Robust Sentence Classification by Solving Out-of-Vocabulary Problem with Auxiliary Word Predictor

Sang-Seok Park[1], Yunseok Noh[1], Seyoung Park[1](✉), and Seong-Bae Park[2]

[1] School of Computer Science and Engineering, Kyungpook National University, Daegu 41566, Korea
{sspark,ysnoh}@sejong.knu.ac.kr, seyoung@knu.ac.kr
[2] Department of Computer Engineering, Kyung Hee University, Yongin 17104, Korea
sbpark71@khu.ac.kr

Abstract. In recent years, deep learning methods have achieved outstanding performances in sentence classification. However, many sentence classification models do not consider the out-of-vocabulary (OOV) problem, which generally appears in sentence classification tasks. Input units smaller than words, such as characters or subword units, have been considered the basic unit for sentence classification to cope with the OOV problem. Although this approach naturally solves the OOV problem, it has obvious performance limitations because a character by itself has no meaning, whereas a word has a definite meaning. In this paper, we propose a neural sentence classification model that is robust to the OOV problem, even though the proposed model utilizes words as the basic unit. To this end, we introduce the *unknown word prediction* (UWP) task as an auxiliary task to train the proposed model. Owing to joint training of the proposed model with the objectives of classification and UWP, the proposed model can represent the meanings of entire sentences robustly even if a sentence includes a number of unseen words. To demonstrate the effectiveness of the proposed model, a number of experiments are conducted using several sentence classification benchmarks. The proposed model consistently outperforms two baselines over all four benchmark datasets in terms of the classification accuracy.

Keywords: Sentence classification · Out-of-vocabulary problem · Neural network · Multi-task learning

1 Introduction

Sentence classification is a fundamental task in natural language processing (NLP), which is being studied extensively for sentiment analysis in social media

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2016-0-00145, Smart Summary Report Generation from Big Data Related to a Topic).

and political ideology analysis, among other purposes [3,15]. Recently, deep learning approaches that employ recurrent neural networks (RNNs), convolutional neural networks (CNNs), and attention mechanisms have been shown to be effective for sentence classification. Among them, BLSTM2DCNN [18] employs a two-dimensional (2D) convolutional network on top of a bidirectional LSTM for text classification. BLSTM2DCNN has been shown to perform well on several text-classification tasks because the network can capture the dependency among feature vector dimensions via 2D convolutions and the bidirectional long-term contextual information via BLSTM. By contrast, DARLM [19] is another recently developed neural network for sentence classification. This model includes two attention subnets that attending different parts of a sentence with each other. Then, an example discriminator assigns a label to the given sentence by utilizing attention information from both subnets. The neural models proposed in these studies have network architectures and decision mechanisms optimized for sentence classification.

Despite these advances in deep learning for sentence classification, many sentence classifiers do not consider the out-of-vocabulary (OOV) problem, which appears in almost all sentence classification tasks. During training, neural classifiers have access to complete information about a sentence to be classified. By contrast, in practice, these classifiers may be applied to sentences containing multiple unseen words. This OOV problem interferes with the prediction of neural classifiers, and the problem becomes severe when the unseen words in a sentence deliver the key information that determines the class of the sentence. One possible solution for the OOV problem is using characters or subwords [13] as basic units for sentence classification instead of words [17]. Because characters and subwords are smaller units than words, sentence classifiers based on such small units can avoid the OOV problem naturally. However, the performance of character-level sentence classifiers is inferior to that of word-level models, even though character-level sentence classifiers have considerably deeper and more complex network structures [7]. This is because a character by itself has no meaning, whereas a word has a definite meaning.

In this paper, we propose a neural sentence classification model that is robust to the OOV problem, even if the proposed model utilizes words as the basic unit for classification. To this end, we introduce the *unknown word prediction* (UWP) task as an auxiliary task to train the proposed model. The UWP task predicts the would-be word when an unknown word is included in a sentence. To train a network for this task, some proportion of the words in training sentences are randomly selected and replaced with the $\langle unk \rangle$ token. Then, a network is trained to predict the words to be substituted instead of the $\langle unk \rangle$ tokens by considering all other words in a sentence. The objective of UWP is similar to that of the *masked language model* (MLM) [2], which has been proved to be useful for obtaining robust and contextual word representations of a given sentence.

The proposed neural network consists of three sub-networks, namely, a shared sentence encoder, sentence classification network, and an UWP network. The sentence encoder takes a sentence, that is, a sequence of words, as input and outputs a sequence of contextual word representations. Because the auxiliary UWP task is executed to generate robust word representations that can be used for sentence classification, the sentence encoder should be shared across both the main and the auxiliary task-specific networks. The UWP network is placed on top of the sentence encoder, and then the network predicts the word that is the original word of a given $\langle unk \rangle$ token based on all other known words in the sentence. Concurrently, the sentence classification network takes a sequence of word representations from the sentence encoder and performs classification based on the given sequence.

In multi-task learning, an auxiliary task can give the model useful hints, which are difficult to learn in the main task [1]. By applying the UWP task to a network as an auxiliary task for sentence classification, the proposed model obtains such hints for solving the OOV problem from two perspectives. The first perspective is that a neural classifier can be configured to predict the approximate meanings of unseen words. Thus, the UWP task provides a direct solution for the OOV problem encountered in sentence classification. Another perspective is that word representations of *known* words become more contextual. As a result, the meaning representation of the entire sentence becomes robust, even if a sentence includes a number of unseen words.

To demonstrate the effectiveness of the proposed model, a number of experiments are conducted on several sentence classification benchmarks including SST-1, SST-2, TREC-6, and TREC-50. In a comparison with the baselines without the UWP auxiliary task, the proposed model consistently outperforms the baselines over all four benchmark datasets; especially, on the SST-1 benchmark, the performance gain in terms of accuracy is up to 1.2% compared to that of the baselines.

The rest of this paper is organized as follows. In Sect. 2, we briefly introduce previous works on recent neural models for sentence classification. In Sects. 3 and 4, we describe the learning algorithm and the architecture of the proposed model. The experimental setting and the results are given in Sects. 5 and 6. Finally, we conclude the study in Sect. 7.

2 Related Work

In recent years, deep learning methods, including modern neural modules such as recurrent units, convolutions, and attention mechanisms, have yielded notable performance when applied to sentence classification. Even more recently, the developers of most deep learning models have blended more than two of the aforementioned modules to enhance performance. In BLSTM2DCNN [18], 2D convolutions are introduced, and their filters are defined across the feature vector dimension as well as the word sequence. These 2D convolutions summarize the contextual information generated by a bidirectional LSTM to classify a sentence. The DARLM [19] combines all three of the aforementioned modules; it is thus composed of a convolutional layer for text encoding, two different attention mechanisms for feature selection, and two LSTM layers on top of each attention mechanism for aggregating the contextual information. These studies show the importance of generating appropriate contextual representations and summarizing them.

Breaking words down into smaller units is one of solutions to the OOV problem. A number of character-level [8,17] and subword-level [5,13] classification models have been proposed. Essentially, these models seem to eliminate the OOV problem from sentence classification. However, the smaller units rarely convey meanings and increase the length of the input sequence. By contrast, the proposed model uses a word as the basic unit and infers contextual meanings by considering the surrounding words.

Multi-task learning is widely used to improve sentence classification performance [12]. To facilitate multi-task learning, the tasks to be performed should be related each other. When this condition is satisfied, an auxiliary task can help improve the performance of the associated main task through the provision of additional hints that can only be obtained from the auxiliary task. For instance, the execution of a word-level sentiment classification task can improve the performance of the associated sentence level sentiment classification task [16]. This makes sense because the polarity of each word in a sentence is crucial for determining the sentiment of the entire sentence. It is also intuitive that understanding the contextual meaning of each word in a sentence is very important for predicting the class of the sentence. In addition, human beings understand a sentence that contains words unknown to them by contextually approximating the meaning of those words. Based on this intuition, we set UWP as the auxiliary task for sentence classification.

The proposed UWP task was motivated by the masked language model (MLM), which is used for training BERT [2]. In the training procedure of MLM, a neural network is forced to predict the original words of masked words in a sentence. With this training, the network produces more robust word representations, even when a word is masked. Unlike the MLM in BERT, which applied to a very large-scale corpus to obtain robust word representations, we show that with relatively small data, the UWP task is adequately effective as an auxiliary task for sentence classification.

3 Sentence Classification with Auxiliary Word Predictor

Figure 1 describes the overall architecture of the proposed model. As shown in this figure, the proposed model follows a general network structure for multi-task learning, which comprises one shared sub-network and multiple task-specific sub-networks [12]. In the proposed model, the shared network is a sentence encoder, and two task-specific networks constitute a sentence classification network for the main task and an UWP network for the auxiliary task. Let $D = \{(x, \mathbf{y})\}$ be the training dataset for sentence classification, where $x = (w_1, \ldots, w_L)$ is a input sentence of length L, and $\mathbf{y} \in \{0, 1\}^C$ is a one-hot vector for the class label of a sentence x. Because D does not provide the training data for the auxiliary task, we first generate a training dataset $D' = \{(x', \mathbf{y})\}$ for both tasks. x' is a

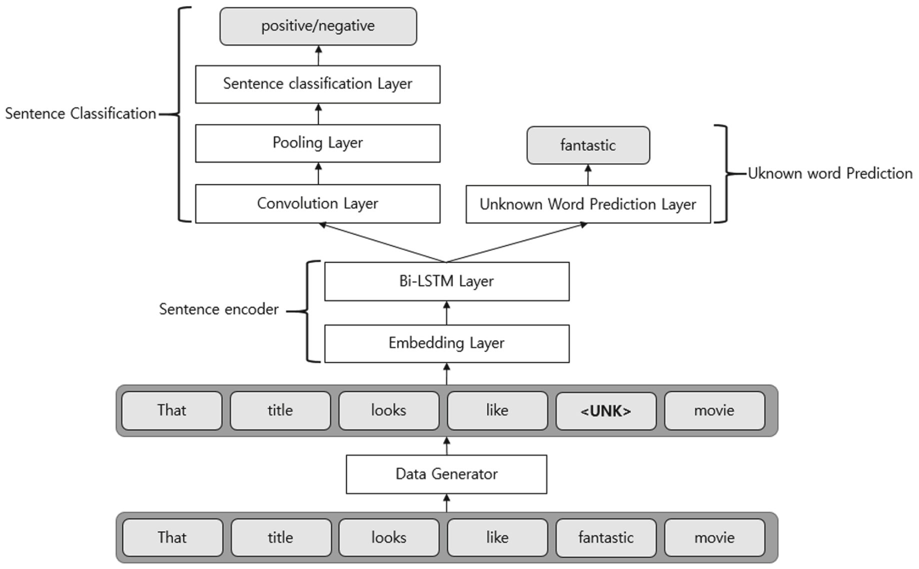

Fig. 1. Architecture of proposed sentence classification network with an auxiliary word predictor.

corrupted x obtained by replacing a few words with the unknown token $\langle unk \rangle$. In this study, 15% of the words in x were randomly replaced with the $\langle unk \rangle$ token.

After D' is prepared, x' with U unknown words is input to the RNN-based sentence encoder f_{enc} to produce a sequence of word vectors $H = (\mathbf{h}_1, \ldots, \mathbf{h}_L)$. Then, the CNN-based sentence classification network f_{cls} takes H and predicts the probability distribution of the class label $\mathbf{y}$ of x. Thus, two parameter sets θ_{cls} and θ_{enc} with respect to f_{enc} and f_{cls} are trained to minimize

$$\sum_{(x',\mathbf{y}) \in D'} \mathcal{L}_{cls}(\mathbf{y}, f_{cls}(f_{enc}(x'; \theta_{enc}); \theta_{cls})), \tag{1}$$

where $\mathcal{L}_{cls}$ is the cross-entropy loss.

The UWP network f_{aux} for the auxiliary task is a feed-forward neural network, and it takes H as the input. Because more than one $\langle unk \rangle$ token can be included in x', f_{aux} predicts the original word for each $\langle unk \rangle$ token. Then, θ_{aux}, a set of all parameters of the auxiliary word predictor, is jointly trained with θ_{enc} to minimize

$$\sum_{(x', i: w_i = \langle unk \rangle) \in D'} \mathcal{L}_{aux}(\mathbf{w}_i, f_{aux}(f_{enc}(x'; \theta_{enc}); \theta_{aux})), \tag{2}$$

where $\mathbf{w}_i$ is a one-hot vector for the i-th word in x, which is replaced by the $\langle unk \rangle$ token, and $\mathcal{L}_{aux}$ is the cross-entropy loss.

Algorithm 1. Training procedure of entire proposed model

```
input      : Training set $D = (x, \mathbf{y})$, hyperparameters $\alpha$ and $\lambda$
Parameters : $\Theta = (\theta_{enc}, \theta_{cls}, \theta_{aux})$
initialize : All parameters $\Theta$ are randomly initialized.
1  repeat
2      $D_{batch} \leftarrow$ sample$(D, b)$ // sample a minibatch of size $b$
3      $D'_{batch} \leftarrow \emptyset$ // initialize the corrupted training set
4      for $(x, \mathbf{y}) \in D_{batch}$ do
5          $(x', \mathbf{y}) \leftarrow$generate$(D_{batch})$ // sample a corrupted tuple
6          $D'_{batch} \leftarrow D'_{batch} \cup \{(x', \mathbf{y})\}$
7      end
       // joint training of entire networks
8      Train $f_{enc}$ and $f_{cls}$ by Eq. 1
9      Train $f_{enc}$ and $f_{aux}$ by Eq. 2
10 until convergence;
```

The goal of the auxiliary task is to help the sentence encoder produce a robust representation of x. Thus, f_{enc} should be optimized to jointly minimize both loss $\mathcal{L}_{cls}$ and $\mathcal{L}_{aux}$. As a result, the final loss of the proposed model is as follows.

$$\mathcal{L} = \mathcal{L}_{cls} + \alpha\mathcal{L}_{aux} + \lambda\|\Theta\|_2, \tag{3}$$

where $\Theta = (\theta_{enc}, \theta_{cls}, \theta_{aux})$ denote all parameters of the proposed model, and the hyperparameter α balances the main classification task and the auxiliary word prediction task. λ is an l_2 regularization hyperparameter.

Algorithm 1 describes the detailed procedure for training the proposed model. The proposed model contains three parameter sets, namely, $\theta_{enc}, \theta_{cls}$, and θ_{aux}, which come from f_{enc}, f_{cls}, and f_{aux}, respectively. All these parameters are initialized randomly before training. In each epoch of the algorithm, a small set of tuples is sampled from the training set and corrupted with $\langle unk \rangle$ tokens. Once a corrupted training set is prepared, the entire network is trained jointly with the loss given in Eq. 3 by lines 8–9 until the training converges.

4 Network Implementation

The proposed model begins with a shared sentence encoder f_{enc}. The shared sentence encoder consists of an embedding layer and a Bi-LSTM layer. The embedding layer converts an input sentence x of L words into a sequence of word vectors in the form of the matrix $X = [\mathbf{v}_1^T, \ldots, \mathbf{v}_L^T]$. The Bi-LSTM layer encodes X into a contextual representation H by reflecting the left and right contexts against each d-dimensional embedding vector $\mathbf{h}_i$. That is,

$$\mathbf{h}_i = \overrightarrow{\mathbf{h}_i} \oplus \overleftarrow{\mathbf{h}}_i, \tag{4}$$

where $\oplus$ is the element-wise sum. Thus, the output of the sentence encoder is $H = [\mathbf{h}_1^T, \ldots, \mathbf{h}_L^T]$, where $H \in \mathbb{R}^{L \times d}$. This H is fed to both the sentence classification network and the UWP network.

We employ a CNN as the sentence classification network f_{cls} because the performance of CNNs in sentence classification has been demonstrated [18]. Most CNNs used for sentence classification generally apply one-dimensional (1d) convolution and 1d pooling operations [4]. However, Zhou et al. [18] introduced 2d convolution and 2d pooling operations to sentence classification and showed the effectiveness of the 2d operations in practice. Following the work of Zhou et al., we use a 2d convolutional layer and a 2d max pooling layer for f_{cls}. The convolution operation of the convolutional layer involves a 2D filter $m \in \mathbb{R}^{k \times d_m}$, which is applied to a window of k words and d_m feature dimensions. After the convolution operation is applied to H, the convolutional layer outputs a feature matrix $O_{conv} \in \mathbb{R}^{(l-k+1) \times (d-d_m+1)}$. The 2d max pooling operation is then applied to obtain a summarized feature map. With the pooling size $p \in \mathbb{R}^{p_1 \times p_2}$, the operation is applied to O_{conv} for extracting the maximum value features. By flattening the max-pooled feature map, a fixed-sized feature vector $\mathbf{o} \in \mathbb{R}^{\lfloor (l-k+1)/p1 \rfloor \cdot \lfloor (d-d_m+1)/p2 \rfloor}$ is obtained. Finally, $\mathbf{o}$ is fed to the classification layer, and the target class label is determined by

$$\mathbf{y} = softmax(W_y \cdot \mathbf{o} + \mathbf{b}_y), \tag{5}$$

where W_y and $\mathbf{b}_y$ denote a weight matrix and a bias vector of the classification layer, respectively.

The UWP network f_{aux} consists of a fully-connected layer that serves as a word prediction layer. f_{aux} takes H, the output of f_{enc}, as its input. Then, the network computes the probability distribution of the output words at each position i by

$$[\mathbf{w}_1, \ldots, \mathbf{w}_L] = softmax(W_{aux} \cdot H^T + \mathbf{b}_{aux}), \tag{6}$$

where W_{aux} and $\mathbf{b}_{aux}$ are the weight matrix and the bias vector, respectively. The output of the word prediction layer includes all L predicted words, but only U words at the same positions as the $\langle unk \rangle$ tokens are words of interest for the auxiliary task. To solve this problem, a one-hot masking vector $\mathbf{m}_i \in \mathbb{R}^L$ that indicates the position of a $\langle unk \rangle$ token at the i-th position is used to generate the final output of f_{aux} as follows.

$$\mathbf{w}_i = [\mathbf{w}_1, \ldots, \mathbf{w}_L] \cdot \mathbf{m}_i^T. \tag{7}$$

Note that this operation is executed for all U $\langle unk \rangle$ tokens.

5 Experiments

To demonstrate the effectiveness of the proposed model, we conducted a number of experiments on four widely used benchmark datasets for sentence classification.

Table 1. Summary statistics of datasets. c: number of classes, l: average sentence length, m: max sentence length, train/dev/test: train/development/test set size, vocab: vocabulary size in training data, unk_num: number of sentences that include at least 1 unknown word, and unk_max: max number of unknown word in a sentence.

Data	c	l	m	train	dev	test	vocab	unk_num	unk_max
SST-1	5	19	56	8544	1101	2210	16581	1240	9
SST-2	2	19	56	6920	872	1821	14830	1080	9
TREC-6	6	7	17	5452	-	500	8679	266	4
TREC-50	50	7	17	5452	-	500	8679	266	4

- **SST-1:** Stanford Sentiment Treebank was introduced by Socher et al. [14]. This dataset includes reviews with fine-grained labels (very negative, negative, neutral, positive, very positive).
- **SST-2:** This dataset is a coarse-grained version of SST-1. Thus, this dataset contains only the sentences with positive and negative labels from SST-1.
- **TREC-6:** A question classification dataset [9]. This dataset contains questions of six types, namely, abbreviation, description, entity, human, location, and numeric value.
- **TREC-50:** Another question classification dataset [9]. This dataset was created to classify a question into one of the fine-grained 50 question types.

Table 1 summarizes the statistics of the four benchmark datasets. As shown in this table, over 50% of the test sentences contain unseen words during training time. Thus, we can infer that the unseen words may significantly influence the classification performance of the proposed model.

The classification performance of the proposed model is compared with that of two baseline models. The first baseline model is a neural network with the same architecture as that of the proposed model, except for the auxiliary word predictor. Thus, this baseline did not encounter the $\langle unk \rangle$ token during training. Note that this baseline is a re-implemented version of BLSTM2DCNN [18], which exhibits the state-of-the-art performances on several sentence classification benchmarks. The second baseline model has the same architecture as the first baseline model. However, this baseline model is trained with the corrupted dataset D'. The injection of some noises into the training dataset has the effect of network regularization, which often improves performance.

5.1 Training Details and Hyperparameters

In the experiments, the Word2Vec embeddings trained by [11] were utilized as the pretrained word vectors. We initialized the vectors of the words that appeared only in the benchmark training datasets through random sampling from a uniform distribution in the range of $[-0.1, 0.1]$. The dimensions of the word embedding vector $\mathbf{v}_i$ and the contextual word vector $\mathbf{h}_i$ from the sentence encoder f_{enc} were set to 300. We used 100 convolutional filters with the window

Table 2. Classification results obtained with four sentence classification benchmarks. **BLSTM2DCNN:** the performance reported in [18]. **BLSTM2DCNN baseline:** a re-implemented version of BLSTM2DCNN. **BLSTM2DCNN baseline w/ noise injection:** a re-implemented version of BLSTM2DCNN with $\langle unk \rangle$ tokens injected into the training dataset.

Model	SST-1	SST-2	TREC-6	TREC-50
BLSTM2DCNN [18]	52.4	89.5	96.1	-
BLSTM2DCNN baseline	47.2	86.6	95.0	86.6
BLSTM2DCNN baseline w/ noise injection	47.5	87.1	94.1	86.0
Proposed model	**48.4**	**87.1**	**95.6**	**87.0**

size of (3,3). The 2D pooling size was set to (2,2). We performed mini-batch training with a batch size of 10. AdaDelta was used as an optimizer with the default learning rate of 0.1. For regularization, we employed the dropout operation with a rate of 0.5 for word embeddings, 0.2 for the Bi-LSTM layer, and 0.4 for the output of the pooling layer. Moreover, we imposed the l_2 penalty with the coefficient 10^{-5} over all parameters.

6 Results and Analysis

Table 2 shows the classification results obtained with four benchmark datasets. Unfortunately, we could not reproduce the exact performance of BLSTM2DCNN because the accuracy of the BLSTM2DCNN baseline is 1%–5% lower on three datasets than the corresponding performance reported in the original paper. As a result, the proposed model failed to exceed the result reported in the work of [18]. However, the proposed model consistently outperformed two baseline models on all four benchmark datasets. The proposed method achieved the best accuracies of 48.4% on SST-1, 95.6% on TREC-6, and 84.0% on TREC-50 relative to the baseline models.

It is known that training a neural network with noise-injected data regularizes the network, which may improve network performance. In our experiments, this was true for the tasks of SST-1 and SST-2 but not for the tasks of TREC-6 and TREC-50. More specifically, noise injection into the training data increased the accuracy of the BLSTM2DCNN baseline by 0.3% on SST-1 and by 0.5% on SST-2, while it decreased the accuracy of the BLSTM2DCNN baseline by 0.9% on TREC-6 and 0.6% on TREC-50. However, the proposed model yielded additional performance gains by introducing UWP as an auxiliary training task. This can be ascribed to the fact that the proposed auxiliary word predictor ensures that the sentence encoder produces not only more robust contextualized word representations but also well-approximated meaning representations for $\langle unk \rangle$ tokens.

We can understand the reason for performance improvement by observing the sentence representations produced by different models. Figure 2 shows two visu-

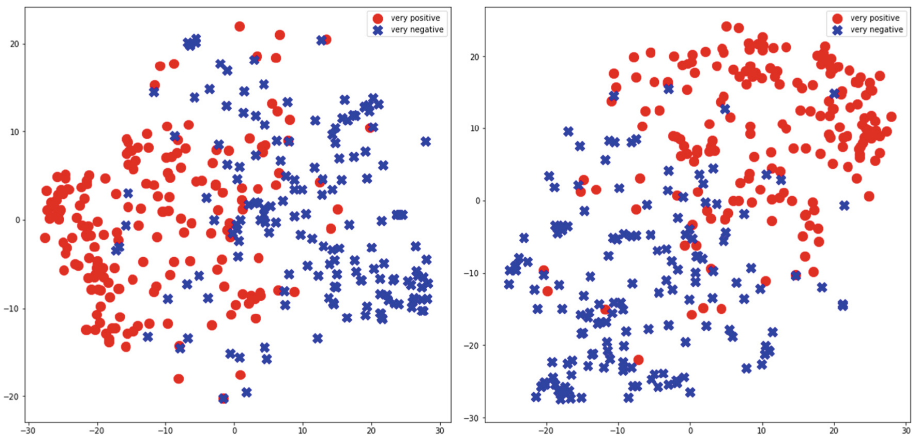

Fig. 2. Two visualizations of sentence representations by BLSTM2DCNN baseline (left) and proposed model (right) on SST-1 test dataset. All sentences in this figure contain $\langle unk \rangle$ tokens. (Color figure online)

Table 3. Performance comparison among the proposed model and BLSTM2DCNN baselines with different basic units on the benchmark datasets. Note that none of the models use any pretrained embeddings.

Model	SST-1	SST-2	TREC-6	TREC-50
BLSTM2DCNN using character	31.1 ± 0.5	63.7 ± 0.8	86.4 ± 0.5	76.4 ± 0.2
BLSTM2DCNN using subword2000	36.8 ± 0.4	76.2 ± 0.6	91.2 ± 0.2	80.2 ± 0.4
BLSTM2DCNN using subword4000	36.9 ± 1.1	76.0 ± 0.7	92.2 ± 0.6	81.9 ± 0.1
BLSTM2DCNN using word	39.7 ± 1.3	79.5 ± 0.9	93.0 ± 0.1	82.6 ± 0.5
Proposed model	$\mathbf{41.1 \pm 0.7}$	$\mathbf{81.3 \pm 1.0}$	$\mathbf{93.2 \pm 0.2}$	$\mathbf{83.7 \pm 0.3}$

alizations of sentence representations projected using T-SNE [10]. The left visualization in Fig. 2 shows sentence representations generated by the BLSTM2DCNN baseline, while the right one shows those generated by the proposed model. Because SST-1 is a difficult task, and the sentences in this figure contain more than one $\langle unk \rangle$ tokens, the red circles (very positive sentences) and blue crosses (very negative sentences) are jumbled in both figures. Nonetheless, in the right figure, the two areas of positive (top-right) and negative sentences (bottom-left) are more distinguishable than those in the left figure. These sentence representations were generated by summarizing $\mathbf{h}_i$'s in Eq. 4. Thus, the difference between the left and the right figures can be ascribed to the contextual representation power of the sentence encoder, which is jointly optimized for the UWP task.

Table 4. Examples of sentence classifications and unknown word predictions. All sentences are taken from SST-1 test dataset.

Sentence	Every good actor needs to do his or her own $\langle unk \rangle$	The film is surprisingly well-directed by brett $\langle unk \rangle$, who keeps things moving well – at least until the problematic third act
Baseline prediction	Positive	Negative
Proposed model prediction	Neutral	Positive
Original word of $\langle unk \rangle$	Hamlet	Ratner
Top-5 most probable words of $\langle unk \rangle$	Time . character way one	Character comedy way time director

Consequently, this different representation power inevitably contributes to the superior sentence classification performance of the proposed model.

Table 3 summarizes the classification results of the BLSTM2DCNN baselines with various basic units and those of the proposed model. The BLSTM2DCNNs using characters and subword units [6] eliminated the OOV problem by breaking down words into smaller units so that the vocabulary opened up. For the subword-level models, we limited the vocabulary size to 2,000 and 4,000. As can be seen, the results obtained with the BLSTM2DCNNs with the smaller units (character-level and two subword-level) are inferior to those achieved with the word-level BLSTM2DCNN over all benchmarks. These results indicate that the use of smaller units requires more complex and sophisticated architecture design. Finally, the proposed model achieved the best performance on all benchmarks because it replaced $\langle unk \rangle$ tokens with appropriate contextualized meaning representations.

Lastly, in Table 4, we introduce two example sentences that were correctly classified by the proposed model but misclassified by the baseline model. Each sentence in this table includes a $\langle unk \rangle$ token and the table shows the originals word of them as well as top-5 most likely words predicted by the proposed model. For the first sentence, the unknown word is '*hamlet*' which means a representative character or acting methods. Thus words like '*character*' and '*way*' predicted by the proposed model are quite appropriate for the $\langle unk \rangle$ token. Similarly, the actual word of $\langle unk \rangle$ token in the second sentence is '*ratner*' – the last name of the film director. Again, the proposed model correctly predicted the unknown word as the word '*director*.' Although, unknown words in both sentences are not very critical for classification, the proposed model was able to make right decisions through appropriately contextualized word representations as well as properly estimated $\langle unk \rangle$ tokens.

7 Conclusion

In this paper, we propose a neural sentence classifier with an auxiliary UWP. To improve the classification performance of the model during testing, the proposed model was trained to predict not only the class label of the given sentences but also unknown words by considering all other words as contextual information. As a result, the proposed model generated robust representations of unknown words. In addition, the proposed auxiliary task enhanced the robustness of the entire sentence representation, which improved the classification performance of the proposed model. In the experiments, the proposed model consistently outperformed two baselines in terms of the sentence classification performance on four benchmark datasets.

References

1. Cheng, H., Fang, H., Ostendorf, M.: Open-domain name error detection using a multi-task RNN. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 737–746 (2015)
2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding (2018). arXiv preprint arXiv:1810.04805
3. Iyyer, M., Enns, P., Boyd-Graber, J., Resnik, P.: Political ideology detection using recursive neural networks. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1113–1122 (2014)
4. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1746–1751 (2014)
5. Kudo, T.: Subword regularization: improving neural network translation models with multiple subword candidates. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 66–75 (2018)
6. Kudo, T., Richardson, J.: Sentencepiece: a simple and language independent subword tokenizer and detokenizer for neural text processing. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 66–71 (2018)
7. Le, H.T., Cerisara, C., Denis, A.: Do convolutional networks need to be deep for text classification? In: Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence (2018)
8. Lee, J., Cho, K., Hofmann, T.: Fully character-level neural machine translation without explicit segmentation. Trans. Assoc. Comput. Linguist. **5**, 365–378 (2017)
9. Li, X., Roth, D.: Learning question classifiers. In: Proceedings of the 19th International Conference on Computational Linguistics, vol. 1, pp. 1–7. Association for Computational Linguistics (2002)
10. van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. J. Mach. Learn. Res. **9**(Nov), 2579–2605 (2008)
11. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)

12. Ruder, S.: An overview of multi-task learning in deep neural networks (2017). arXiv preprint arXiv:1706.05098
13. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1715–1725 (2016)
14. Socher, R., et al.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1631–1642 (2013)
15. Wang, S., Manning, C.D.: Baselines and bigrams: simple, good sentiment and topic classification. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers, vol. 2, pp. 90–94. Association for Computational Linguistics (2012)
16. Yu, J., Jiang, J.: Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 236–246 (2016)
17. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Advances in Neural Information Processing Systems, pp. 649–657 (2015)
18. Zhou, P., Qi, Z., Zheng, S., Xu, J., Bao, H., Xu, B.: Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 3485–3495 (2016)
19. Zhou, Q., Wang, X., Dong, X.: Differentiated attentive representation learning for sentence classification. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence, pp. 4630–4636. AAAI Press (2018)