Senti-EGCN: An Aspect-Based Sentiment Analysis System Using Edge-Enhanced Graph Convolutional Networks

Chen Li[1,*], Junjun Zheng[2,*], Peng Ju[3], and Yasuhiko Morimoto[3]
[1]Graduate School of Informatics, Nagoya University, Chikusa, Nagoya 464-8602, Japan
[2]Graduate School of Information Science and Technology, Osaka University, Suita, Osaka 565-0871, Japan
[3]Graduate School of Engineering, Hiroshima University, Higashi-Hiroshima, Hiroshima 739-8527, Japan
li.chen.z2@a.mail.nagoya-u.ac.jp, jzheng@ist.osaka-u.ac.jp, jupeng9511@gmail.com, morimo@hiroshima-u.ac.jp
*corresponding authors

Abstract—Aspect-based sentiment analysis systems aim to classify sentences according to specified aspects. Most previous studies have used graph convolutional networks (GCNs) to analyze syntactic features and used word dependencies for syntactic context and aspects. However, traditional GCNs have limitations in exploring syntactic dependency graphs, such as missing information on edges in syntactic dependency trees. For accurate sentiment prediction on specific aspects, an aspect-based **senti**ment analysis system using **e**dge-enhanced **g**raph **c**onvolutional **n**etworks (Senti-EGCN) is developed. In particular, a bidirectional long short-term memory (Bi-LSTM) network is employed to extract the contextual features of sentences. Thereafter, a transformer encoder with a self-attention mechanism is used to analyze the interrelationships and global features of words in long texts. Next, the Bi-LSTM network is used to enforce the sentence structure through the word dependency tree. The dependency tree analyzes the words' dependencies to enhance the representation of information. A bidirectional GCN (Bi-GCN) uses message passing to propagate information across the nodes in a parsed dependency tree. In addition, an aspect-specific masking technique is applied to reduce redundant information in the hidden representation by masking contextual information outside the aspect and improve the accuracy. Finally, attention scores are calculated in the last layer for sentiment classification. The experimental results demonstrated that the proposed Senti-EGCN outperforms other baseline models in most metrics on the three benchmark datasets.

Keywords–Aspect-based sentiment analysis system; graph convolutional networks; syntactic dependency tree; aspect-specific masking

1. Introduction

Aspect-based sentiment analysis [1] is a branch of sentiment classification. The goal of the analysis is to classify sentences according to specified aspects. A sentence has various aspects, and the sentiment of each aspect may belong to a different category. Sentiment analysis systems for aspects have a wide range of application in practice. For example, a customer comments in the review system of a restaurant: "The menu is limited, but all the food taste good." If the "menu" is considered as the aspect, then the sentiment of the customer is negative (i.e., "limited"). However, when "food" is considered as the aspect, the sentiment is positive (i.e., "good"). When significant contextual information for each aspect of the text is not well captured, aspect-based sentiment analysis system may make incorrect predictions. Therefore, the correct classification of text in such a system can help analyze user demands and thus provide better services to users.

Nowadays, deep learning-based natural language processing models are being used in aspect-based sentiment analysis systems to model the various aspects of contextual information in texts. The effectiveness of these neural networks (e.g., recurrent neural networks (RNNs) and Transformer) has also been demonstrated [2, 3, 4]. However, these models cannot capture the syntactic dependencies between word pairs and aspects well. For example, the contextual word (e.g., "taste good") associated with a specific aspect (e.g., "food") should earn a higher attention score. Although adding syntactic constraints to attention mechanisms is effective to a certain extent, this issue has still not been fundamentally addressed. In addition, as a phrase sometimes determines the sentiment of a sentence more than a word, convolutional neural networks (CNNs) can also be used to identify the aspects of descriptive multiword phrases. For example, the phrase "a bit more friendly" in the sentence "The staff should be a bit more friendly" has a long-term dependency on the subject "staff." The aspect of the sentence can be better described. However, traditional CNNs cannot well determine the sentiment depicted by non-adjacent multiple words owing to the limitation of the local receptive field.

The tree structure of dependency trees in graph convolutional networks (GCNs) can shorten the distances among the various aspects of a sentence and its associated words. Moreover, it provides a discriminable syntactic path for the text to propagate information through the tree. Hence, a dependency tree can capture the syntactic relationships among words in an efficient structure. These features of dependency trees can allow deep learning approaches to effectively capture long-range dependencies. Therefore, the representation of nodes and their positional relationships in graphs can be learned using the dependency tree of GCNs. That is, a GCN can project the syntactically relevant words to the target aspect through the syntactic dependency tree. However, although the edges in the syntactic dependency tree contain structured information

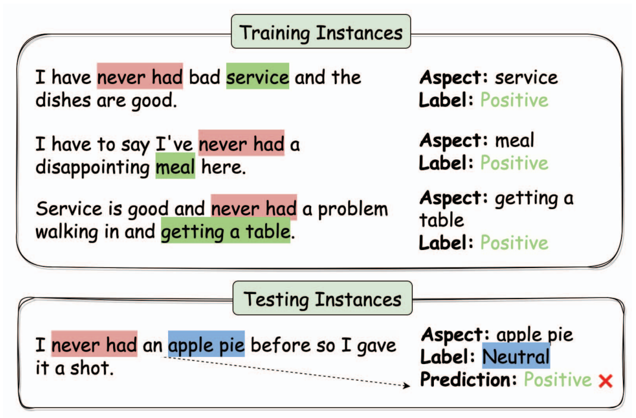
Training Instances
I have never had bad service and the dishes are good.
Aspect: service
Label: Positive
I have to say I've never had a disappointing meal here.
Aspect: meal
Label: Positive
Service is good and never had a problem walking in and getting a table.
Aspect: getting a table
Label: Positive
Testing Instances
I never had an apple pie before so I gave it a shot.
Aspect: apple pie
Label: Neutral
Prediction: Positive ✗

Figure 1. Example of spurious associations of a customer review system.

about the graph, they are not used in GCNs.

The complexity of natural language syntax and semantics may lead to spurious associations between aspects and contexts. Deep learning approaches with GCNs that lack edge information are more prone to making wrong predictions. Figure 1 provides an example of a customer review of a restaurant to explain the spurious associations that exist in aspect-based sentiment analysis systems. A high association exists between the contextual word "never had" and the sentiment polarity label "Positive" in historical reviews, even without considering the aspect words. However, the sentiment label of the review by the new customer is not "Positive." Therefore, the trained sentiment classifier cannot discriminate the association between "never had" and the sentiment label "Neutral" in the test instance. Furthermore, the deep models of such systems are more limited in their ability to extract feature representations for long texts, which significantly reduces the robustness and generalization ability of the models.

Inspired by the above mentioned limitations, we propose an aspect-based **senti**ment analysis system using the **e**dge-enhanced **GCN** (Senti-EGCN), which operates on the dependency trees of sentences for the accurate sentiment analysis of specific aspects. In particular, the Senti-EGCN first gains the node embeddings of words through a bidirectional long short-term memory (Bi-LSTM) network and then uses a Bi-GCN to model the structure of sentences through dependency trees of words. Furthermore, the dependency tree enhances the information representation of words by analyzing their dependencies. The Bi-LSTM captures and extracts the features of contextual information between consecutive words. The Bi-GCN follows the syntactic path of a dependency tree to enhance the embedding by modeling dependencies. The information can be transferred from the context to the target aspect, which indicates that the encoding of the target aspect can be used for sentiment classification. The major contributions of this study are summarized as follows:

- **A novel hybrid architecture:** The Bi-LSTM, transformer, and edge-enhanced Bi-GCN are employed in an aspect-based sentiment analysis system. The Senti-EGCN uses the syntax between words to construct a graph and then obtains the representation of the text through the graph neural network.
- **Enhanced semantic edges:** Using a dependency tree shortens the distance between aspects and associated words and also provides a syntactic path for the text. Furthermore, the edge information of syntactic dependency trees is considered in the Senti-EGCN. Such information can enhance the syntactic projection between the associated words and the target aspect.
- **Superior performance:** The effectiveness of the proposed approach is validated on three benchmark datasets. The experimental results demonstrate that the Senti-EGCN model outperforms other baseline models in most metrics.

2. Related Work

2.1. Aspect-based Sentiment Analysis Systems

Aspect-based sentiment analysis systems are a branch of natural language processing. In earlier studies in this context, a set of features (e.g., bag-of-words features and sentiment dictionary features) have been extracted to train sentiment classifiers. The rule-based and statistical-based approaches [5] are commonly used, which rely on labor-intensive feature engineering. Typically, an aspect-based sentiment analysis system consists of four main tasks. The first task is called the aspect term sentiment analysis (ATSA), which aims to identify the sentiment polarity of the specified aspect terms in a sentence. The ATSA is usually modeled as a classification problem. The aspect category is a predefined set that describes the things implicitly expressed in a sentence. Aspect category sentiment analysis (ACSA), the second task, aims to identify the sentiment polarities of specified aspect categories in a sentence. ACSA is also a classification problem. The other two tasks, aspect term extraction (ATE) and aspect category extraction (ACE) are inverse tasks of the first two tasks. ATE in a given sentence can be modeled as a sequence labeling problem. ACE aims to identify predefined aspect categories in sentences describing things, which is usually modeled as a multilabel classification problem.

2.2. RNN-based Sentiment Analysis Systems

In recent years, with the successful application of deep learning in various fields, more and more researchers have started to use deep learning to solve problems [6, 7, 8]. LSTM [9] is the most basic deep learning model for aspect-based sentiment analysis systems. It extracts the features of the current subsequence to feed into the next time step. The hidden-layer features of the last time step are used as the input of the classifier for sentiment classification. TD-LSTM [10] uses a Bi-LSTM to learn the dependency between the context and target. ATAE-LSTM [2] combines an LSTM with an attention mechanism and embeds aspects in the calculation of attention weights. RAM [3] combines a Bi-LSTM and multiple attention mechanisms. TNet-LF [11] is a Bi-attention mechanism model that learns the attention weights of contexts and aspect words interactively.

A transformer is powerful in capturing and extracting long-range dependency features of long sequences. It calculates the attention score for the current word and all the words in the sentence. Therefore, it can directly capture long-range dependent features without the need to pass back the sequence of hidden-layer nodes similar to an RNN or increase the network depth to capture long-range features similar to a CNN. Furthermore, a transformer has parallel computing capability and does not need to rely on the features of previous time steps.

2.3. GCN-based Sentiment Analysis Systems

GCNs are a type of deep learning model that can effectively capture the relationships between the entities in a graph. ASGCN [12] was the first GCN that modeled the relationships between aspects and other entities in the input text, which effectively captures the sentiment polarity of specific aspects and improves the performance of aspect-based sentiment analysis systems. Inspired by the advantages of the Bi-LSTM, transformer, and GCNs, in this paper, we propose a hybrid deep learning model for aspect-based sentiment analysis. Concretely, a Bi-LSTM is used to extract the deep features of sentences. A transformer encoder with self-attention guides model learning from a global viewpoint. A bi-GCN with a dependency-parsing layer aims to learn the graph features of syntactic dependency tree.

3. SENTI-EGCN MODEL

3.1. Overview

Figure 2 demonstrates an overview of the proposed Senti-EGCN sentiment analysis system. The Senti-EGCN primarily consists of seven submodules: (a) A word embedding layer parses sentences and embeds words into the vector space; (b) A Bi-LSTM network captures the contextual information of sentences; (c) A transformer with a self-attention mechanism analyzes the interrelationships and global features of words in long texts; (d) A dependency-parsing layer identifies the dependencies among words in a sentence and creates a dependency tree; (e) A Bi-GCN uses message passing to propagate information across the nodes in the parsed dependency tree; (f) An Aspect-specific masking layer reduces redundant information in the hidden representation by masking contextual information outside the aspect and improves the accuracy of the system; (g) A sentiment classification layer aims to classify the sentiment of a given sentence into one of several predefined categories, typically positive class, negative class, or neutral class.

3.2. Word Embedding Layer

A sentence composed of n words containing an m-word aspect can be tokenized as a sequence $\boldsymbol{S}$ as follows:

$$\boldsymbol{S} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_\gamma, \cdots, \boldsymbol{w}_{\gamma+m}, \cdots, \boldsymbol{w}_n\}, \tag{1}$$

where $\boldsymbol{w} \in \boldsymbol{S}$ denotes the word token and γ denotes the starting index of an aspect. Each word token is embedded into a low-dimensional vector $\boldsymbol{e}_\gamma \in \mathbb{R}^{d_w}$ and here d_w is the dimension of the word embedding.

3.3. Bi-LSTM

Word embedding vectors are then used as the input to the Bi-LSTM network. The Bi-LSTM is employed to capture the contextual information of sentences. In general, a Bi-LSTM network consists of two submodules, i.e., the forward and backward LSTMs. For the forward LSTM, the hidden vector $\overrightarrow{\boldsymbol{h}}_t^{LSTM}$ of the current time step is calculated based on the input embedding $\boldsymbol{e}_t$ and $\overrightarrow{\boldsymbol{h}}_{t-1}^{LSTM}$ of the previous time step. The backward LSTM is constructed using $\boldsymbol{e}_t$ and $\overleftarrow{\boldsymbol{h}}_{t+1}^{LSTM}$ of the next time step. Finally, $\overrightarrow{\boldsymbol{h}}_t^{LSTM}$ and $\overleftarrow{\boldsymbol{h}}_t^{LSTM}$ are concatenated into the hidden vector $\boldsymbol{h}_t^{LSTM}$. The corresponding calculations are as follows:

$$\overrightarrow{\boldsymbol{h}}_t^{LSTM} = \text{LSTM}(\boldsymbol{e}_t, \overrightarrow{\boldsymbol{h}}_{t-1}^{LSTM}),\ \overrightarrow{\boldsymbol{h}}_t^{LSTM} \in \mathbb{R}^{d_h}, \tag{2}$$

$$\overleftarrow{\boldsymbol{h}}_t^{LSTM} = \text{LSTM}(\boldsymbol{e}_t, \overleftarrow{\boldsymbol{h}}_{t+1}^{LSTM}),\ \overleftarrow{\boldsymbol{h}}_t^{LSTM} \in \mathbb{R}^{d_h}, \tag{3}$$

$$\boldsymbol{h}_t^{LSTM} = \overrightarrow{\boldsymbol{h}}_t^{LSTM} \oplus \overleftarrow{\boldsymbol{h}}_t^{LSTM},\ \boldsymbol{h}_t^{LSTM} \in \mathbb{R}^{2\times d_h}, \tag{4}$$

where d_h denotes the hidden-layer dimension of the Bi-LSTM and $\oplus$ is the concatenation operator.

3.4. Transformer

A transformer encoder with a self-attention mechanism is employed to analyze the interrelationship and global features of words in long sentences. The sequential positions of a sentence are input to the sinusoidal positional encoding functions, which are given by

$$\text{P}_{pos,2i} = \sin(\frac{pos}{10000^{\frac{2i}{d_{model}}}}), \tag{5}$$

$$\text{P}_{pos,2i+1} = \cos(\frac{pos}{10000^{\frac{2i}{d_{model}}}}), \tag{6}$$

where pos is the position of a word in a sentence and i denotes the ith dimension of the embedding with the dimension d_{model}. Thereafter, the superposition vector of the embedding and positional encoding of the input sentence is used as the input of the transformer. A transformer calculates attention weights as follows:

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^{\text{T}}}{\sqrt{d_k}})\boldsymbol{V}, \tag{7}$$

where $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ denote the query, key, and value matrices with the dimensions d_k, d_k, and d_v. Typically, a transformer uses a multihead attention mechanism to project the query, key, and value matrices h times to enhance the internal association of contexts with the following formulas:

$$\text{Multihead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = [\text{Head}_1, \cdots, \text{Head}_h]\boldsymbol{W}, \tag{8}$$

$$\text{Head}_i = \text{Attention}(\boldsymbol{Q}\boldsymbol{W}_Q, \boldsymbol{K}\boldsymbol{W}_K, \boldsymbol{V}\boldsymbol{W}_V). \tag{9}$$

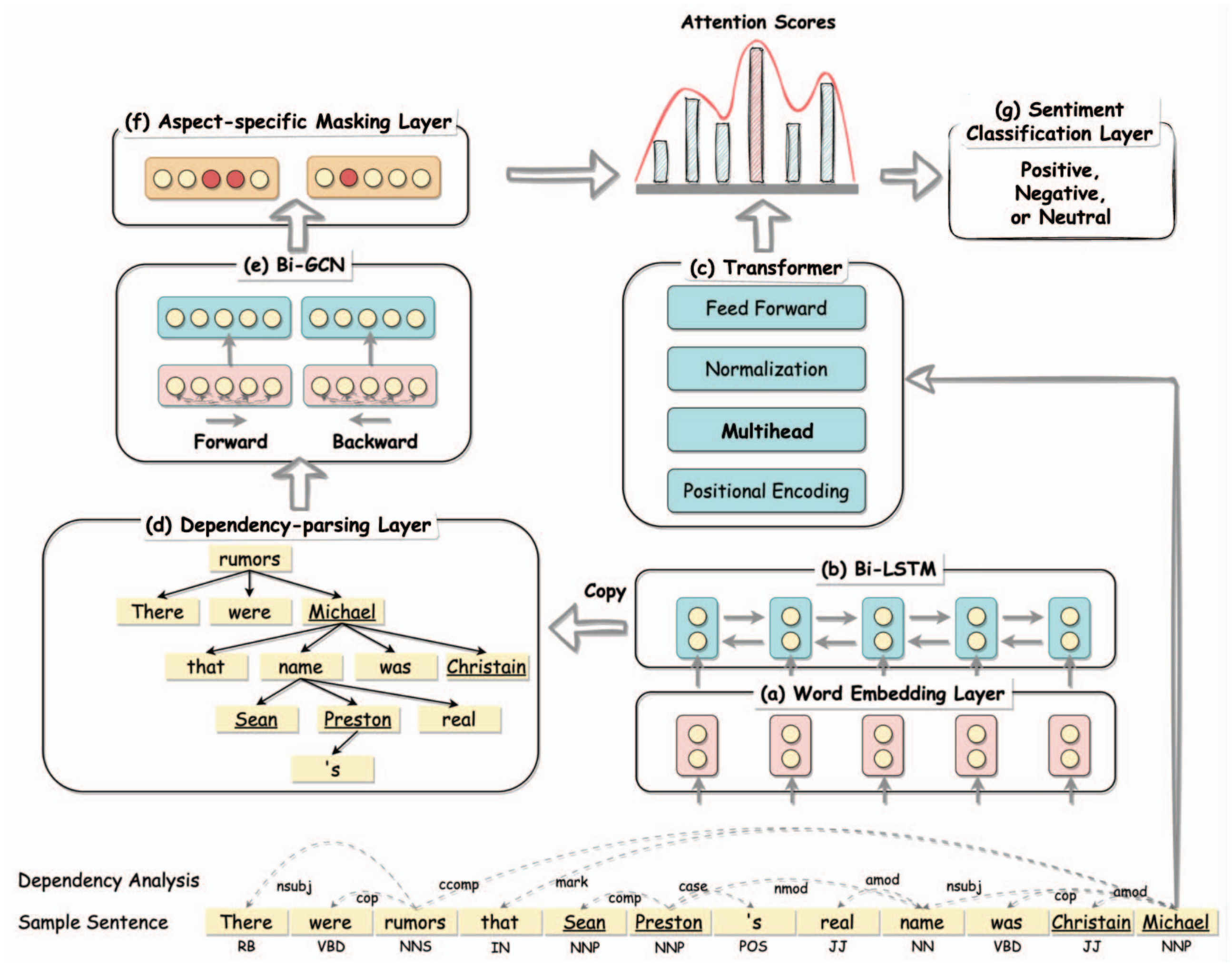

Figure 2. Overview of the Senti-EGCN architecture for the aspect-based sentiment analysis system.

In the above $\boldsymbol{W}$, $\boldsymbol{W}_Q$, $\boldsymbol{W}_K$, and $\boldsymbol{W}_V$ indicate the weight matrices of the self-attention mechanism. Next, the transformer encoder applies two normalization layers to extract the deep features of the context.
Let $\mathbf{Z}_{out}$ be the output matrix of the transformer encoder, and it can be formulated as

$$\boldsymbol{Z}_{out} = \text{Transformer}(\boldsymbol{S}). \tag{10}$$

Finally, the calculated global attention scores are used in the final sentiment classification layer.

3.5. Dependency-parsing Layer

The output hidden vector $\boldsymbol{h}_t$ of the Bi-LSTM network is used as an input to the dependency-parsing layer of a Bi-GCN, and the dependencies among words in a sentence are identified by creating a dependency tree.
Formally, the Bi-GCN initializes a matrix to store the weights indicating the edge information. First, the adjacency matrix of $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is obtained from the dependency tree of a given sentence. The traditional adjacency matrix can be specified as follows:

$$\boldsymbol{A}_{ij} = \begin{cases} 1, & i = j \\ 1, & i \neq j,\ i \text{ and } j \text{ have dependencies} \\ 0, & \text{otherwise} \end{cases}. \tag{11}$$

However, owing to the discrete representation of the adjacency matrix (i.e., 0 and 1), a GCN cannot effectively capture the more rich edge information. Therefore, the adjacency matrix containing nondiscrete edge information is proposed as follows:

$$\boldsymbol{A}_{ij} = \begin{cases} 1, & i = j \\ \text{SDI}(i,j), & i \neq j,\ i \text{ and } j \text{ have dependencies} \\ 0, & \text{otherwise} \end{cases}, \tag{12}$$

where $\text{SDI}(i,j)$ represents the function of syntactic dependency information between the words i and j. SDI can be calculated as shown below.

$$\mathrm{SDI}(i,j) = \frac{\mathrm{Count}(\mathrm{sd}(i,j))}{\mathrm{Count}(\mathrm{sd}(\cdot))}, \tag{13}$$

where $\mathrm{sd}(i,j)$ and $\mathrm{sd}(\cdot)$ denote the syntaxes of the word pair i and j and dataset, respectively.

3.6. Bi-GCN

A GCN is a variant of CNN that can be used to encode the information of structured graph data. For a given text graph with n words, an adjacency matrix can be obtained by searching the syntactic matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ for syntactic dependency information. The output of word i at layer l is $\boldsymbol{h}_i^l$. The graph is represented as follows:

$$\boldsymbol{h}_i^l = \sigma(\sum_{j=1}^{n} \tilde{\boldsymbol{A}}_{ij}\boldsymbol{W}^l\boldsymbol{h}_j^{l-1} + \boldsymbol{b}^l), \tag{14}$$

where $\boldsymbol{W}^l$ denotes the weight of the linear transformation, $\boldsymbol{b}^l$ indicates a bias value, and σ represents a nonlinear activation function (e.g., sigmoid or tanh function). In a GCN, the sentence's dependency tree offers syntactic constraints for an aspect of a sentence to determine the descriptive words based on syntactic distance. The GCN can handle the case where noncontiguous words describe the polarity of an aspect. Thus, the GCN is selected for aspect-based sentiment analysis. Here, we use the Bi-GCN to update each word's representation with the following equation:

$$\overrightarrow{\boldsymbol{h}}_i^l = \sum_{j=1}^{n} \tilde{\boldsymbol{A}}_{ij}\boldsymbol{W}^l\boldsymbol{h}_j^{l-1}, \tag{15}$$

$$\overleftarrow{\boldsymbol{h}}_i^l = \sum_{j=1}^{n} \tilde{\boldsymbol{A}}_{ij}^{\mathrm{T}}\boldsymbol{W}^l\boldsymbol{h}_j^{l-1}, \tag{16}$$

where $\overrightarrow{\boldsymbol{h}}_i^l$ and $\overleftarrow{\boldsymbol{h}}_i^l$ denote the forward and backward representations of the word i in lth layer, respectively. Finally, the two representations are concatenated by

$$\tilde{\boldsymbol{h}}_i^l = \overrightarrow{\boldsymbol{h}}_i^l \oplus \overleftarrow{\boldsymbol{h}}_i^l, \tag{17}$$

$$\boldsymbol{h}_i^l = \mathrm{ReLU}\left(\frac{\tilde{\boldsymbol{h}}_i^l}{d_i + 1} + \boldsymbol{b}^l\right), \tag{18}$$

here d_i is the degree of the ith token in the adjacency matrix. It should be noted that the weight $\boldsymbol{W}^l$ and the bias $\boldsymbol{b}^l$ are trainable parameters.

3.7. Aspect-specific Masking Layer

Aspect-specific masking layer can reduce redundant information in the Bi-GCN hidden representation by masking the non-aspect text. Concretely, the aspect-specific masking can be represented as

$$\boldsymbol{h}_i^l = 0,\ 1 \le i < \gamma,\ \gamma + m - 1 < t \le n. \tag{19}$$

The output of the zero mask layer is aspect-oriented features:

$$\boldsymbol{H}_{mask}^l = \left\{\boldsymbol{0}, \cdots, \boldsymbol{h}_\gamma^l, \cdots, \boldsymbol{h}_{\gamma+m-1}^l, \cdots, \boldsymbol{0}\right\}. \tag{20}$$

With the graph convolution, the feature $\boldsymbol{H}_{mask}^l$ has a perceptual context around aspects while considering syntactic dependencies and long-range multiword relations. In addition, the attention mechanism can be used to obtain a vector representation of the hidden layer. The attention weights are calculated as follows:

$$\beta_i = \sum_{j=1}^{n} \boldsymbol{h}_j^{\mathrm{T}}\boldsymbol{h}_j^l, \tag{21}$$

$$\alpha_i = \frac{\exp(\beta_i)}{\sum_{j=1}^{n}\exp(\beta_j))}. \tag{22}$$

Finally, the sum of the above representation scores and attention scores of the transformer are used for the classification of sentiments as follows:

$$\boldsymbol{r}_{out} = \sum_{i=1}^{n} \alpha_i\boldsymbol{h}_i + \boldsymbol{Z}_{out}. \tag{23}$$

3.8. Sentiment Classification Layer

The $\boldsymbol{r}_{out}$ calculated in the previous layer is used as an input to the fully connected layer for sentiment classification as follows:

$$\boldsymbol{p} = \mathrm{softmax}(\boldsymbol{W}_p\boldsymbol{r}_{out} + \boldsymbol{b}). \tag{24}$$

3.9. Training

Owing to the cross entropy and L_2 regularization, the loss function of the Senti-EGCN is designed as follows:

$$\mathrm{Loss} = -\sum_{\hat{p}\in C} \log \boldsymbol{p}_{\hat{p}} + \lambda \left\|\Theta\right\|_2, \tag{25}$$

where $\hat{p}$ and C indicate the label and dataset, respectively, $\boldsymbol{p}_{\hat{p}}$ represents the $\hat{p}$th element of $\boldsymbol{p}$, and Θ and λ are the trainable parameters and coefficient of the L_2-regularization. For a better understanding of the Senti-EGCN model, Algorithm 1 shows the training and testing processes of the proposed Senti-EGCN model.

4. Experiments

We conducted experiments to demonstrate the usefulness of our presented Senti-EGCN model. Three benchmark datasets (i.e., Rest14 [13], Rest15 [14], and Rest16 [15]) are used for the evaluation.

4.1. Datasets

The three benchmark datasets contain comments and rating information of customers for restaurants. Specifically, the Rest14 dataset consists of a total of 4,728 customer comments. The numbers of data used for training and testing are 3,608 and 1,120, respectively. The Rest15 dataset contains a total of 1,746 reviews. In particular, 1,204 customer review messages are used for training and 542 are used for testing. Rest16 is the benchmark dataset for 2016 and includes 2,364 reviews. The number of training data used are 1,748. The remaining 616 reviews are testing data. Based on previous studies, we deleted text that had conflicting polarity or unclear meaning in the sentences of Rest15 and Rest16 datasets. Table I describes the statistics of the three datasets in detail.

Algorithm 1: Procedures of the Senti-EGCN.

```
   // Read sample sentences from datasets
   // A word embedding layer
1  Tokenize a sample sentence with Eq. (1).
2  Embed word tokens in the word embedding layer.
3  Perform dependency analysis for the sample sentence.
   // A Bi-LSTM layer
4  Capture the contexual features of the sample sentence
     using a Bi-LSTM layer with Eqs. (2)–(4).
   // A transformer layer
5  Encode positions for the transformer encoder layer using
     Eqs. (5) and (6).
6  Use a transformer encoder to capture the global features
     of sentences using Eqs. (7)–(10).
   // A dependency-parsing layer
7  Replace the weight matrix (11) using Eq. (12) to store
     more edge information of a graph.
8  Compute SDI using Eq. (13).
   // A Bi-GCN submodule
9  Employ a Bi-GCN submodule to extract the features of
     the structured graph using Eqs. (14)–(18).
   // An aspect-specific masking layer
10 Reduce redundant information in a Bi-GCN using the
     aspect-specific masking technique using Eqs. (19))–(23).
   // A sentiment classification layer
11 Construct a classifier by applying fully connected layers
     to classify sentiments using Eq. (24).
   // Model training
12 Train the Senti-EGCN model using Eq. (25).
   // Testing
13 Evaluate the performance of the proposed model using
     the test dataset.
```

TABLE I
THE STATISTICS FOR THE THREE DATASETS.

Dataset		# Positive	# Neutral	# Negative
Rest14	Train	2,164	637	807
	Test	728	196	196
Rest15	Train	912	36	256
	Test	326	34	182
Rest16	Train	1,240	69	439
	Test	469	30	117

4.2. Experimental Setting

Pretrained GloVe vectors [16] are used in the experiments as initialization vectors for the word embedding layer. The dimensionality of the word embedding and hidden layer of Bi-LSTM is 300. The number of Bi-GCN layers is 3. The weights in the model are initialized with uniform distribution. Additionally, the Adam optimizer with a learning rate of 0.001 is used to optimize the proposed Senti-EGCN model. The L_2-regularization and batch size for training are set to 10^{-5} and 32, respectively. During the training phase, Senti-EGCN performs up to 100 epochs.

4.3. Baselines

The proposed Senti-EGCN model is compared to a range of baseline models to evaluate its effectiveness. Baseline models are described as follows:

- **SVM**: The support vector machine (SVM) is a machine learning model applied on SemEval 2014 task 4 using the traditional methodology for feature extraction.
- **LSTM** [9]: The hidden-layer vector of the last time step of the LSTM is used for sentiment classification.
- **TD-LSTM** [10]: The target-dependent LSTM (TD-LSTM) extracts the interaction features between the target sentiment and whole context for sentiment prediction.
- **MemNet** [17]: MemNet uses textual information as an external memory and obtains sentence representations using deep memory networks with word embeddings.
- **AOA** [18]: Attention-over-attention neural networks (AOA) is a neural network-based deep learning model that uses the attention-over-attention mechanism to improve the classification.
- **IAN** [11]: Interactive attention networks (IAN) use two LSTM variants to extract the representations for aspect terms and contexts separately. Thereafter, the IAN concatenates the context representations and aspects.
- **TNet-LF** [19]: TNet-LF uses a forward context-preserving transformation to save and enhance the information part of the context.
- **ASGCN** [12]: The first aspect-specific graph convolutional network (ASGCN) uses a GCN to analyze the sentiment of graph-based aspects.
- **MGAN** [4]: Multi-grained attention network (MGAN) aims to identify the sentiment expressed towards specific aspects or entities in a given text.
- **Sentic LSTM** [20]: Sentic LSTM uses an aspect-specific attention mechanism, a commonsense knowledge embedding module, and an attentive LSTM for sentiment analysis.

4.4. Evaluation Measures

The accuracy value (Acc) and Macro F1 score (F1) are used to test and validate the performance of the proposed Senti-EGCN. Acc and F1 are the commonly used measures in multiclassification tasks, which intuitively qualify the model capability of prediction and classification. In particular, Acc is the ratio of correctly classified samples and F1 is the weighted average of the accuracy and recall. Formally, Acc and F1 are computed as follows:

$$\text{Acc} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{26}$$

TABLE II
AVERAGE ACCURACY AND MACRO-F1 SCORE RESULTS OF THE PROPOSED MODEL IN COMPARISON WITH BASELINES.

Model	Rest14		Rest15		Rest16	
	Acc (%) ↑	F1 (%) ↑	Acc (%) ↑	F1 (%) ↑	Acc (%) ↑	F1 (%) ↑
SVM	80.16	–	–	–	–	–
LSTM	78.13	67.47	77.37	55.17	86.80	63.88
TD-LSTM	78.00	66.73	76.39	58.70	82.16	54.21
MemNet	79.61	69.64	77.31	58.28	85.44	65.99
AOA	79.97	70.42	78.17	57.02	87.50	66.21
IAN	79.26	70.09	78.54	52.65	84.74	55.21
TNet-LF	80.42	71.03	78.47	59.47	**89.07**	**70.43**
ASGCN	80.86	72.19	79.34	60.78	88.69	66.64
MGAN	81.25	71.94	79.36	57.26	87.06	62.29
Sentic LSTM	79.43	70.32	**79.55**	60.56	83.01	68.22
Senti-EGCN	**81.70**	**73.63**	79.34^{3rd}	**61.99**	88.80^{2nd}	68.96^{2nd}

* The values highlighted in bold represent the maximum values of the corresponding columns. The "2nd" and "3rd" indicate the second and third maximum values attained within the respective columns, respectively.

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{27}$$

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{28}$$

$$\text{F1} = 2 \times \frac{\text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}, \tag{29}$$

where TP indicates the counts of true-positive samples, and TN represents the counts of true-negative samples, while FP, and FN denote the false-positive and false-negative samples, respectively.

4.5. Results

Table II presents a comparison of the Acc and F1 scores of the proposed Senti-EGCN model with the other 10 baseline models. The Senti-EGCN model outperforms the other baseline models for most measures on the Rest14, Rest15, and Rest16 datasets. The Senti-EGCN model performs the best on the Rest14 dataset, with the Acc and F1 scores significantly higher than the other baseline models. On the Rest15 dataset, Sentic LSTM had the highest Acc score (i.e., 79.55), but its F1 score was low at 60.56%, which is quite smaller than Senti-EGCN (i.e., 61.99%). On the Rest16 dataset, the Acc and F1 scores of TNet-LF were only slightly higher than those of the proposed Senti-EGCN. A possible reason is that the feature vectors of the parent nodes is as critical as the features originating from the sub-nodes, and relying on the tree to be processed as a directed graph leads to loss of information. The results indicate that the Senti-EGCN model achieves results that are comparable to those of the TNet-LF model on the REST16 dataset. The Senti-EGCN model outperforms the ASGCN on all datasets, indicating that word dependencies can improve classification results.

In summary, the proposed Senti-EGCN model performs well on the Acc and F1 measures on the three bechmark datasets, which demonstrates the usefulness and effectiveness of the Senti-EGCN model.

5. CONCLUSION

In this paper, we presented an aspect-based sentiment analysis system using an edge-enhanced bidirectional graph convolutional network. The Senti-EGCN model uses Bi-LSTM to learn the syntax and semantics of the text, and the features of its hidden layer are used for syntactic dependency tree building in the Bi-GCN. Furthermore, a transformer encoder layer extracts global information from the text and is used to improve the ability of the Senti-EGCN to capture features in long texts. The dependency-parsing layer in the Bi-GCN extracts the graph structure of the syntax of texts and learns the association between aspect and context. Thereafter, the aspect-specific masking layer is used to reduce the redundant information in the hidden layer of the GCN and improve the accuracy of the analysis system. Finally, the outputs of the Bi-GCN and transformer are used as the inputs of the classifier for sentiment classification.

In the future, we will investigate more advanced feature extraction techniques, such as utilizing pre-trained language models such as BERT or GPT, to capture richer semantic information and further improve Senti-EGCN's ability to understand the nuances of complex text. In addition, we will extend the Senti-EGCN model to support multiple languages (e.g., Chinese and Japanese languages) to enable cross-lingual sentiment analysis. This may require adapting the model's architecture and training procedures to effectively handle different linguistic constructs.

Acknowledgement

This research was supported by JSPS KAKENHI Grant Number JP21K17742.

References

[1] N. K. Laskari and S. K. Sanampudi, "Aspect based sentiment analysis survey," *IOSR Journal of Computer Engineering (IOSR-JCE)*, vol. 18, no. 2, pp. 24–28, 2016.

[2] Y. Wang, M. Huang, X. Zhu, and L. Zhao, "Attention-based LSTM for aspect-level sentiment classification," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016, pp. 606–615.

[3] P. Chen, Z. Sun, L. Bing, and W. Yang, "Recurrent attention network on memory for aspect sentiment analysis," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 452–461.

[4] F. Fan, Y. Feng, and D. Zhao, "Multi-grained attention network for aspect-level sentiment classification," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 3433–3442.

[5] L. Jiang, M. Yu, M. Zhou, X. Liu, and T. Zhao, "Target-dependent twitter sentiment classification," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011, pp. 151–160.

[6] C. Li, M. He, M. Qaosar, S. Ahmed, and Y. Morimoto, "Capturing temporal dynamics of users' preferences from purchase history big data for recommendation system," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 5372–5374.

[7] C. Li, C. Yamanaka, K. Kaitoh, and Y. Yamanishi, "Transformer-based objective-reinforced generative adversarial network to generate desired molecules," in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, 2022, pp. 3884–3890.

[8] C. Li, J. Zheng, H. Okamura, and T. Dohi, "Software reliability prediction through encoder-decoder recurrent neural networks," *International Journal of Mathematical, Engineering and Management Sciences*, vol. 7, no. 3, p. 325, 2022.

[9] D. Tang, B. Qin, X. Feng, and T. Liu, "Effective LSTMs for target-dependent sentiment classification," in *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 3298–3307.

[10] Y. Ma, H. Peng, T. Khan, E. Cambria, and A. Hussain, "Sentic LSTM: A hybrid network for targeted aspect-based sentiment analysis," *Cognitive Computation*, vol. 10, pp. 639–650, 2018.

[11] D. Ma, S. Li, X. Zhang, and H. Wang, "Interactive attention networks for aspect-level sentiment classification," *ArXiv Preprint ArXiv:1709.00893*, 2017.

[12] C. Zhang, Q. Li, and D. Song, "Aspect-based sentiment classification with aspect-specific graph convolutional networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 4568–4578.

[13] M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, and S. Manandhar, "SemEval-2014 task 4: Aspect based sentiment analysis," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, 2014, pp. 27–35.

[14] M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar, and I. Androutsopoulos, "Semeval-2015 task 12: Aspect based sentiment analysis," in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, 2015, pp. 486–495.

[15] M. Pontiki, D. Galanis, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, M. AL-Smadi, M. Al-Ayyoub, Y. Zhao, B. Qin, O. De Clercq *et al.*, "Semeval-2016 task 5: Aspect based sentiment analysis," in *Proceedings of the Workshop on Semantic Evaluation (SemEval-2016)*, 2016, pp. 19–30.

[16] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

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

[18] B. Huang, Y. Ou, and K. M. Carley, "Aspect level sentiment classification with attention-over-attention neural networks," in *Proceedings of the 11th International Conference on SBP-BRiMS 2018*, 2018, pp. 197–206.

[19] X. Li, L. Bing, W. Lam, and B. Shi, "Transformation networks for target-oriented sentiment classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018, pp. 946–956.

[20] Y. Ma, H. Peng, and E. Cambria, "Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive LSTM," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.