Exploiting Multiple Word Embeddings and One-hot Character Vectors for Aspect-Based Sentiment Analysis

Duc-Hong Pham[a,c], Anh-Cuong Le[b,*]

[a] *Faculty of Information Technology, Electric Power University, Hanoi City, Vietnam*
hongpd@epu.edu.vn
[b] *NLP-KD Lab, Faculty of Information Technology, Ton Duc Thang University, HoChiMinh City, Vietnam*
leanhcuong@tdt.edu.vn
[c] *Faculty of Information Technology, University of Engineering and Technology, Vietnam National University, Hanoi, Vietnam*

Abstract

Representing words as real-value vectors is an effective approach in many natural language processing tasks. In addition, some studies have used lower level character-based representations. The main problem is how to exploit the advantages of different representations of input sources. Therefore, we propose a model for combining different representations of inputs, which includes two word embedding methods (i.e., Word2Vec and GloVe) and a one-hot vector for character representation. Our model is formulated in a multichannel framework using convolutional neural networks. The proposed model addresses the problem of aspect-based sentiment analysis. Our experimental results demonstrate that the proposed model can achieve state-of the-art performance in aspect category detection and aspect sentiment classification tasks.

Keywords: aspect category detection, aspect sentiment classification, convolutional neural network, one-hot character vector, word embedding.

1. Introduction

Word embeddings represent the words in a vocabulary as real-valued vectors in a multidimensional space. They are trained using a large set of unlabeled data

*Corresponding author

and formulated as real-valued vectors based on the word appearance contexts. Word embeddings can capture syntactic and semantic information without using labeled data, and thus they are very useful for many natural language processing (NLP) tasks, such as text classification, information extraction, information retrieval, question answering, sentiment analysis, and machine translation.

At a lower level of performance, some studies (e.g., dos Santos et al. [1], Zhang et al. [2]) have effectively used character representations instead of word representations. One-hot character vectors are representations of characters in the character vocabulary. A one-hot vector contains only a position with a value of one, which corresponds to the character index in the vocabulary, and zeros are in all the other positions. Therefore, the dimensionality of the one-hot character vectors equals the character vocabulary size.

Several sets of pre-trained word embedding data are now readily available on the Web, which were generated from different models and corpora, such as C&W (Collobert et al. [3]), Word2Vec (Mikolov et al. [4]), and GloVe (Pennington et al. [5]). Different embedded vectors may encode various aspects of language. In our opinion, different representations of the input may potentially offer complementary information with respect to the target (e.g., in a classification task). The main problem is how to combine different representations of input sources, including different word embeddings and one-hot character vectors into a unified model.

Sentiment analysis is a type of data mining that measures the inclination of people's opinions through natural language processing (NLP), computational linguistics and text analysis, which are used to extract and analyze subjective information from social networks, Twitter, forums, blogs, etc. Some problems have been solved, such as analysis of discussions in Twitter (Alsinet et al. [6]), compute text polarity by leveraging on information extracted from domain-specific models (Dragoni et al. [7]). Aspect-based sentiment analysis is a special type of sentiment analysis, its task is to identify the different aspects of entities in textual reviews and to determine the sentiments associated with these aspects. Some previous studies in this area employed traditional approaches

based on lexical information (e.g., n-grams as features) and classification machine learning methods (e.g., support vector machines), such as those by Ganu et al. [8], Wagner et al. [9], and Kiritchenko et al. [10]. Recently, many studies have used word embeddings as inputs for neural network models, such as aspect extraction (Poria et al. [11]), aspect-level sentiment classification (Tang et al. [12], Wang et al. [13]), and determining aspect ratings and aspect weights (Pham et al. [14]). However, most of these studies only used one set of the learned word embeddings and they ignored the character-level representations.

In recent years, it has been shown that convolutional neural networks (CNNs) are highly effective and they have achieved state-of-the-art results for some NLP tasks, such as sentence modeling (Kalchbrenner et al. [15]), sentence classification (Lakshmana et al. [16]), and learning semantic representations for web search (Shen et al. [17]). CNNs provide an efficient mechanism for aggregating information at a higher level of abstraction. Some studies have used MCNNs for sentence classification. For example, Kim et al. [18] proposed a multichannel representation architecture based on variable-size filters. However, their multichannel model operates only with a single version of the pre-trained embeddings (i.e., pre-trained Word2Vec embeddings), where one is kept stable and the other is fine tuned by back-propagation. Yin et al. [19] developed this method further by incorporating diverse embedding versions, but their model requires input word embeddings with the same dimensions. Zhang et al. [20] improved this model for sentence classification by treating different word embeddings as distinct groups and applied CNNs independently to each, before concatenating all of the vectors obtained in the classification layer.

In this study we address the problem of how to integrate different information sources to form a unified representation for the task of aspect-based sentiment analysis. We focus on the three representations of inputs as different information sources needed to be integrated, which includes character based representations (e.g. one-hot character vectors), the Word2Vec representations from (Mikolov et al. [4]), and the GloVe representations from the GloVe model (Pennington et al. [5]). We will propose a joint model called Multichannel framework using

Convolutional Neural Networks (called MCNN in brief) for this task, which is inspired from the work in Zhang et al. [20]. In this model we will learn the shared representation (i.e. the unified representation) from the three different input sources, in which we use CNN models for generating the individual representations for each of the initial input sources. It is worth to emphasize that the individual representations are learned in a joint model in which each affects the others in generating the unified representation.

Three representations of inputs of MCNN model have different roles. Two representations of Word2Vec and GloVe capture syntactic and semantic information during word level. They are the learned word embeddings (i.e. word vectors) from different methods and based on different training data sets. Technically, Word2Vec method is a "artificial neural network predictive" model with two skip-gram and cbow (continuous bag of words) architectures, whereas GloVe method is a "count-based" model. The skip-gram model uses the current word to predict the context words. The CBOW model is the opposite of the skip-gram model, the current word is predicted from the given context words. GloVe model is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. For training data, word embeddings from Word2Vec were trained on part of Google News dataset with the size of the vocabulary is 100 billion, and word embeddings from GloVe were trained on web data with the size of the vocabulary is 1.9M.

Both Word2Vec and GloVe are still treat words as atomic units and ignore morphological and shape similarities between different words. Using the character based representation in a CNN can capture morphological and shape information from words must take into consideration all characters of the word and select which features are more important for the task at hand (dos Santos et al. [1]). Therefore, we believe that the combination of this representation and two representations of Word2Vec and GloVe in the framework of CNN based multichannel can capture more information than using single channel (i.e. one source of word embeddings) about aspects of semantics.

In the experiments, we used about 52,574 reviews from the domain of restaurant products[1]. We evaluated the effectiveness of our MCNN model by applying it to two aspect-based sentiment analysis tasks, which comprised aspect category detection and aspect sentiment classification. The experimental results showed that our MCNN model performed better than the methods proposed in previous studies such as CNN models (Kim et al. [18]) and the CharSCNN model (dos Santos et al. [1]).

2. Aspect-based Sentiment Analysis Formulation

Aspect-based sentiment analysis involves predicting the aspects of a predefined object (e.g., the *price, food, service, ambience, anecdotes* aspects of the object "restaurant") and the associated sentiments (e.g., *positive, negative*) assigned to each aspect in a certain context (we assume that the context is a sentence). According to [21, 22], we formulate the aspect category detection and aspect sentiment classification problems as follows.

Aspect category detection: Given a set of k predefined aspect categories for an entity A, denoted by $\{\mathrm{A}_1,\mathrm{A}_2,...,\mathrm{A}_k\}$, for an input sentence d, we need to predict a binary label vector $a_d \in \mathrm{R}^{\mathrm{k}}$. In particular, $a_{di} = 1$ means that sentence d contains the aspect category A_i and $a_{di} = 0$ means that it does not contain the aspect category A_i.

Aspect sentiment classification: Suppose that the aspect task is that defined above and we are given a set of l predefined aspect sentiment labels $O = \{\mathrm{O}_1,\mathrm{O}_2,...,\mathrm{O}_l\}$ (e.g., *positive, negative, neutral*), for the input sentence d, which is determined by an aspect category label. We need to predict a binary label vector $o_d \in \mathrm{R}^{\mathrm{l}}$ for sentiment classification. In particular, $o_{di} = 1$ means that sentence d contains the aspect sentiment O_i and $o_{di} = 0$ means that it does not contain the aspect sentiment O_i.

For each task (i.e., aspect category detection or aspect sentiment classifi-

[1] http://spidr-ursa.rutgers.edu/datasets/

cation), we are given a set of sentences extracted from a collection of textual reviews of some entities in a particular domain (e.g., restaurant), where each sentence is assigned with the correct label (i.e., aspect category or aspect sentiment class). The problem is how to build a model based on this training data in order to predict the most appropriate label for a new input sentence. Similar to [23], for simplicity, we only consider a single class problem, which means that each training sentence is assigned with only one label.

3. Proposed MCNN-based Model

In this section, we first explain the convolution component, which contains two layers comprising the convolution layer and pooling layer. We then use this convolution component to design an MCNN model for aspect-based sentiment analysis.

3.1. Convolution Component

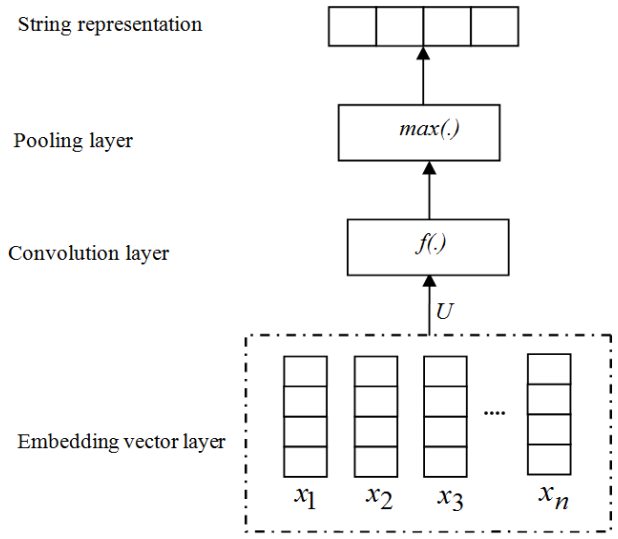

Figure 1: General convolution component.

A traditional convolution neural network contains one convolution layer and one pooling layer. The general architecture of a convolution component is shown in Figure 1. Let $e_1, e_2, ..., e_n$ be a sequence of features for an input string d (in this case, we use a sequence of words for an input sentence) and $x_1, x_2, ..., x_n$ are the corresponding embedding vectors for this sequence. The two layers of the general convolution component are presented as follows.

Convolution layer This layer receives $x_1, x_2, ..., x_n$ as inputs and uses the convolution operation to obtain a new vector sequence $y_1^1, y_2^1, ..., y_n^1$ according to the following equation:

$$y_i^1 = f(U.x_{i:i+h-1} + b), \tag{1}$$

where $x_{i:i+h-1}$ denotes the concatenation of the embedding vectors $x_i, x_{i+1}, ..., x_{i+h-1}$, h is the window size for the embedding vectors that need to be combined, $f(.)$ is an element-wise activation function such as a nonlinear function (we use $f(t) = \tanh(t) = \frac{e^t - e^{-t}}{e^t + e^{-t}}$), $U \in \mathrm{R}^{\mathrm{C} \times \mathrm{m}}$ and $b \in \mathrm{R}^{\mathrm{C}}$ are component parameters learned during the training stage, and C is the output dimension.

Pooling layer We apply the max pooling operation [24] to mix the features from the convolution layer into one vector with a fixed dimension:

$$y^2 = [max(y_{i1}^1), max(y_{i2}^1), ..., max(y_{iC}^1)], \tag{2}$$

where y_{ij}^1 denotes the j-th dimension of y_i^1 and $y^2 \in R^C$ is the output vector of the convolution component.

3.2. MCNN for Aspect-based Sentiment Analysis

Next, we present our proposed MCNN model for aspect-based sentiment analysis. In this section, we illustrate this model based on the aspect category detection problem, but the method is the same for the aspect sentiment classification problem. The architecture of the model is shown in Figure 2, which includes three CNN channels, where the first channel uses pre-trained Word2Vec word embeddings, the second channel uses pre-trained Glove word embeddings, and the third channel uses one-hot character vectors as inputs.

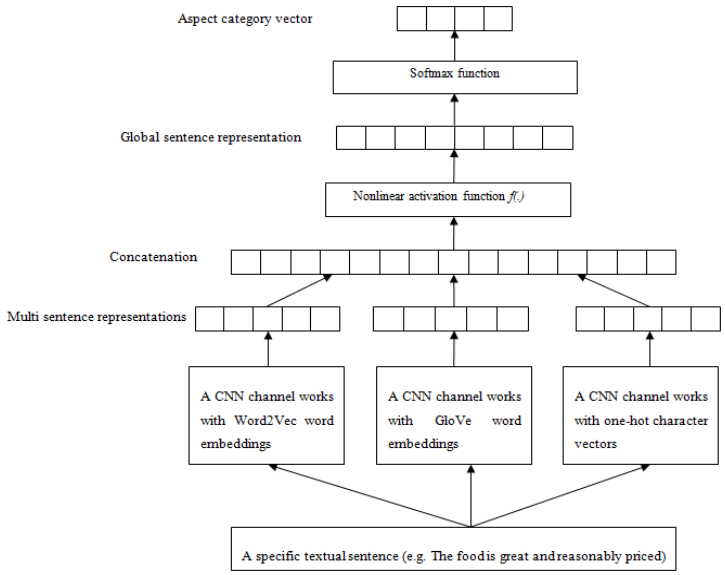

Figure 2: Illustration of the MCNN model for the aspect category detection task.

Given an input sentence d containing n words $\{w_1, w_2, ..., w_n\}$, we present each channel and each layer with the necessary formulations and notations as follows.

Figure 2 illustrates our proposed model which can be interpreted as follows: - Firstly the input sentence is represented by the three different representations including: the word2vec representation from (Mikolov et al. [4]), the word2vec representation from the GloVe model (Pennington et al. [5]), and the character based one-hot vector.

- Secondly each of the representations above is putted into an CNN module (called CNN channel) and then consequently we obtain three representation vectors that we call multiple sentence representations.

- Thirdly these three sentence representations at the second step are concatenated and then putted through a non-linear activation function to generate

the called global sentence representation.

- Finally we use the global representation as input for a single neural network (i.e. a perceptron) using the softmax function to generate the aspect category vector for aspect prediction.

Note that all the components in this proposed architecture are unified in a joint model. It means all parameters of the individual components (i.e. the CNN channels as well as the neural network layers) are simultaneously learned to optimize the same target function. That is the reason why we say it is a joint model for learning the shared presentation (or unified presentation) from individual information sources. This is different from the hybrid model in which each CNN channel is learned separately and then the outputs are combined to form the global representation for prediction task. This hybrid model is presented in section 3.4.

In the following we will formulate the training phase for this proposed MCNN in which the CNN channels as well as the learning algorithm are described in detail.

Channel 1

Taking the Word2Vec word embeddings $x_1^1, x_2^1, ..., x_n^1$ of words in the input sentence d as inputs, we use three convolutional filters with widths of one, two, and three to semantically encode the unigrams, bigrams, and trigrams in d. For the j-th convolutional filter, $j \in \{11, 12, 13\}$, we denote $U^j \in \mathrm{R}^{\mathrm{C^j} \times \mathrm{m^j}}$ as a weight matrix and $b^j \in \mathrm{R}^{\mathrm{C^j}}$ as a bias vector, where C^j is the output dimension, $m^j = h^j.m^1$, m^1 is the dimension of word embedding, and h^j is the window size for combining the word embedding features. We apply the convolution component on $x_1^1, x_2^1, ..., x_n^1$ to generate the output vector $s^j = [s_1^j, s_2^j, ..., s_{C^j}^j]$.

We then concatenate the output vectors of the three convolutional filters (corresponding to $j = 11, 12, 13$) to obtain the resulting vector as the sentence representation denoted by:

$$s^1 = [s^{11}, s^{12}, s^{13}] \tag{3}$$

where $s^1 \in \mathrm{R}^{\mathrm{C^{11}} + \mathrm{C^{12}} + \mathrm{C^{13}}}$.

Channel 2

By taking the GloVe word embeddings $x_1^2, x_2^2, ..., x_n^2$ of words in the input sentence d as inputs, in a similar manner to Channel 1, we use the three convolutional filters and apply the convolution component on $x_1^2, x_2^2, ..., x_n^2$ to obtain the output vector as the representation of d, which is denoted by:

$$s^2 = [s^{21}, s^{22}, s^{23}], \tag{4}$$

where $\mathrm{s}^2 \in \mathrm{R}^{\mathrm{C}^{21}+\mathrm{C}^{22}+\mathrm{C}^{23}}$. C^{21}, C^{22}, and C^{23} are the corresponding output dimensions of the filters.

Channel 3

This channel takes the one-hot character vectors of words in the input sentence d as inputs. The word embedding vector of each word in d is computed by applying the convolution component. In particular, given a word $w \in \{w_1, w_2, ..., w_n\}$ containing the one-hot character vectors $e_1, e_2, ..., e_q$, the i-th character embedding is computed as follows:

$$c_i = W.e_i, \tag{5}$$

where $W \in \mathrm{R}^{\mathrm{m}^{\mathrm{ch}}\mathrm{x}|\mathrm{V}|}$ is a character embedding matrix, each column of W is the m^{ch}-dimensional embedded vector, V is the character vocabulary, and $|V|$ is the character vocabulary size.

Let $\mathrm{U}^{\mathrm{ch}} \in \mathrm{R}^{\mathrm{C}^{\mathrm{ch}} \times 3\mathrm{m}^{\mathrm{ch}}}$ and $\mathrm{b}^{\mathrm{ch}} \in \mathrm{R}^{\mathrm{C}^{\mathrm{ch}}}$ be the parameters of the model at the character-level and we apply the convolution component to the sequence of character embeddings $c_1, c_2, ..., c_q$ to obtain the word vector x_w^3 of word w with fixed dimensions C^{ch}. Note that we use a window size of three to combine the character embedding features, which is denoted by $3m^{ch}$.

After obtaining the word vectors $x_1^3, x_2^3, ..., x_n^3$, we use the three convolutional filters and apply the convolution component on $x_1^3, x_2^3, ..., x_n^3$ to obtain the output vector as:

$$s^3 = [s^{31}, s^{32}, s^{33}], \tag{6}$$

where $\mathrm{s}^3 \in \mathrm{R}^{\mathrm{C}^{31}+\mathrm{C}^{32}+\mathrm{C}^{33}}$; C^{31}, C^{32}, and C^{33} are the corresponding output

dimensions of the filters.

Hidden layer

First, we set $s = [s^1, s^2, s^3]$ as the concatenation of the sentence representation vectors obtained from the three channels described above. The global sentence representation is then generated by using a nonlinear hidden layer, which is formulated as:

$$s^* = \phi(U^g.s + b^g), \tag{7}$$

where ϕ is an element-wise activation function such as sigmoid, $U^g \in \mathrm{R}^{\mathrm{p}\times\mathrm{m}}$ and $b^g \in \mathrm{R}^{\mathrm{p}}$ are model parameters (which are learned at the training stage), p is the size of this layer, and $m = \sum_{i=1}^{3}\sum_{j=1}^{3} C^{ij}$.

Output layer

Finally, an output layer is applied to score all possible labels according to the features in the hidden layer. The aspect category vector of the input sentence is computed as follows:

$$\hat{a} = g(U^o.s^* + b^o), \tag{8}$$

where g is an element-wise activation function such as softmax, $U^o \in \mathrm{R}^{\mathrm{kxp}}$ is a weight matrix, and $b^o \in \mathrm{R}^{\mathrm{k}}$ is a bias vector.

3.3. Training Model

We are given a set of labeled sentences $D = \{\mathrm{d}_1,\mathrm{d}_2,...,\mathrm{d}_{|D|}\}$ as the training data set. Our model is trained by minimizing a negative likelihood over the training set. For each sentence $d \in D$, let $a_d \in \mathrm{R}^{\mathrm{k}}$ and $\hat{a}_d \in \mathrm{R}^{\mathrm{k}}$ be the desired target vector and the predicted vector respectively. These vectors are representations of the aspect category label in sentence d. The cross entropy function for the data set D is computed by:

$$E(\theta) = -\sum_{d\in\mathrm{D}}\sum_{i=1}^{k} a_{di}\log\hat{a}_{di} + \frac{1}{2}\lambda\|\theta\|^2, \tag{9}$$

where $\theta = [U^j, b^j, W]$, for $j \in \{11, 12, 13, 21, 22, 23, 31, 32, 33, ch, g, o\}$, λ is the regularization parameter and $\|\theta\|^2 = \sum_i {\theta_i}^2$ is a norm regularization term. In order to compute the parameters θ, we apply a back-propagation algorithm with stochastic gradient descent to minimize this cost function. Each element of the weights in the parameters θ is updated at time $t + 1$ according to the formula:

$$\theta(t+1) = \theta(t) - \eta \frac{\partial E(\theta)}{\partial \theta}, \tag{10}$$

where η is the learning rate.

As an overall view, we present Algorithm1, which includes the necessary steps for learning the proposed model's parameters.

Algorithm 1 Learning Model for Aspect Category Detection.

Input: A set of textual sentences $D = \{\mathrm{d}_1, \mathrm{d}_2, ..., \mathrm{d}_{|D|}\}$, each sentence $d \in$ D is assigned with an aspect category label a_d, a set of GloVe word embeddings, a set of Word2Vec word embeddings, and a character vocabulary V;

Output: The model (i.e., parameters in θ)

Step 1: Initialize the values: the learning rate η, error threshold ε, iterative threshold I, and regularization parameter λ;

Initialize the parameters in θ;

Step 2: for t=*1* to I do

for each sentence $d \in$ D do

Compute representation vector of d in Channel 1 using Eq. 3

Compute representation vector of d in Channel 2 using Eq. 4

Compute representation vector of d in Channel 3 using Eq. 6

Compute the global sentence representation s^* using Eq. 7;

Compute the aspect category vector $\hat{a}_d$ using Eq. 8;

endfor

Update parameters in θ at time t+*1* using Eq. 10;

Compute the objective function by: $\frac{1}{|D|} \sum_{d=1}^{|D|} \left| a_d - \hat{a}_d(t) \right|$

Break if the objective function is less than the error threshold ε;

endfor

It should be noted that when implementing this algorithm, we employ the mini-batching technique (Bottou et al. [25]; Cotter et al.[26]) to solve the problem of large data. In this case, when the number of textual sentences in D is large, the data set D is then divided into smaller subsets and we implement step 2 of the algorithm 1 for each subset in a step by step manner. This solution will make the algorithm faster.

3.4. A hybrid variant model of MCNN

In this section we will represent the hybrid architecture as an optional type for combining the individual CNN channels of representations. This architecture is illustrated in the Figure 3. We treat each input representation as a single task for the problem of aspect category detection. As shown in this figure, firstly we train independently the three CNN models corresponding with the three kinds of sentence representation including the word2vec representation from (Mikolov et al. [4]), the word2vec representation from the GloVe model (Pennington et al. [5]), and the character based one-hot vector. And after that, for each input sentence we will generate its three representations by using these obtained CNN models. These output vectors are concatenated to get the global representation for the input sentence, which is then used as input in a neural network model for the aspect category detection as described in the Figure 3d. We call this architecture the CNN1+CNN2+CNN3 hybrid model.

Note that this hybrid architecture is different from the MCNN architecture on the important point that: in the MCNN model, the global sentence representation are learned from the individual CNN channels in a joint model meanwhile in the Hybrid model the global sentence representation is trivial combination from the separate CNN models.

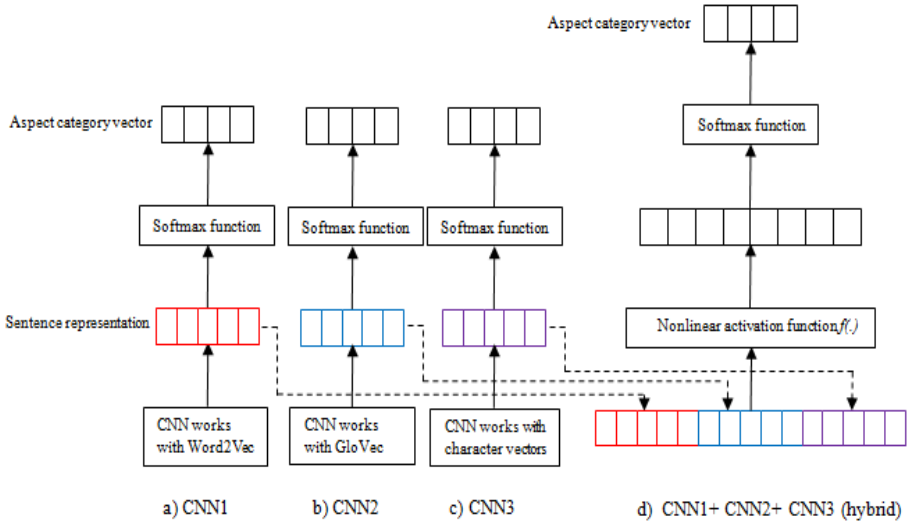

Figure 3: An illustration of the CNN1 + CNN2 + CNN3 hybrid model for aspect category detection task

4. Experiments

4.1. Experimental data and implementation of the MCNN model

We used a data set[2] containing 190,655 sentences extracted from 52,574 reviews, as used in previous studies (Brody et al. [27]; Ganu et al. [8]; Wang et al. [28]). The data set contained six aspect category labels comprising *Price, Food, Service, Ambience, Anecdotes*, and *Miscellaneous*, and four aspect sentiment labels, i.e., *Positive, Negative, Neutral* and *Conflict*. Each sentence was labeled with both aspect category and sentiment labels. We randomly selected 75% of the labels for training our MCNN model and the remaining 25% were used for evaluation. Some key statistics for the data set are shown in Table 1.

[2]http://spidr-ursa.rutgers.edu/datasets/

Table 1: Statistics for the dataset

Aspect	Number of sentences
Price	5848
Food	59882
Service	29959
Ambience	23638
Anecdotes	24528
Miscellaneous	46800
Total	**190,655**

For the word embedding sets used in Channel 1 and Channel 2 of the proposed model, we used Google Word2Vec[3] and GloVe[4], respectively. We also used a set of 52 common English characters (including the English alphabet characters, numbers, special characters, and unknown character) to obtain the character set used in Channel 3 of the proposed model.

To implement the MCNN model, we applied algorithm 1 with the following parameters:

- For $j \in \{11, 12, 13, 21, 22, 23, 31, 32, 33\}$, the output dimension of the filter j-th in Channel 1 and Channel 2, C^j=50.

- In Channel 3, the char embedding dimension and word embedding dimension were set to 7 and 50, respectively.

- The size of the hidden layer p=150.

- For $j \in \{11, 12, 13, 21, 22, 23, 31, 32, 33, ch, g, o\}$, the bias vectors b^j were initialized as zero. According to Glorot et al. [29], we also initialized all of the matrices U^j with uniform samples in $(-\sqrt{\frac{6}{r^j+c^j}}, \sqrt{\frac{6}{r^j+c^j}})$, where r^j and c^j are the numbers of rows and columns, respectively, and the character embedding matrix W was also initiated in a similar manner.

- We set the regularization rate $\lambda = 10^{-4}$, learning rate $\eta = 0.035$, error

[3]https://code.google.com/archive/p/Word2Vec/

[4]http://nlp.stanford.edu/projects/glove/

threshold $\varepsilon = 0.00001$, and iterative threshold $I = 50$.

It should be noted that Algorithm 1 is for the aspect category detection task, but the aspect sentiment classification task can also be performed in a similar manner.

4.2. Experimental Environment and Execution Time

The proposed method was implemented in JAVA with NetBean IDE 7.2 on an ASUS PC with the following specifications: Intel Core i5-2450M CPU @ 2.50 GHz processor, 4.00 GB memory, andWindows 7 Ultimate Service Pack 1 Operating System. The average execution time for each iteration was 2 minutes, 33 seconds.

4.3. Evaluation

As mentioned earlier, our MCNN model can be considered as a combination of the three CNN channels, where each channel is treated as a single CNN model. Therefore, for comparison, we designed each channel and different combinations of them as variants of the MCNN model. To simplify the presentation, we divided them into three groups as follows.

- (1) The models in group 1 used only a single CNN channel, where the CNN1 model use only Channel 1 with Word2Vec word embeddings as inputs, the CNN2 model use only Channel 2 with the GloVe word embeddings as inputs, and the CNN3 model used Channel 3 with the one-hot character vectors as inputs.

- (2) The models in group 2 used combination of the channels to create models with hybrid features, where we implement a CNN1+CNN2 hybrid model and CNN1+CNN2+CNN3 hybrid model. (Note that in Section 3.4 we have explained about the CNN1+CNN2+CNN3 hybrid model).

- (3) The models in group 3 used combination of the channels to create joint learning models, where the CNN1+CNN2 model was a combination of two channels, i.e., CNN1 and CNN2, and the MCNN model was a combination of three channels, i.e., CNN1, CNN2, and CNN3.

We also evaluated the models in our model groups based on comparisons with some baseline models as follows.

- **NLSE** (Astudillo et al. [30]): A model for identifying an embedding sub-space projection to improve the word embeddings for a given task.
- **HFL: Hybrid Feature Learning** (Zhou et al. [22]): A model for learning hybrid features for aspect category detection. The inputs for this model are the sentence vectors computed by averaging all the word vectors. The shared and aspect-specific features are learned in the hidden layer of the model.
- **CNN-non-static** (Kim et al. [18]): This model is used for fine tuning the word embeddings for each task.
- **CNN-multichannel**: This model proposed by Kim et al. [18] uses two channels with two sets of word vectors, where one is kept stable and the other is fine tuned using a back-propagation algorithm.
- **CharSCNN** (dos Santos et al. [1]): This model exploits the one-hot character vectors and pre-trained word embeddings for sentiment analysis in short texts, but its architecture only uses a version of the pre-trained embeddings (i.e., Word2Vec).

All of these models were evaluated in two aspect-based sentiment analysis tasks: aspect category detection and aspect sentiment classification. Each model was run five times, before obtaining the average values for the metrics in the testing phase. It should be noted that in our method, the baseline models used the pre-trained GloVe word embeddings as inputs.

Aspect Category Detection Results

Table 2 shows the mean values for the precision, recall, and F1 scores with each method. For simplicity, we only used the F1-scores to analyze the results. In group 1, most of the models performed worse than the baseline models

Table 2: Aspect category detection results

	Method	Precision	Recall	F1 score
Baselines	NLSE	77.82	81.53	79.63
	HFL	79.11	80.97	80.03
	CNN-non-static	79.08	81.23	80.14
	CNN-multichannel	80.18	81.41	80.79
	CharSCNN	82.30	80.17	81.22
Our	CNN1	78.45	76.87	77.65
	CNN2	80.02	78.57	79.29
	CNN3	77.78	73.18	75.41
	CNN1+CNN2 (hybrid)	81.88	79.79	80.82
	CNN1+CNN2+CNN3 (hybrid)	81.91	80.25	81.07
	CNN1+CNN2	83.40	81.27	82.32
	MCNN	**83.94**	**81.61**	**82.76**

because the baseline models (except the HFL model) fine tuned the word embeddings during training. The HFL model learns hybrid features and does not fine tune the word embeddings, but it also performed slightly better than the CNN1, CNN2, and CNN3 models. Among the models in group 1, the CNN2 model performed slightly better than both CNN1 and CNN3 because the inputs for CNN2 are the learned word embeddings from GloVe, which capture more semantics than the Word2Vec word embeddings. The CNN3 model did not perform as well as the CNN1 and CNN2 models, thereby indicating that using the one-hot character vectors was not as effective as the word embeddings.

In group 2, we found that although the models did not fine tune the word embeddings during the training stage in the same manner as the baseline models, they performed better than all of the models in group 1 and the baseline models, except for the CharSCNN model. In addition, the CNN1+CNN2+CNN3 (hybrid feature) model performed slightly better than the CNN1+CNN2 (hybrid feature) model, thereby demonstrating the important role of character-level information in the hybrid features. In group 3, all of the models performed better than the baseline models and better than the models in group 1 and group 2.

Table 3: Aspect sentiment prediction results

	Method	Accuracy
Baselines	NLSE	81.49
	HFL	81.71
	CNN-non-static	82.13
	CNN-multichannel	82.79
	CharSCNN	83.35
Our	CNN1	79.97
	CNN2	80.50
	CNN3	77.83
	CNN1+CNN2 (hybrid)	82.81
	CNN1+CNN2+CNN3 (hybrid)	83.02
	CNN1+CNN2	83.68
	MCNN	**84.16**

These results indicate that the proposed MCNN models with inputs represented by the two different types of word embeddings (i.e., Word2Vec and GloVe) and the one-hot character vectors performed the best.

Aspect Sentiment Prediction Results

Table 3 shows the accuracy metrics for the aspect sentiment classification task. The models in group 3 achieved better results than those in group 1 and group 2, thereby confirming the effectiveness of the MCNNs in our proposed method.

5. Conclusion

In this study, we proposed a new MCNN model for exploiting multiple word embeddings and one-hot character vectors in aspect-based sentiment analysis. Various models were developed with different combinations of input representations. The experimental results showed that the combination of Word2Vec word embeddings, GloVe word embeddings, and one-hot character vectors in the framework of the MCNN-based model achieved the best accuracy. We also consider that this model may be useful for other sentiment analysis problems

such as aspect ratings detection. In addition, this model could be applied efficiently to languages other than English.

Acknowledgement

This study was supported by The Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.01-2014.22.

References

[1] C. N. dos Santos, M. Gatti, Deep convolutional neural networks for sentiment analysis of short texts, in: COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland, 2014, pp. 69–78.

[2] X. Zhang, J. J. Zhao, Y. LeCun, Character-level convolutional networks for text classification, in: NIPS, 2015, pp. 649–657.

[3] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P. P. Kuksa, Natural language processing (almost) from scratch, Journal of Machine Learning Research 12 (2011) 2493–2537.

[4] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, CoRR abs/1301.3781 (2013).

[5] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, 2014, pp. 1532–1543.

[6] T. Alsinet, J. Argelich, R. Béjar, C. Fernández, C. Mateu, J. Planes, Weighted argumentation for analysis of discussions in twitter, Int. J. Approx. Reasoning 85 (2017) 21–35.

[7] M. Dragoni, G. Petrucci, A fuzzy-based strategy for multi-domain sentiment analysis, Int. J. Approx. Reasoning 93 (2018) 59–73.

[8] G. Ganu, N. Elhadad, A. Marian, Beyond the stars: Improving rating predictions using review text content, in: 12th International Workshop on the Web and Databases, WebDB 2009, Providence, Rhode Island, USA, June 28, 2009, 2009.

[9] J. Wagner, P. Arora, S. Cortes, U. Barman, D. Bogdanova, J. Foster, L. Tounsi, DCU: aspect-based polarity classification for semeval task 4, in: Proceedings of the 8th International Workshop on Semantic Evaluation, SemEval@COLING 2014, Dublin, Ireland, August 23-24, 2014., 2014, pp. 223–229.

[10] S. Kiritchenko, X. Zhu, C. Cherry, S. Mohammad, Nrc-canada-2014: Detecting aspects and sentiment in customer reviews, in: Proceedings of the 8th International Workshop on Semantic Evaluation, SemEval@COLING 2014, Dublin, Ireland, August 23-24, 2014., 2014, pp. 437–442.

[11] S. Poria, E. Cambria, A. F. Gelbukh, Aspect extraction for opinion mining with a deep convolutional neural network, Knowl.-Based Syst. 108 (2016) 42–49.

[12] D. Tang, B. Qin, T. Liu, Aspect level sentiment classification with deep memory network, in: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016, 2016, pp. 214–224.

[13] Y. Wang, M. Huang, X. Zhu, L. Zhao, Attention-based LSTM for aspect-level sentiment classification, in: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016, 2016, pp. 606–615.

[14] D. Pham, A. Le, Learning multiple layers of knowledge representation for aspect based sentiment analysis, Data Knowl. Eng. 114 (2018) 26–39.

[15] N. Kalchbrenner, E. Grefenstette, P. Blunsom, A convolutional neural network for modelling sentences, in: Proceedings of the 52nd Annual Meeting

of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers, 2014, pp. 655–665.
URL http://aclweb.org/anthology/P/P14/P14-1062.pdf

[16] M. Lakshmana, S. Sellamanickam, S. K. Shevade, K. Selvaraj, Learning semantically coherent and reusable kernels in convolution neural nets for sentence classification, CoRR abs/1608.00466 (2016).

[17] Y. Shen, X. He, J. Gao, L. Deng, G. Mesnil, Learning semantic representations using convolutional neural networks for web search, in: 23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014, Companion Volume, 2014, pp. 373–374.

[18] Y. Kim, Convolutional neural networks for sentence classification, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, 2014, pp. 1746–1751.

[19] W. Yin, H. Schütze, Multichannel variable-size convolution for sentence classification, in: Proceedings of the 19th Conference on Computational Natural Language Learning, CoNLL 2015, Beijing, China, July 30-31, 2015, 2015, pp. 204–214.

[20] Y. Zhang, S. Roller, B. C. Wallace, MGNC-CNN: A simple approach to exploiting multiple word embeddings for sentence classification, in: NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016, 2016, pp. 1522–1527.

[21] A. Alghunaim, M. Mohtarami, S. Cyphers, J. Glass, A vector space approach for aspect based sentiment analysis, in: Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing, VS@NAACL-HLT 2015, June 5, 2015, Denver, Colorado, USA, 2015, pp. 116–122.

[22] X. Zhou, X. Wan, J. Xiao, Representation learning for aspect category detection in online reviews, in: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA., 2015, pp. 417–424.

[23] W. X. Zhao, J. Jiang, H. Yan, X. Li, Jointly modeling aspects and opinions with a maxent-lda hybrid, in: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP 2010, 9-11 October 2010, MIT Stata Center, Massachusetts, USA, A meeting of SIGDAT, a Special Interest Group of the ACL, 2010, pp. 56–65.

[24] R. Collobert, J. Weston, A unified architecture for natural language processing: deep neural networks with multitask learning, in: Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008, 2008, pp. 160–167.

[25] L. Bottou, Stochastic learning, in: Advanced Lectures on Machine Learning, ML Summer Schools 2003, Canberra, Australia, February 2-14, 2003, Tübingen, Germany, August 4-16, 2003, Revised Lectures, 2003, pp. 146–168.

[26] A. Cotter, O. Shamir, N. Srebro, K. Sridharan, Better mini-batch algorithms via accelerated gradient methods, in: Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain., 2011, pp. 1647–1655.

[27] S. Brody, N. Elhadad, An unsupervised aspect-sentiment model for online reviews, in: Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 2-4, 2010, Los Angeles, California, USA, 2010, pp. 804–812.

[28] L. Wang, K. Liu, Z. Cao, J. Zhao, G. de Melo, Sentiment-aspect extraction based on restricted boltzmann machines, in: Proceedings of the 53rd

Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers, 2015, pp. 616–625.

[29] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010, 2010, pp. 249–256.

[30] R. F. Astudillo, S. Amir, W. Ling, M. J. Silva, I. Trancoso, Learning word representations from scarce and noisy data with embedding subspaces, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers, 2015, pp. 1074–1084.