J Inf Process Syst, Vol.17, No.5, pp.867~878, October 2021
https://doi.org/10.3745/JIPS.02.0163

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Multi-channel Long Short-Term Memory with Domain Knowledge for Context Awareness and User Intention

Dan-Bi Cho*, Hyun-Young Lee*, and Seung-Shik Kang*

Abstract

In context awareness and user intention tasks, dataset construction is expensive because specific domain data are required. Although pretraining with a large corpus can effectively resolve the issue of lack of data, it ignores domain knowledge. Herein, we concentrate on data domain knowledge while addressing data scarcity and accordingly propose a multi-channel long short-term memory (LSTM). Because multi-channel LSTM integrates pretrained vectors such as task and general knowledge, it effectively prevents catastrophic forgetting between vectors of task and general knowledge to represent the context as a set of features. To evaluate the proposed model with reference to the baseline model, which is a single-channel LSTM, we performed two tasks: voice phishing with context awareness and movie review sentiment classification. The results verified that multi-channel LSTM outperforms single-channel LSTM in both tasks. We further experimented on different multi-channel LSTMs depending on the domain and data size of general knowledge in the model and confirmed that the effect of multi-channel LSTM integrating the two types of knowledge from downstream task data and raw data to overcome the lack of data.

Keywords

Context Awareness, Domain Adaptation, Multi-channel LSTM, User Intention

1. Introduction

As human-computer interaction (HCI) and artificial intelligence develop, the communication between humans and computers increases. Examples of such technologies include Amazon Alexa and Google Assistant. To enable proper interaction between humans and computers, many studies focused on context awareness, which aims to provide humans with optimized feedback by recognizing context and understanding the users' intentions [1-3]. Several research were conducted in natural language processing systems, such as chatbots and spam email filters, to identify the context and user needs for classification based on text characteristics. Tung and Soo [2] and Pashtan et al. [3] considered the users' context in terms of distance, economy, preference, travel route, and intention to recommend an optimized restaurant or travel course on the travel domain. Wang et al. [4] developed a system that recognizes the context of emails to automate subsequent user actions. Shim et al. [5] explored the context awareness task to achieve smooth communication between doctors and patients in a medical environment. Cho et al. [6] identified the political bias of presses by classifying the conservative and liberal tendencies in political news

Manuscript received January 7, 2021; first revision March 3, 2021; second revision April 23, 2021; accepted May 4, 2021.
Corresponding Author: Seung-Shik Kang (sskang@kookmin.ac.kr)
* Dept. of Computer Science, Kookmin University, Seoul, Korea (daanv319@kookmin.ac.kr, hyunyoung2@kookmin.ac.kr, sskang@kookmin.ac.kr)
Dan-Bi Cho and Hyun-Young Lee contributed equally to this work.

articles.

Because context awareness and user intention tasks are domain-dependent, constructing a labeled dataset suitable for specific domain knowledge is expensive, particularly in rare domains [7]. To complement the lack of labeled data, prior research pretrained a large corpus such as Wikipedia and a news article dataset [8,9]. Additionally, this approach played an important role in improving the performance of deep learning [10,11]. Kim [10] showed the effect of pretrained vectors using a convolutional neural network (CNN) and demonstrated the superior performance of the pretraining method over random initialization. Joshi et al. [11] investigated self-supervised learning using the span-level pretraining method by verifying the efficiency of the pretraining method. However, the distribution between downstream task data and a large raw corpus for the pretraining method differs because previous research did not consider the difference in distribution from downstream task data when pretraining a large raw corpus. Thus, when the downstream task data comprise a dialog or review dataset and the large raw corpus for the pretraining method is a Wikipedia or news article corpus, the data distributions and domain knowledge differ. Some noise occurs in this case owing to collisions in the syntactic and semantic meanings of domain knowledge from each dataset. For example, “run fast” has positive connotations in news articles regarding sports, whereas in reviews about electronic devices, “run fast” indicates that a battery depletes quickly, which is negative [7]. We argue that domain knowledge is an important factor in the classification task, particularly for context awareness and user intention tasks with dialog or review.

Domain adaptation has been researched to generalize the data distribution between the source and target domains to process data with multiple domains of knowledge [7,12,13]. Domain adaptation reduces collisions of syntactic and semantic meaning by fusing knowledge of multiple domains; however, it causes a catastrophic forgetting that makes it difficult to preserve the knowledge of multiple domains [14]. If the vector pretraining Wikipedia or the news article corpus is updated with dialog or review data on the downstream task, the pretrained vector causes catastrophic vector forgetting. Chen et al. [15] applied co-training, which requires two data perspectives for training, to the domain adaptation task by creating two mutually exclusive datasets and training each classifier to resolve catastrophic forgetting in domain adaptation. Co-training independently trains each model on the divided dataset and then shares their features, resulting in semi-supervised learning that effectively improves the compatibility of different domain knowledge [16].

Inspired by the concept of sharing different features in co-training, we devised a method for mutually integrating the knowledge from different domains between the downstream task data and raw corpus on the context awareness and user intention tasks. Herein, we define the two types of knowledge as task knowledge and general knowledge. The task knowledge is domain knowledge of downstream task data, and the pretrained vector using the downstream task data is called task vector. The general knowledge is raw corpus domain knowledge, not downstream task knowledge, and the general vector is a pretrained vector obtained using the raw corpus. We obtain two vectors, one for task knowledge and one for general knowledge, and we combine them to obtain a multi-channel vector. A multi-channel vector prevents two vectors from colliding while sharing features. Long short-term memory (LSTM) remembers the history of a previous time in sequential data using a gate mechanism, making it more effective than CNN in processing text data context [17]. Hence, we learn the multi-channel vector by applying it to the LSTM and refer to it as multi-channel LSTM. We expect that multi-channel LSTM is competent to solve the lack of labeled data and catastrophic forgetting because the multi-channel LSTM integrates the different

knowledge of domains between task vector and general vector. We further analyze our model depending on the domain and data size of general knowledge. Our research is structured as follows. In Section 2, we introduce an architecture of multi-channel LSTM. We construct the dataset for context awareness and user intention tasks in Section 3. In Section 4, the experimental results of multi-channel LSTM, obtained by comparing with a single-channel LSTM as a baseline, are presented. Finally, we summarize the achievements of our research and expected effectiveness.

2. Multi-channel LSTM with Domain Knowledge

The pretrained vector is applied to the classification task in one of two ways, depending on whether the vector is maintained or updated: static and dynamic [10]. Static method trains the model continuously while maintaining the syntactic and semantic meaning by the pretrained vector without an update. Dynamic method, conversely, fine-tunes the vector to the task by updating the vector during model training with backpropagation to determine the optimal weight by iteratively calculating the gradient of the loss function. Utilizing these two methods, we construct pretrained vectors including each knowledge.

To analyze the effect of domain knowledge on context awareness and user intention tasks, we propose a multi-channel LSTM, which organizes task knowledge and general knowledge into multiple channels as shown in Fig. 1. Before implementing the downstream task, we pretrain task knowledge to vector with downstream task data and general knowledge to vector with raw corpus individually, and then we represent them as continuous vectors in parallel. By mapping the input sentence $S = (s_1, s_2, \dots, s_n)$ to each pretrained vector according to the knowledge, we obtain two pretrained vectors: the task vector $T = (t_1, t_2, \dots, t_n)$ and general vector $G = (g_1, g_2, \dots, g_n)$, where t_i and g_i are pretrained vectors including each knowledge domain encoded in the i-th token s_i of the input sentence, respectively. If the vector-matrix T and G do not have the vector of token s_i, we encode the token s_i into the random vector of the same dimension as T and G, respectively.

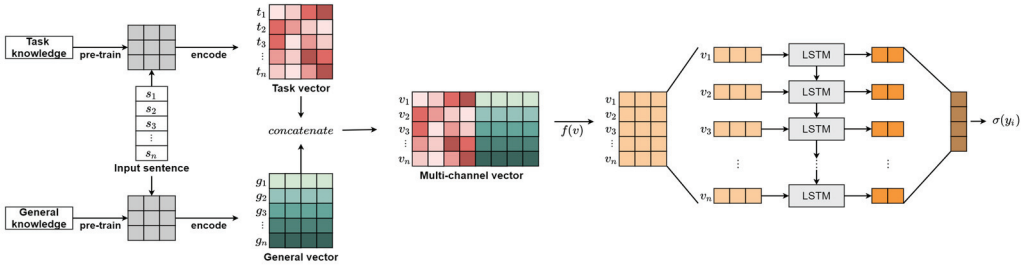

Fig. 1. Multi-channel LSTM model.

As the task vector aims to optimize the model by pretraining downstream task data, we apply the task vector to the model using the dynamic method. We expect that the dynamic task vector makes the model robust and stable for the downstream task. However, we use the static method to apply a general vector to the model because the general vector represents knowledge of raw corpus as a role of alleviating the lack of labeled data. Owing to the different domains between task knowledge and general knowledge, catastrophic forgetting will occur in the general vector if the task and general vectors are simultaneously

trained on the model using the dynamic method. We use the static method to apply a general vector to the model to alleviate the collision between task knowledge and general knowledge caused by catastrophic forgetting. We hypothesize that when a dynamic task vector and a static general vector are concatenated as in co-training, the model has a synergistic effect. Thus, we construct a multi-channel vector by concatenating the dynamic task vector $(t_1, t_2, \dots, t_n)$ and static general vector $(g_1, g_2, \dots, g_n)$ as $v_i = concatenate(t_i, g_i)$. The input sentence $S = (s_1, s_2, \dots, s_n)$ is represented as a multi-channel vector $(v_1, v_2, \dots, v_n)$. The sequential task vector $T = (t_1, t_2, \dots, t_n)$ and general vector $G = (g_1, g_2, \dots, g_n)$ are $T \in \mathbb{R}^{d \times n}$ and $G \in \mathbb{R}^{d \times n}$, respectively, and the multi-channel vector $V = (v_1, v_2, \dots, v_n)$ is represented as $(V \in \mathbb{R}^{2d \times n})$. To use the multi-channel vector as an input vector to LSTM, we perform the affine transformation: $f(v)$. The vector converted by affine transformation is used as the input for LSTM, which preserves a memory based on history while solving the long-term dependency problem via gate mechanism [17].

The affine transformation with 128 hidden units was used to reduce the dimension of the multi-channel vector, and the sequential vector information was summarized in the LSTM layer. At each iteration, the LSTM produced 64 vector units with *tanh* as an activation function, and the vectors were concatenated to form a one-dimensional array. The output layer presented the probability distribution using *softmax* $\sigma(y_i)$, where y_i is a predicted label for the *i*-th text. To minimize the model loss, cross-entropy was used in our model as a loss function.

$$Cross\ Entropy\ Loss = -\sum_{i=1}^{n} p_i \log \widehat{p_i} \tag{1}$$

Here, p_i and $\widehat{p_i}$ are the *i*-th actual probability and predicted probability, respectively. We optimized the model using the statistical gradient descent (SGD) with a learning rate of 0.001. The multi-channel LSTM was trained with a batch size of 32 and 100 epochs. In our experiments, we implemented models with TensorFlow version 2.2 and 32 GB NVidia Tesla V100 on docker images by Ubuntu 18.04.

3. Dataset and Setup

3.1 Dataset

To evaluate our multi-channel LSTM, we test two classification tasks: voice phishing with context-awareness and movie review sentiment classification. For voice phishing with context awareness, our goal is to identify whether a text is in the context of voice phishing. To classify the context of voice phishing, we constructed a dataset using two different datasets. One indicates the context of voice phishing, and the other has similar but non-voice phishing. For the voice phishing dataset, we used 315 voice phishing cases in the conversation between fraudsters and victims, which were released by the financial supervisory service (http://phishing-keeper.fss.or.kr/fss/vstop/avoid/this_voice_l.jsp). We examined the conversation to see whether it suggested the context of voice phishing. In the voice phishing dataset, the fraudster poses as a financial institution, such as a bank, and requests personal information about financial or economic details from the victim. The victims respond to the fraudster's questions. However, because victims' responses are sometimes brief, such as "yes" or "no," and contain personal

information, the texts mentioned by the victims do not imply knowledge of voice phishing. Therefore, we extracted texts mentioned by fraudsters, which imply information about voice phishing based on the types of questions and contain less personal information. In the extracted texts, we replaced personal information, such as names and account numbers, as the special tokens and separated texts into sentence units. The result yields a dataset of a total of 8,171 sentences of voice phishing.

We also needed a dataset representing the non-voice phishing for the classification task. To construct the non-voice phishing dataset, we used a dialog dataset belonging to the "real estate" category on AI-Hub (https://www.aihub.or.kr/aidata/ 85). It is a dialog of buyer's questions and sellers' answers about shopping malls, real estate, and land. Therefore, the non-voice phishing dataset contains the same financial and economic topics as the voice phishing dataset. For example, "How much is the deposit?" We extracted the buyers' texts, which are questions, from the non-voice phishing dataset, just as the fraudsters' texts were extracted from the voice phishing dataset. Consequently, the non-voice phishing dataset contained 4,940 sentences.

When training a model with unbalanced data, the model is biased toward data labeled on voice phishing. It is because the model learns more features of voice phishing (8,171 sentences), whose data size is larger than that of non-voice phishing (4,940 sentences). The size of the voice phishing and non-voice phishing datasets should be equal to prevent data-based bias. Consequently, we randomly selected 4,940 sentences from the voice phishing dataset's 8,171 sentences to match the data size with the non-voice phishing dataset's 4,940 sentences. Herein, the extracted sentences in the voice phishing and non-voice phishing datasets are referred to as a VP dataset, and Table 1 shows the data distribution of voice phishing and non-voice phishing, specifically the number of sentences, word tokens, and vocabulary size. The VP dataset contains 9,880 sentences in total, and we split the VP dataset at a ratio of 8:2 to construct training and test sets of 7,904 and 1,976 sentences, respectively.

To conduct the user intention task related to context awareness, we experimented on the sentiment classification task, which traditionally has been studied in natural language processing [17,19]. For the sentiment classification task, we used the NSMC dataset (https://github.com/e9t/nsmc), which is publicly available. The NSMC dataset contains positive and negative classes for audience reviews and publishes a training set (150,000 sentences) and a test set (50,000 sentences). We evaluated our model using the NSMC dataset for the movie review sentiment classification task to identify the user intention.

Table 1. Distribution of the VP dataset

Data type	Number of sentences	Number of word tokens	Vocabulary size
Voice phishing	4,940	49,595	12,902
Non-voice phishing	4,940	24,037	6,230

3.2 Setup for Pretrained Vector

Word2Vec is a prediction-based embedding model that learns the syntactic and semantic meanings of words to represent words in a continuous vector space, such as CBOW or skip-gram [19,20]. During the pretraining process, we used skip-gram, which predicts the surrounding words by inputting the vector of the central word, to represent text data as a continuous vector. Prior research that considered the characteristics of the Korean language, which is an agglutinative language, conducted morpheme

embedding to learn text in the morpheme unit [6,21]. In the embedding task for the Korean language, it was demonstrated that morpheme embedding outperforms word embedding. Therefore, we tested our model with each embedding method (i.e., word and morpheme embedding) and chose KLT2000 (we executed index2018.exe on KLT2000, https://cafe.naver.com/nlpkang) and Okt for high-speed morphological analysis for morpheme embedding. After representing the text as a token sequence for each token type, we trained the multi-channel LSTM with the vector, which is pretrained by applying the token sequence represented in each token type to the skip-gram.

Because the pretrained vector learns contextual information in advance, there is a difference in performance based on how much data are used for pretraining and how many iterations of training are performed. Consequently, we pretrained the VP dataset for the voice phishing task and the NSMC dataset for the movie review task in hyperparameter settings with iteration 300 and min-count 1. We used the training set from each downstream task, particularly when representing the pretrained vector with task knowledge (i.e., 7,904 in VP and 150,000 in NSMC). By setting different hyperparameters according to data size, we pretrained the KCC dataset (http://nlp.kookmin.ac.kr/kcc/), which is a large corpus for general knowledge as full-set KCC, with iteration 10 and a min-count of 5. Unlike the full-set KCC, however, we pretrained the reduced KCC, which is reduced to the same size as downstream task data, with the same hyperparameters as task knowledge using the training set of each downstream task (iteration 300 and min-count 1).

4. Experiments and Results

We assumed that the multi-channel LSTM can preserve each knowledge while preventing catastrophic forgetting; thus, we tested our model for two classification tasks: voice phishing and movie review tasks. Before evaluating the multi-channel LSTM, we performed single-channel LSTM as a baseline for comparison. The single-channel LSTM inputs input-only task vector, which is trained using the downstream task data, in LSTM without a general vector. In Section 4.1, we explore the single-channel LSTM experiments as a baseline model. As shown in GPT-2, introduced by Radford et al. [22], we expected a large corpus to improve the performance; thus, we test the multi-channel LSTM depending on the data size of general knowledge with the difference of domain knowledge in Section 4.2.

4.1 Single-Channel LSTM for Baseline

To verify the efficiency of the pretrained vector and compare the performance when applying static or dynamic pretrained vectors to the model, we tested the end-to-end learning and the pretraining methods for single-channel LSTM. In end-to-end learning, the model optimizes the weight of the network from input to output instead of training the data in advance, which allows the model to be trained by representing the downstream task data with a randomly initialized vector. In contrast, pretraining methods train vectors in advance and the vector is mapped to the input text. In single-channel LSTM, the pretrained vector is applied to LSTM in the role of the task vector, which is a single-channel vector.

Instead of representing the pretrained vector using downstream task data, we trained the vector using the KCC dataset to compare the performance according to the domain in the single-channel LSTM with

pretraining. Because the single-channel vector has a vector obtained through pretraining on the KCC dataset, the model learns the knowledge of the KCC dataset. We tested the model on two different KCC datasets: reduced KCC and full-set KCC. Reduced KCC represents a vector with data reduced to the same size as that of the downstream task data to compare only the difference in domain knowledge, i.e., for the voice phishing task and the movie review task, we pretrained vectors by randomly extracting 7,904 sentences and 150,000 sentences in full-set KCC, respectively. Full-set KCC represents a vector by increasing the data size, which includes the large corpus KCC dataset. Thus, full-set KCC differs from downstream task data in terms of the domain as well as data size. In the single-channel LSTM with pretraining, we evaluated three pretrained vectors, namely, the downstream task data, reduced KCC, and full-set KCC, using the static and dynamic pretraining methods.

4.2 Multi-channel LSTM for Text Classification

For voice phishing context awareness and movie review sentiment classification tasks, we fixed the task knowledge as the VP and NSMC datasets, respectively, to represent the task vector of multi-channel LSTM. Because the goal of task knowledge is to make the model more robust, optimally representing the contextual information in the downstream task data is critical. General knowledge, unlike task knowledge, conveys contextual information of the raw corpus used as background information for the model; hence, performance varies depending on which domain knowledge is used to represent the vector. Our experiment considers two cases of general domain knowledge:

- Multi-channel LSTM on the same domain knowledge: The general knowledge domain is equal to the task knowledge domain.
- Multi-channel LSTM on different domain knowledge: The general knowledge domain is different from the task knowledge domain.

First, the multi-channel LSTM on the same domain knowledge, called multi-SDK, represents the domain of general knowledge as VP or NSMC in each task, as the domain of task knowledge is represented as downstream task data, i.e., VP or NSMC. The data distribution of the task and general knowledge have the same domain as the downstream task data. Although these data have the same domain knowledge, the task knowledge is applied by dynamically updating the pretrained vector in multi-channel LSTM, whereas the general knowledge is applied by statically maintaining the pretrained vector. Second, multi-DDK, or multi-channel LSTM on different domain knowledge, represents general knowledge as a separate domain from task knowledge. To compare the efficiency following the domain knowledge, we used a reduced KCC dataset as general knowledge and restricted variables, such as data size and parameters, except for the general knowledge domain, in multi-DDK.

Because the data size affects performance, we reduced the data size of general knowledge in the multi-DDK to produce the same conditions as those of general knowledge in multi-SDK. This means that the general vector is represented as reduced KCC in the multi-channel LSTM, as the task vector of single-channel LSTM is represented using reduced KCC. Given that a large corpus has been used to solve the lack of data in previous research, we evaluated the performance of multi-DDK with a large corpus to examine the effect of data size. We investigated the multi-DDK with a large corpus, using the full-set KCC as a general vector. The reason for using multi-DDK, not multi-SDK, in the experiment according to the data size of general knowledge is that it is difficult to construct a large corpus of downstream task data.

4.3 Results and Analysis

Before analyzing the experimental results of multi-channel LSTM, we analyzed the experimental results of the single-channel LSTM, which is the baseline model, to support our model. We tested the single-channel LSTM with end-to-end learning as well as static and dynamic pretraining methods on VP and NSMC, which are downstream task data. Table 2 shows the experimental results, and the accuracy is compared by token type (e.g., word and morpheme with KLT2000 and Okt). Using the baseline model, end-to-end learning in the voice phishing task showed the best accuracy of 82.69% in Okt, whereas the accuracy of pretraining methods with VP was 92.71% and 93.02% for static and dynamic methods, respectively, using Okt. Even higher accuracy was achieved with KLT2000. The movie review sentiment classification task also showed 83.82% accuracy in end-to-end learning and achieved 85.01% and 85% for the static and dynamic methods, respectively, with NSMC under the conditions of Okt. For both tasks, we found that the performance of static and dynamic methods is higher than that of end-to-end learning, which supports the assertion that the pretrained vector improves model performance.

Table 2. Accuracy (%) of single-channel LSTM

Task	Dataset for pretraining	Vector representation method	Word	KLT2000	Okt
Voice phishing context-awareness	VP	End-to-end	74.39	71.61	82.69
		Static	90.49	93.22	92.71
		Dynamic	90.89	93.62	93.02
	Reduced KCC	Static	85.17	89.17	89.22
		Dynamic	82.99	90.89	89.27
	Full-set KCC	Static	92.21	94.03	94.18
		Dynamic	92.11	94.89	94.84
Movie review sentiment classification	NSMC	End-to-end	79.30	82.88	83.82
		Static	81.16	84.38	85.01
		Dynamic	81.61	84.44	85.00
	Reduced KCC	Static	72.76	78.03	80.01
		Dynamic	79.10	82.59	84.38
	Full-set KCC	Static	77.59	82.54	84.00
		Dynamic	79.93	83.17	84.93

We confirmed that the performances of the static and dynamic methods are superior to that of end-to-end learning using a baseline model. We also determined two key points from the results presented in Table 2. First, representing a pretrained vector using downstream task data (VP and NSMC) achieves higher accuracy than representing a pretrained vector using other data, such as the KCC dataset, under the same condition of controlling variables other than the domain in single-channel LSTM. In the voice phishing task, VP demonstrated 92.71% and 93.02% accuracy for static and dynamic methods, respectively, whereas the results of the experiment with reduced KCC, which is the KCC dataset reduced to the same data size as VP, demonstrated low accuracies of 89.22% and 89.27% for static and dynamic methods under Okt. Similarly, in the movie review task, the reduced KCC, which is the same size as NSMC in this case, performed worse than NSMC under Okt conditions. These findings support the importance of domain knowledge by demonstrating that when a single-channel vector is represented in

the same domain knowledge as the downstream task data, performance improves.

Second, we found that the dynamic method of updating a pretrained vector improves performance, rather than the static method that maintains the pretrained vector when applied to the model. We compared the difference in performance depending on whether the pretrained vector is maintained or updated for the two tasks, i.e., we applied a single-channel vector for the static and dynamic method to LSTM. VP, reduced KCC, and full-set KCC achieved higher accuracy in the dynamic method for the voice phishing task, and the same result was obtained for the movie review task, as listed in Table 2. These results indicate that the model is fine-tuned to fit the task by dynamically updating the pretrained vector in the model. We verified the efficiency of the pretrained vector, the importance of domain knowledge, and fine-tuning effect of the dynamic method through single-channel LSTM as a baseline model.

We proposed the multi-channel LSTM and evaluated the performance of the multi-channel LSTM, which is composed of pretrained vectors with two types of knowledge, as two models using the results obtained through the baseline model (e.g., multi-SDK and multi-DDK). We confirmed that multi-channel LSTM outperforms single-channel LSTM, as shown in Table 3. In the voice phishing task, the single-channel LSTM using VP with the dynamic method achieved 90.89%, 93.62%, and 93.02% accuracy in word, KLT2000, and Okt, respectively. In contrast, the multi-SDK, which is trained as downstream task data for both the task and general vectors, showed 92.4%, 95.44%, and 96.05% in the word, KLT2000, and Okt, respectively. In the movie review task, similarly, single-channel LSTM using NSMC with dynamic method showed 81.61%, 84.44%, and 85% accuracy in word, KLT2000, and Okt, respectively, whereas the multi-SDK showed 81.54%, 84.90%, and 85.60%. Thus, the multi-channel LSTM is better than single-channel LSTM using the dynamic method.

Table 3. Accuracy (%) of multi-channel LSTM

	Voice phishing with context-awareness			**Movie review sentiment classification**		
	Word	**KLT2000**	**Okt**	**Word**	**KLT2000**	**Okt**
Multi-SDK	92.40	95.44	96.05	81.54	84.90	85.60
Multi-DDK	92.61	95.39	95.95	80.98	84.43	85.28
Multi-DDK with a large corpus	93.31	96.20	95.95	81.64	84.93	86.04

In multi-channel LSTM, we confirmed the importance of domain knowledge through the multi-SDK and multi-DDK. As shown in Table 3, when embedding was represented using Okt, both tasks were evaluated to have the highest performance, and we found that multi-SDK outperforms multi-DDK. In the voice phishing task, the multi-SDK and multi-DDK using Okt achieved 96.05% and 95.95% accuracy, respectively, whereas in the movie review task, the multi-SDK and multi-DDK achieved 85.60% and 85.28% accuracy. Although the difference in performance between the two multi-channel LSTM models was minor, it led to the conclusion that the domain is an important factor for classification tasks. That is, we confirmed that the multi-channel LSTM outperformed the single-channel LSTM in the results of the two tasks and that the multi-SDK outperformed the multi-DDK.

We analyzed the effect of data size on the single-channel LSTM. As shown in Table 2, the single-channel LSTM model using full-set KCC outperforms the model using reduced KCC in both tasks. Based on these results, we evaluated the multi-channel LSTM by using full-set KCC as the general knowledge of multi-DDK instead of reduced KCC. Therefore, multi-DDK with a large corpus is used to train LSTM

by concatenating a task vector from downstream task data and a general vector from full-set KCC. When general knowledge performance was compared based on data size, as shown in Table 3, multi-DDK achieved 92.61%, 95.39%, and 95.95% in the word, KLT2000, and Okt tasks, respectively, whereas multi-DDK with a large corpus achieved 93.31%, 96.20%, and 95.95% for the voice phishing task. For the movie review task, similarly, multi-DDK achieved 80.98%, 84.43%, and 85.28% in the word, KLT2000, and Okt, respectively, whereas multi-DDK with a large corpus achieved 81.64%, 84.93%, and 86.04%. That is, the multi-DDK with a large corpus using full-set KCC showed better performance than multi-DDK using reduced KCC. Comparing the multi-SDK and multi-DDK with a large corpus, multi-DDK with a large corpus showed better performance than multi-SDK. However, between both models, the accuracy of multi-DDK with a large corpus using word and KLT2000 was higher than that of multi-SDK; in contrast, in Okt, multi-SDK showed 96.05% higher performance than multi-DDK with a large corpus. Even in the movie review task, the multi-DDK with a large corpus exhibited higher accuracy when compared with multi-SDK in all token types; however, no significant difference was found. From these results, we know that multi-SDK is a competitive model, even though multi-DDK with a large corpus has the highest accuracy among the multi-channel LSTM models (e.g., multi-SDK, multi-DDK, and multi-DDK with a large corpus).

After comparing it with the single-channel LSTM, we proposed a multi-channel LSTM model and analyzed the performance of three types, multi-SDK, multi-DDK, and multi-DDK with a large corpus, for two tasks. By integrating vectors of two knowledge, multi-channel LSTM outperformed single-channel LSTM in our work, and we maintain that the multi-channel LSTM is resistant to catastrophic forgetting. Among the multi-channel LSTM models compared by data domain and data size, multi-DDK with a large corpus showed the best performance at 96.20% and multi-SDK achieved competitive performance.

5. Conclusion

To resolve the lack of data and catastrophic forgetting, we introduced multi-channel LSTM for context awareness and user intention tasks. By integrating the knowledge domains between downstream task data and raw corpus, the multi-channel LSTM is effective at reducing vector confusion. The multi-channel LSTM outperforms the single-channel LSTM in both tasks with voice phishing and movie review datasets constructed while avoiding data-based bias. We discovered three points in the experiment with baseline, single-channel LSTM: (1) The pretraining method is effective for representing a vector, (2) the dynamic method outperforms the static method, and (3) pretraining performance improves as data size increases.

We tested the multi-channel LSTM such as multi-SDK, multi-DDK, and multi-DDK with a large corpus based on the results of single-channel LSTM. The results of multi-channel LSTM demonstrated that multi-SDK outperforms multi-DDK and multi-DDK with a large corpus outperforms multi-SDK. We discovered that performance improves (1) when the domains of downstream task data and raw corpus are the same, and (2) when the data size for pretraining is large. These findings emphasize the importance of addressing the issue of disparities in data distribution between downstream task data and raw corpus data. We expect that our research contributes not only to the classification task but also to other NLP tasks such as sequence labeling tasks, machine reading comprehension, and translation.

Acknowledgement

This work was supported by the Ministry of the Republic of Korea and the National Research Foundation of Korea (No. NRF-2019S1A5A2A03046571).

References

[1] A. Gupta, P. Zhang, G. Lalwani, and M. Diab, "CASA-NLU: context-aware self-attentive natural language understanding for task-oriented chatbots," 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), Hong Kong, 2019, pp. 1285-1290.

[2] H. W. Tung and V. W. Soo, "A personalized restaurant recommender agent for mobile e-service," in *Proceedings of IEEE International Conference on e-Technology, e-Commerce and e-Service*, Taipei, Taiwan, 2004, pp. 259-262.

[3] A. Pashtan, A. Heusser, and P. Scheuermann, "Personal service areas for mobile web applications," *IEEE Internet Computing*, vol. 8, no. 6, pp. 34-39, 2004.

[4] W. Wang, S. Hosseini, A. H. Awadallah, P. N. Bennett, and C. Quirk, "Context-aware intent identification in email conversations," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, Paris, France, 2019, pp. 585-594.

[5] C. B. Shim, Y. W. Shin, and B. R. Park, "An Implementation of Context-Awareness Support System based on voice Service for Medical Environments," *Journal of the Korea Society of Computer and Information*, vol. 10, no. 4, pp. 29-36, 2005.

[6] D. B. Cho, H. Y. Lee, J. H. Park, and S. S. Kang, "Automatic bias classification of political news articles by using morpheme embedding and SVM," in *Proceedings of the Korea Information Processing Society Conference*, Virtual events, 2020, pp. 451-454.

[7] X. Chen and C. Cardie, "Multinomial adversarial networks for multi-domain text classification," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, New Orleans, LA, 2018, pp. 1226-1240.

[8] P. Wang and C. Domeniconi, "Building semantic kernels for text classification using Wikipedia," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Las Vegas, NV, 2008, pp. 713-721.

[9] D. Cho, H. Lee, and S. Kang, "Voice phishing with context-awareness using large corpus," in *Proceedings of the Korea Software Congress*, Pyeongchang, Korea, 2020, pp. 310-312.

[10] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 2014, pp. 1746-1751.

[11] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, "SpanBERT: improving pre-training by representing and predicting spans," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 64-77, 2020.

[12] S. Li and C. Zong, "Multi-domain adaptation for sentiment classification: using multiple classifier combining methods," in *Proceedings of 2008 International Conference on Natural Language Processing and Knowledge Engineering*, Beijing, China, 2008, pp. 1-8.

[13] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: a deep learning approach," in *Proceedings of the 28th International Conference on Machine Learning (ICML)*, Bellevue, WA, 2011, pp. 513-520.

[14] H. E. Kim, S. Kim, and J. Lee, "Keep and learn: continual learning by constraining the latent space for knowledge preservation in neural networks," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2008*. Cham, Switzerland: Springer, 2008, pp. 520-528.

[15] M. Chen, K. Q. Weinberger, and J. Blitzer, "Co-training for domain adaptation," *Advances in Neural Information Processing Systems*, vol. 24, pp. 2456-2464, 2011.

[16] K. Nigam and R. Ghani, "Analyzing the effectiveness and applicability of co-training," in *Proceedings of the 9th International Conference on Information and Knowledge Management*, McLean, VA, 2000, pp. 86-93.

[17] 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*, Austin, TX, 2016, pp. 606-615.

[18] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin, "Learning sentiment-specific word embedding for twitter sentiment classification," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, MD, 2014, pp. 1555-1565.

[19] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in Neural Information Processing Systems*, vol. 26, pp. 3111-3119, 2013.

[20] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proceedings of the 1st International Conference on Learning Representations (ICLR)*, Scottsdale, AZ, 2013.

[21] D. Lee, Y. Lim, and T. T. Kwon, "Morpheme-based efficient Korean word embedding," *Journal of KIISE*, vol. 45, no. 5, pp. 444-450, 2018.

[22] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019 [Online]. Available: https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf.

Dan-Bi Cho **https://orcid.org/0000-0003-1611-3192**

She received her B.A. degree in bigdata analytics business statistics from Kookmin University in 2020. She is an M.S. candidate in Computer Science at Kookmin University since 2020. Her research interests include natural language processing, machine learning, deep learning, and text mining.

Hyun-Young Lee **https://orcid.org/0000-0003-2553-6576**

He received his B.S. degree in Computer Engineering and M.E. degree in Computer Science from Kookmin University in 2016 and 2019, respectively. He is a Ph.D. candidate in Computer Science at Kookmin University since 2019. His research interests include natural language processing, machine learning, deep learning, and recommendation system.

Seung-Shik Kang **https://orcid.org/0000-0003-3318-6326**

He received his B.S. degree in Computer Science from Seoul National University in 1986 and M.S. and Ph.D. degrees in Computer Science from the same University in 1988 and 1993, respectively. He worked for Hansung University as an associate professor from 1994 to 2001. Currently, he is working for Kookmin University as a full-time professor. His research interests include natural language processing, information retrieval, text mining, big data processing, and machine learning.