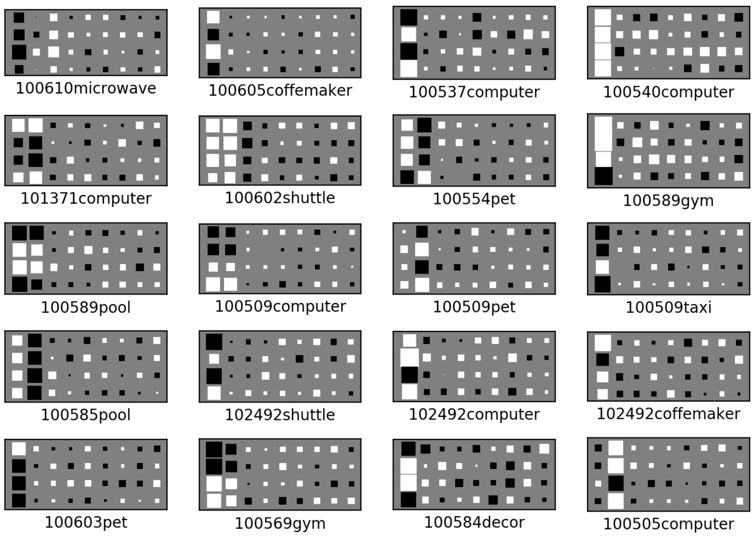


Figure 5.5: A snippet of the interactive effects of attractive attributes (the number below each diagram is the hotel ID used during the experiment)

values. We find that the features of attractive attributes have appearance distinctively different from that of the must-be and the one-dimensional attributes. A snippet of the results is shown in Figure 5.5 and Figure 5.6. In these figures, the features of attractive attributes are usually preoccupied with tiny blocks after the second column, while in the features of non-attractive attributes relatively large blocks appear across the whole matrix.

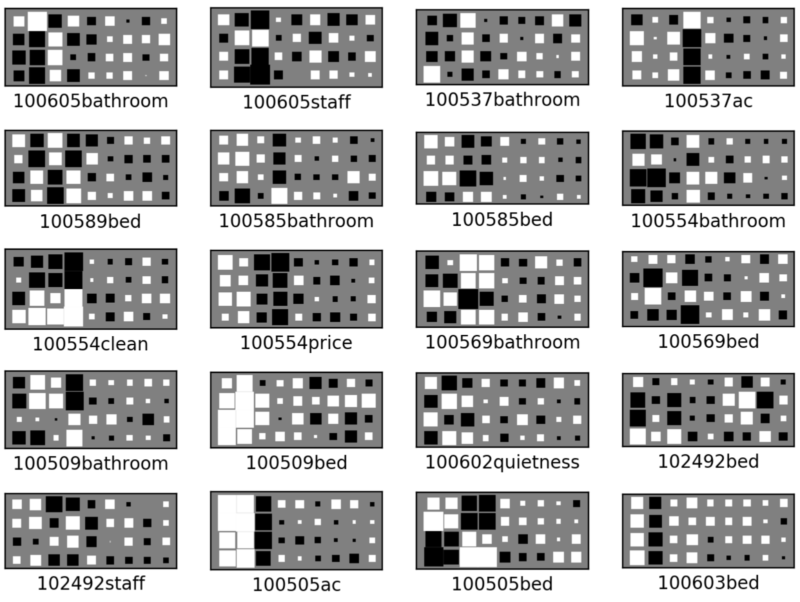


Figure 5.6: A snippet of the interactive effects of non-attractive attributes

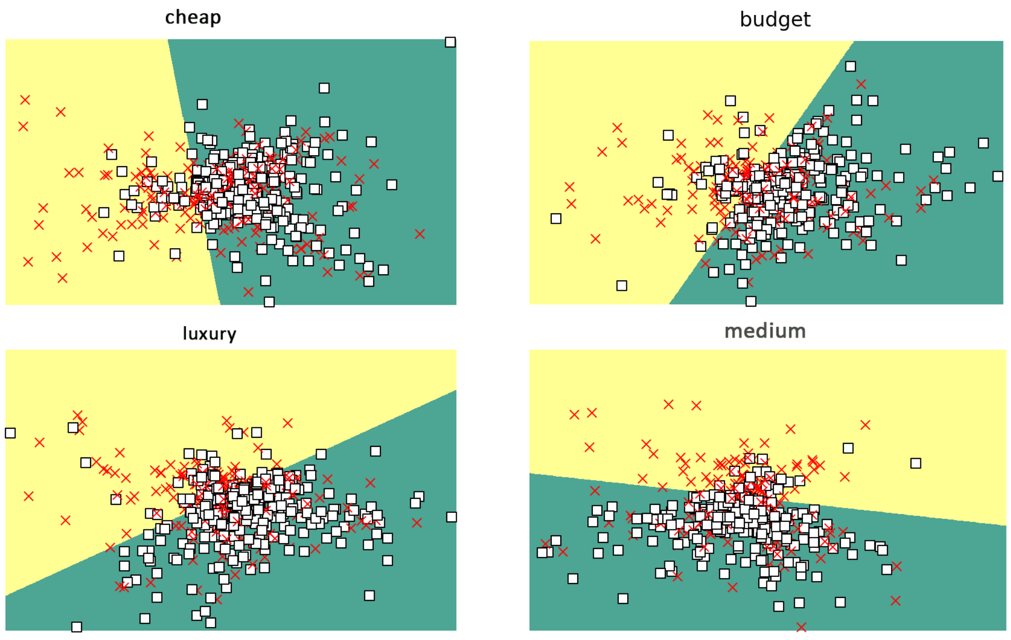


Figure 5.7: The results of K-means clustering with the interactive effect features

The red cross marks represent attractive attributes, and the white squares non-attractive attributes.

To further demonstrate the discriminatory power of the features, K-means clustering is performed on the features. The number of clusters is set to 2, one cluster represents attractive attributes, and the other non-attractive attributes. In the results, the clustering purities for the cheap, budget, medium, and luxury categories are 0.63, 0.62, 0.75, and 0.68, respectively. We visualise the clustering results on 20 hotels randomly chosen from each hotel category in Figure 5.7. As the 4 plots indicate, the majority of the attractive attributes fall in the cluster with yellow background, and the majority of the non-attractive aspects in the cluster with green background. It is noteworthy that the clustering quality on the medium category data is the best. This is partly because the number of favourable and unfavourable reviews in this category is relatively balanced, providing evenly distributed information about the attractive attributes for the neural network.

5.5.4 The attractive attribute classifiers

Each detected aspect of each hotel is treated as a data instance, and the following 3 features are used to train 3 separate SVM classifiers: the empirical effect features, the interactive effect features, and the concatenation of the two types of features. We use the 5-fold cross validation to evaluate the classifiers. We also evaluate the PRCA [90] and CIT [89] methods on the data and compare their performance with the classifiers. The results are shown in Table 5.6.

Firstly, the results show that the average precision of the 3 classifiers is 79.3% and the average recall is 73.6%. This can be interpreted as that around 80% of the attractive attributes classified by the classifiers are actually attractive in the ground truth and the classifiers can retrieve more than 70% of the attractive attributes in the ground truth. Among the 3 classifiers, the one trained with the concatenation features is the best, followed by the one trained with the interactive effect features. The classifier trained with the empirical effect features is the worst. It is noteworthy that the concatenation features and the interactive effect features have more clear advantages over the empirical effect features on the luxury category. One possible reason is that the favourable reviews dominantly outnumber the unfavourable reviews in this category, therefore, information concerning negative opinions is severely inadequate. The empirical effect features are more sensitive to such lack of information.

Secondly, the performance of the classifiers is over 10% higher than that of the statistical models in terms of the F1 score. According to the analysis in Section 1, the low performance of the statistical models is partly caused by the noisy nature of the derived attribute-level sentiments and their ineffectiveness in modelling the non-linearity of the data.

In the experiment, the average rating of the hotels on which attractive attributes are detected is higher than that of the hotels on which no attractive attribute is detected across the 4 hotel categories. The averag rating of the hotels with attractive attributes is 0.27 higher than that of the hotels without the attributes in the cheap category, 0.31 higher in the budget category, 0.42 higher in the the medium category and 0.21 higher in the luxury category. The highest difference margin occurs in the medium category. This may indicate that there exists very intense competition in this category and the clients of those hotels are more sensitive to the attractive attributes than the clients of other hotel categories.

Table 5.6: Performance of the proposed classifiers and the PRAC/CIT based methods. Prec stands for 'Precision', Rec stands for Recall'.

	SVM+ the empirical effects			SVM+ the interactive effects			SVM+the concatenation features			PRCA [90]			CIT [89]		
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
Luxury	0.76	0.69	0.72	0.86	0.80	0.83	0.88	0.85	0.86	0.65	0.55	0.60	0.65	0.61	0.63
Medium	0.71	0.51	0.59	0.76	0.59	0.66	0.80	0.61	0.69	0.46	0.62	0.53	0.70	0.50	0.58
Budget	0.69	0.89	0.78	0.73	0.91	0.81	0.77	0.91	0.83	0.59	0.74	0.66	0.67	0.59	0.63
Cheap	0.82	0.65	0.73	0.85	0.69	0.76	0.87	0.74	0.80	0.68	0.63	0.65	0.79	0.62	0.69
Average	0.75	0.68	0.70	0.80	0.75	0.76	0.83	0.78	0.79	0.60	0.64	0.61	0.66	0.61	0.63

5.6 Conclusion

This chapter aims to develop predictive classifiers based on machine learning techniques that analyse on-line review data to identify the attractive attributes of a product or service. A critical problem in training such classifiers is deriving discriminative features that can reflect the non-linear relations between attribute-level opinions and overall ratings. Two types of features, the empirical effect feature that is the empirical probabilities of ratings conditioning on various attribute-level opinions, and the interactive effect feature that encodes the interactions among attribute-level opinions in deciding the overall ratings, are used together to train the classifiers. Compared with the existing methods based on statistical models, the proposed classifiers not only encode a much higher degree of non-linearity in customer opinions, but also make use of the global statistics of the training data. The proposed classifiers are evaluated on a hotel review dataset crawled from TripAdvisor. The experiment results indicate that the classifiers reach a precision of 79.3% and outperform the existing statistical models by a margin of over 10%.

One drawback with the classifiers is that the information encoded by the empirical effect feature and the interactive effect feature may overlap to some degree. The information redundancy can result in severe overfitting. In the future, we will improve the attribute representations and investigate the possibilities of applying deep learning techniques to build more reliable classifiers.

Chapter 6

Conclusion

This chapter summarises the main contributions of this dissertation, and conclude the dissertation with future research directions.

6.1 The main contribution

This dissertation discusses two fundamental problems in sentence-level opinion classification on review data: sentence representations and the limited availability of training data.

To address the former problem, this dissertation proposes a weighted average approach to aggregate word vectors to represent short text. The proposed approach assumes there exist some very important abstract keywords, and the weight of each word in a document is determined by the semantic similarity between the word and the abstract keywords. The abstract keywords are integrated into the cost function of the learning model under study, and learned along with other model parameters during the training process. Experiment results show that this approach has better performance than the deep learning models using complex structures, such as CNN and RNN, but much less computationally expensive.

To address the latter problem, this dissertation proposes the Averaged Logits model, that uses ratings, instead of sentence-level sentiment labels, to train the sentence-level sentiment classifiers. Since ratings are prevalently available in review data, no manual labor for data annotation is involved. Evaluation results show that the performance of the proposed approach is close to the traditional learning models using sentence-level labels as the supervision signal .

Also, based on the results of the opinion classification process , this dissertation proposes a machine learning based approach to identify the attractive aspects of a product or service. Evaluation results show that the proposed approach reaches a precision of 79.3% and outperforms the existing statistical model based methods by a margin of over 10%.

6.2 Contributions the community of Knowledge Science

The tasks related review summarisation discussed throughout this dissertation are essentially a data-knowledge transformation problem, that is an important research topic in the community of Knowledge Science. To be specific, the dissertation contributes to the development of Knowledge Science in at least the following aspects :

- Data representation. Chapter 3 proposes the Abstract Keywords model to derive sentence embedding representations in the opinion classification process.
- Data management. Chapter 4 proposes the weakly-supervised Averaged Logits model that requires no manual labelling efforts to train sentence-level classifiers. The proposed model significantly improves the effectiveness of the data management process.
- Processing information into Knowledge. The results of the opinion classification are random, loosely organised information. Chapter 5 proposes a machine learning approach to process the opinion information to discover the selling points of products or services, that is a very useful form of knowledge for consumers' decision-making processes.

6.3 Directions for future work

The transformation from knowledge to wisdom is not covered in this dissertation. In the context of review summarisation, wisdom can be translated as 'personalised', that requires the system to present different review summaries for different consumers based on the personal preferences of the consumers. This dissertation mainly aims to address problems that are generally applicable to all the stakeholders as a whole, and no personalised data, information or knowledge is taken into account. In the future, the author will explore for new methods to incorporate wisdom into the review summarisation process.

Bibliography

[1] J. Yu, Z.-J. Zha, M. Wang, and T.-S. Chua, "Aspect ranking: identifying important product aspects from online consumer reviews," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 1496–1505, Association for Computational Linguistics, 2011.

[2] L. Zhuang, F. Jing, and X.-Y. Zhu, "Movie review mining and summarization," in *Proceedings of the 15th ACM international conference on Information and knowledge management*, pp. 43–50, ACM, 2006.

[3] H. Wang, Y. Lu, and C. Zhai, "Latent aspect rating analysis on review text data: a rating regression approach," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 783–792, ACM, 2010.

[4] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 168–177, ACM, 2004.

[5] F. Sebastiani, "Machine learning in automated text categorization," *ACM computing surveys (CSUR)*, vol. 34, no. 1, pp. 1–47, 2002.

[6] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information processing & management*, vol. 24, no. 5, pp. 513–523, 1988.

[7] J. A. Nelder and R. J. Baker, "Generalized linear models," *Encyclopedia of statistical sciences*, vol. 4, 2004.

[8] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: a simple and general method for semi-supervised learning," in *Proceedings of the 48th annual meeting of*

the association for computational linguistics, pp. 384–394, Association for Computational Linguistics, 2010.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.

[10] J. Mikulić, “The kano model–a review of its application in marketing research from 1984 to 2006,” in *Proceedings of the 1st International Conference marketing theory challenges in transitional societies*, pp. 87–96, 2007.

[11] Q. Xu, R. J. Jiao, X. Yang, M. Helander, H. M. Khalid, and A. Opperud, “An analytical kano model for customer need analysis,” *Design Studies*, vol. 30, no. 1, pp. 87–110, 2009.

[12] J. Mikulić and D. Prebežac, “A critical review of techniques for classifying quality attributes in the kano model,” *Managing Service Quality: An International Journal*, vol. 21, no. 1, pp. 46–66, 2011.

[13] N. Kano, “Attractive quality and must-be quality,” *J. Jpn. Soc. Quality Control*, vol. 14, pp. 39–48, 1984.

[14] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, “Supervised machine learning: A review of classification techniques,” *Emerging artificial intelligence applications in computer engineering*, vol. 160, pp. 3–24, 2007.

[15] W. J. Wilbur and K. Sirotkin, “The automatic identification of stop words,” *Journal of information science*, vol. 18, no. 1, pp. 45–55, 1992.

[16] C. D. Manning, C. D. Manning, and H. Schtze, *Foundations of statistical natural language processing.* MIT press, 1999.

[17] T. Glasmachers, “Limits of end-to-end learning,” *arXiv preprint arXiv:1704.08305*, 2017.

[18] D. M. Blei, “Probabilistic topic models,” *Communications of the ACM*, vol. 55, no. 4, pp. 77–84, 2012.

[19] G. Song, Y. Ye, X. Du, X. Huang, and S. Bie, "Short text classification: A survey," *Journal of Multimedia*, vol. 9, no. 5, p. 635, 2014.

[20] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[21] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *Journal of Big Data*, vol. 2, no. 1, p. 1, 2015.

[22] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[23] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *International Conference on Machine Learning*, pp. 2342–2350, 2015.

[24] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[25] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[26] P. E. Gill, W. Murray, and M. H. Wright, "Practical optimization," 1981.

[27] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[28] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.

[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[30] Y. Dauphin, H. De Vries, J. Chung, and Y. Bengio, "Rmsprop and equilibrated adaptive learning rates for non-convex optimization (2015). arxiv preprint," *arXiv preprint arXiv:1502.04390*.

[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[32] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 675–678, ACM, 2014.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," Feb. 2015.

[34] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," p. 9.

[35] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[36] C. De Boom, S. Van Canneyt, T. Demeester, and B. Dhoedt, "Representation learning for very short texts using weighted word embedding aggregation," *Pattern Recognition Letters*, vol. 80, pp. 150–156, 2016.

[37] M. Iyyer, V. Manjunatha, J. Boyd-Graber, and H. Daum III, "Deep unordered composition rivals syntactic methods for text classification," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, vol. 1, pp. 1681–1691, 2015.

[38] Y. Kim, "Convolutional Neural Networks for Sentence Classification," *arXiv:1408.5882 [cs]*, Aug. 2014. arXiv: 1408.5882.

[39] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," *arXiv preprint arXiv:1404.2188*, 2014.

[40] K. S. Tai, R. Socher, and C. D. Manning, "Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks," *arXiv:1503.00075 [cs]*, Feb. 2015. arXiv: 1503.00075.

[41] T. Mikolov, M. Karafit, L. Burget, J. ernock, and S. Khudanpur, "Recurrent neural network based language model," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[42] D. Tang, B. Qin, and T. Liu, "Document Modeling with Gated Recurrent Neural Network for Sentiment Classification," pp. 1422–1432, Association for Computational Linguistics, 2015.

[43] Y. Wang, M. Huang, and L. Zhao, "Attention-based lstm for aspect-level sentiment classification," in *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 606–615, 2016.

[44] T. Chen, R. Xu, Y. He, and X. Wang, "Improving sentiment analysis via sentence type classification using BiLSTM-CRF and CNN," *Expert Systems with Applications*, vol. 72, pp. 221–230, 2017.

[45] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1480–1489, 2016.

[46] D. Tang, B. Qin, and T. Liu, "Aspect level sentiment classification with deep memory network," *arXiv preprint arXiv:1605.08900*, 2016.

[47] Y. Xiao and K. Cho, "Efficient character-level document classification by combining convolution and recurrent layers," *arXiv preprint arXiv:1602.00367*, 2016.

[48] X. Wang, W. Jiang, and Z. Luo, "Combination of Convolutional and Recurrent Neural Network for Sentiment Analysis of Short Texts," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, (Osaka, Japan), pp. 2428–2437, The COLING 2016 Organizing Committee, Dec. 2016.

[49] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International Conference on Machine Learning*, pp. 1188–1196, 2014.

[50] T. Kenter, A. Borisov, and M. de Rijke, "Siamese CBOW: Optimizing Word Embeddings for Sentence Representations," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Berlin, Germany), pp. 941–951, Association for Computational Linguistics, Aug. 2016.

[51] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler, "Skip-thought vectors," in *Advances in neural information processing systems*, pp. 3294–3302, 2015.

[52] B. Pang and L. Lee, "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales," in *Proceedings of the 43rd annual meeting on association for computational linguistics*, pp. 115–124, Association for Computational Linguistics, 2005.

[53] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, (Seattle, Washington, USA), pp. 1631–1642, Association for Computational Linguistics, Oct. 2013.

[54] B. Pang and L. Lee, "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts," in *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, p. 271, Association for Computational Linguistics, 2004.

[55] X. Li and D. Roth, "Learning question classifiers," in *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pp. 1–7, Association for Computational Linguistics, 2002.

[56] J. Wiebe, T. Wilson, and C. Cardie, "Annotating expressions of opinions and emotions in language," *Language resources and evaluation*, vol. 39, no. 2-3, pp. 165–210, 2005.

[57] B. Pang, L. Lee, and S. Vaithyanathan, “Thumbs up? Sentiment Classification using Machine Learning Techniques,” *arXiv:cs/0205070*, May 2002. arXiv: cs/0205070.

[58] M. Gamon, “Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis,” p. 7.

[59] T. Mullen and N. Collier, “Sentiment Analysis using Support Vector Machines with Diverse Information Sources,” in *Proceedings of EMNLP 2004* (D. Lin and D. Wu, eds.), (Barcelona, Spain), pp. 412–418, Association for Computational Linguistics, July 2004.

[60] H. Cui, “Comparative Experiments on Sentiment Classification for Online Product Reviews,” p. 6.

[61] J. Martineau and T. Finin, “Delta TFIDF: An Improved Feature Space for Sentiment Analysis.,” *Icwsm*, vol. 9, p. 106, 2009.

[62] J. Lafferty, A. McCallum, and F. C. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” 2001.

[63] E. Kouloumpis, T. Wilson, and J. D. Moore, “Twitter sentiment analysis: The good the bad and the omg!,” *Icwsm*, vol. 11, no. 538-541, p. 164, 2011.

[64] L. Qu, R. Gemulla, and G. Weikum, “A Weakly Supervised Model for Sentence-Level Semantic Orientation Analysis with Multiple Experts,” in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, (Jeju Island, Korea), pp. 149–159, Association for Computational Linguistics, July 2012.

[65] B. Yang and C. Cardie, “Context-aware Learning for Sentence-level Sentiment Analysis with Posterior Regularization,” in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Baltimore, Maryland), pp. 325–335, Association for Computational Linguistics, June 2014.

[66] O. Tckstrm and R. McDonald, “Discovering Fine-Grained Sentiment with Latent Variable Structured Prediction Models,” in *Advances in Information Retrieval*

(P. Clough, C. Foley, C. Gurrin, G. J. F. Jones, W. Kraaij, H. Lee, and V. Mudoch, eds.), vol. 6611, pp. 368–374, Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.

[67] F. Wu, J. Zhang, Z. Yuan, S. Wu, Y. Huang, and J. Yan, "Sentence-level Sentiment Classification with Weak Supervision," pp. 973–976, ACM Press, 2017.

[68] C.-Y. Lu, S.-H. Lin, J.-C. Liu, S. Cruz-Lara, and J.-S. Hong, "Automatic event-level textual emotion sensing using mutual action histogram between entities," *Expert Systems with Applications*, vol. 37, pp. 1643–1653, Mar. 2010.

[69] P. D. Turney, "Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews," *arXiv:cs/0212032*, Dec. 2002. arXiv: cs/0212032.

[70] X. Ding, B. Liu, and P. S. Yu, "A Holistic Lexicon-based Approach to Opinion Mining," in *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, (New York, NY, USA), pp. 231–240, ACM, 2008.

[71] F. Xianghua, L. Guo, G. Yanyan, and W. Zhiqiang, "Multi-aspect sentiment analysis for Chinese online social reviews based on topic modeling and HowNet lexicon," *Knowledge-Based Systems*, vol. 37, pp. 186–195, 2013.

[72] F. Li, M. Huang, and X. Zhu, "Sentiment Analysis with Global Topics and Local Dependency.," in *AAAI*, vol. 10, pp. 1371–1376, 2010.

[73] C. Lin and Y. He, "Joint sentiment/topic model for sentiment analysis," p. 10.

[74] A. Kennedy and D. Inkpen, "Sentiment classification of movie reviews using contextual valence shifters," *Computational intelligence*, vol. 22, no. 2, pp. 110–125, 2006.

[75] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, "Lexicon-based methods for sentiment analysis," *Computational linguistics*, vol. 37, no. 2, pp. 267–307, 2011.

[76] T. Wilson, J. Wiebe, and P. Hoffmann, "Recognizing Contextual Polarity: An Exploration of Features for Phrase-Level Sentiment Analysis," *Computational Linguistics*, vol. 35, pp. 399–433, Sept. 2009.

[77] V. Ng, S. Dasgupta, and S. M. N. Arifin, "Examining the Role of Linguistic Knowledge Sources in the Automatic Identification and Classification of Reviews," in *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, (Sydney, Australia), pp. 611–618, Association for Computational Linguistics, July 2006.

[78] Y. Choi and C. Cardie, "Learning with Compositional Semantics as Structural Inference for Subsentential Sentiment Analysis," in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, (Honolulu, Hawaii), pp. 793–801, Association for Computational Linguistics, Oct. 2008.

[79] S.-M. Kim and E. Hovy, "Determining the sentiment of opinions," in *Proceedings of the 20th international conference on Computational Linguistics*, p. 1367, Association for Computational Linguistics, 2004.

[80] A. Valitutti, C. Strapparava, and O. Stock, "Developing affective lexical resources.," *PsychNology Journal*, vol. 2, no. 1, pp. 61–83, 2004.

[81] S. Blair-Goldensohn, K. Hannan, R. McDonald, T. Neylon, G. A. Reis, and J. Reynar, "Building a sentiment summarizer for local service reviews," in *WWW workshop on NLP in the information explosion era*, vol. 14, pp. 339–348, 2008.

[82] M. Ganapathibhotla and B. Liu, "Mining opinions in comparative sentences," in *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pp. 241–248, Association for Computational Linguistics, 2008.

[83] J. Mikuli and D. Prebeac, "A critical review of techniques for classifying quality attributes in the Kano model," *Managing Service Quality: An International Journal*, vol. 21, pp. 46–66, Jan. 2011.

[84] L.-F. Chen, "A novel approach to regression analysis for the classification of quality attributes in the Kano model: an empirical test in the food and beverage industry," *Omega*, vol. 40, no. 5, pp. 651–659, 2012.

[85] G. Tontini and A. Silveira, "Identification of satisfaction attributes using competitive analysis of the improvement gap," *International Journal of Operations & Production Management*, vol. 27, no. 5, pp. 482–500, 2007.

[86] K. C. Tan and X.-X. Shen, "Integrating Kano's model in the planning matrix of quality function deployment," *Total quality management*, vol. 11, no. 8, pp. 1141–1151, 2000.

[87] K. Matzler, F. Bailom, H. H. Hinterhuber, B. Renzl, and J. Pichler, "The asymmetric relationship between attribute-level performance and overall customer satisfaction: a reconsideration of the importanceperformance analysis," *Industrial marketing management*, vol. 33, no. 4, pp. 271–277, 2004.

[88] X. Xu and Y. Li, "The antecedents of customer satisfaction and dissatisfaction toward various types of hotels: A text mining approach," *International Journal of Hospitality Management*, vol. 55, pp. 57–69, May 2016.

[89] W. Lu and S. Stepchenkova, "Ecotourism experiences reported online: Classification of satisfaction attributes," *Tourism Management*, vol. 33, pp. 702–712, June 2012.

[90] G. Tontini, G. d. S. Bento, T. C. Milbratz, B. K. Volles, and D. Ferrari, "Exploring the nonlinear impact of critical incidents on customers general evaluation of hospitality services," *International Journal of Hospitality Management*, vol. 66, pp. 106–116, Sept. 2017.

[91] Y. Zhang and S. T. Cole, "Dimensions of lodging guest satisfaction among guests with mobility challenges: A mixed-method analysis of web-based texts," *Tourism Management*, vol. 53, pp. 13–27, Apr. 2016.

[92] M. Pontiki, D. Galanis, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, M. AL-Smadi, M. Al-Ayyoub, Y. Zhao, B. Qin, and O. De Clercq, "SemEval-2016 task 5: Aspect based sentiment analysis," in *ProWorkshop on Semantic Evaluation (SemEval-2016)*, pp. 19–30, Association for Computational Linguistics, 2016.

[93] T. Brychcn, M. Konkol, and J. Steinberger, "UWB: Machine Learning Approach to Aspect-Based Sentiment Analysis," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, (Dublin, Ireland), pp. 817–822, Association for Computational Linguistics and Dublin City University, Aug. 2014.

[94] S. Kiritchenko, X. Zhu, C. Cherry, and S. Mohammad, “NRC-Canada-2014: Detecting aspects and sentiment in customer reviews,” in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 437–442, 2014.

[95] D. Bespalov, B. Bai, Y. Qi, and A. Shokoufandeh, “Sentiment classification based on supervised latent n-gram analysis,” in *Proceedings of the 20th ACM international conference on Information and knowledge management*, pp. 375–382, ACM, 2011.

[96] C. Long, J. Zhang, and X. Zhut, “A review selection approach for accurate feature rating estimation,” in *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pp. 766–774, Association for Computational Linguistics, 2010.

[97] C. Toprak, N. Jakob, and I. Gurevych, “Sentence and expression level annotation of opinions in user-generated discourse,” in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 575–584, Association for Computational Linguistics, 2010.

[98] H. Guo, H. Zhu, Z. Guo, X. Zhang, and Z. Su, “Product feature categorization with multilevel latent semantic association,” in *Proceedings of the 18th ACM conference on Information and knowledge management*, pp. 1087–1096, ACM, 2009.

[99] J. Lilleberg, Y. Zhu, and Y. Zhang, “Support vector machines and word2vec for text classification with semantic features,” in *Cognitive Informatics & Cognitive Computing (ICCI* CC), 2015 IEEE 14th International Conference on*, pp. 136–140, IEEE, 2015.

[100] F. Li, C. Han, M. Huang, X. Zhu, Y.-J. Xia, S. Zhang, and H. Yu, “Structure-aware review mining and summarization,” in *Proceedings of the 23rd international conference on computational linguistics*, pp. 653–661, Association for Computational Linguistics, 2010.

[101] D. Marcheggiani, O. Tckstrm, A. Esuli, and F. Sebastiani, “Hierarchical Multi-label Conditional Random Fields for Aspect-Oriented Opinion Mining.,” in *ECIR*, pp. 273–285, Springer, 2014.

[102] Y. Jo and A. H. Oh, "Aspect and sentiment unification model for online review analysis," in *Proceedings of the fourth ACM international conference on Web search and data mining*, pp. 815–824, ACM, 2011.

[103] C. Lin and Y. He, "Joint sentiment/topic model for sentiment analysis," in *Proceedings of the 18th ACM conference on Information and knowledge management*, pp. 375–384, ACM, 2009.

[104] A. Mukherjee and B. Liu, "Aspect extraction through semi-supervised modeling," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pp. 339–348, Association for Computational Linguistics, 2012.

[105] Q. Mei, X. Ling, M. Wondra, H. Su, and C. Zhai, "Topic sentiment mixture: modeling facets and opinions in weblogs," in *Proceedings of the 16th international conference on World Wide Web*, pp. 171–180, ACM, 2007.

[106] R. D. Brandt, "A procedure for identifying value-enhancing service components using customer satisfaction survey data," *Add Value to Your Service, Chicago: American Marketing Association*, pp. 61–65, 1987.

[107] D. D. Gremler, "The Critical Incident Technique in Service Research," *Journal of Service Research*, vol. 7, pp. 65–89, Aug. 2004.

[108] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in neural information processing systems*, pp. 396–404, 1990.

[109] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, "The stanford corenlp natural language processing toolkit.," in *ACL (System Demonstrations)*, pp. 55–60, 2014.

[110] Z. Zhai, B. Liu, H. Xu, and P. Jia, "Grouping product features using semi-supervised learning with soft-constraints," in *Proceedings of the 23rd International Conference on Computational Linguistics*, pp. 1272–1280, Association for Computational Linguistics, 2010.

[111] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[112] F. J. Bremner, S. J. Gotts, and D. L. Denham, "Hinton diagrams: Viewing connection strengths in neural networks," *Behavior Research Methods, Instruments, & Computers*, vol. 26, pp. 215–218, June 1994.

Publications

International journals

[1] Wei Ou, Van-Nam Huynh and Songsak Sriboonchitta, Identifying Attractive Attributes from Hotel Reviews: a Machine Learning Approach, *Electronic Commerce Research and Applications*, Elsevier: published, Vol 32, pp.13-22,2018.

[2] Wei Ou, Van-Nam Huynh, Joint Aspect Discovery, Sentiment Classification, Aspect-Level Ratings and Weights Approximation for Recommender Systems by Rating Supervised Latent Topic Model, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, Fuji Technology Press: published, 22(1), pp.17-26, 2018.

International conferences

[3] Wei Ou, Van-Nam Huynh, Rating supervised latent topic model for aspect discovery and sentiment classification in on-line review mining *In: Proceedings of the 13th Modeling Decisions for Artificial Intelligence*, Springer: Refereed, pp. 151-164, 19th-21st September 2016, Sant Julia de Loria, Andorra.

[4] Wei Ou,Anh-Cuong Le, Van-Nam Huynh, Estimating Asymmetric Product Attribute Weights in Review Mining. *In: Proceedings of the International Symposium on Integrated Uncertainty in Knowledge Modelling and Decision Making*, Springer: Refereed, pp. 245-254, November 30-December 2, 2016, Da Nang, Vietnam.