IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

New RNN Activation Technique for Deeper Networks: LSTCM cells

SOO-HAN KANG[1] and JI-HYEONG HAN[2]
[1]Seoul National University of Science and Technology (e-mail: shkang@seoultech.ac.kr)
[2]Seoul National University of Science and Technology (e-mail: jhhan@seoultech.ac.kr)

Corresponding author: Ji-Hyeong Han (e-mail: jhhan@seoultech.ac.kr).

This work was supported by a grant from the National Research Foundation of Korea (NRF) funded by the Korean government (MSIT) (No. 2018R1C1B6007230)

ABSTRACT Long short-term memory (LSTM) has shown good performance when used with sequential data, but gradient vanishing or exploding problem can arise, especially when using deeper layers to solve complex problems. Thus, in this paper, we propose a new LSTM cell termed long short-time complex memory (LSTCM) that applies an activation function to the cell state instead of a hidden state for better convergence in deep layers. Moreover, we propose a sinusoidal function as an activation function for LSTM and the proposed LSTCM instead of a hyperbolic tangent activation function. The performance capabilities of the proposed LSTCM cell and the sinusoidal activation function are demonstrated through experiments on various natural language benchmark datasets, in this case the Penn Tree-bank, IWSLT 2015 English-Vietnamese, and WMT 2014 English-German datasets.

INDEX TERMS Long Short-Term Memory, Language Modeling, Neural Machine Translation

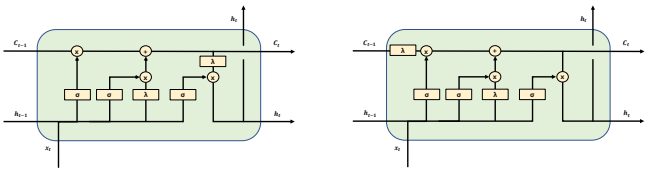
I. INTRODUCTION

RECENTLY, deep learning approaches including feed-forward networks, convolution neural networks (CNNs), and recurrent neural networks (RNNs) have shown good performance in many fields. RNNs perform especially well when applied to sequential problems such as video description [1], [2], speech recognition [3], [4], neural machine translation [5]–[7], sentiment classification from text [8], and detection from multidimensional data [9]. A RNN is a recurrent network which uses the hidden state of the previous time step as input for the current time step t as follows:

$$h_t = \lambda(W_x x_t + W_h h_{t-1} + b) \tag{1}$$

where λ is the activation function; x_t and h_t are the input and hidden state at time step t; and W_x, W_h, and b are trainable weights.

Despite of the good performance of RNNs, real-world problems are becoming more complicated, meaning that plain vanilla RNNs cannot sufficiently solve them. The basic approach to solving complex problems with deep learning is to create a deeper network or a more complex network. This is also true, in RNN research; i.e., the stacking of multiple recurrent layers or the use of more complex cells, such as long short-term memory (LSTM) [10], gated recurrent unit (GRU) [11] and neural architecture search (NAS) [12] cells.



(a) LSTM (b) LSTCM

FIGURE 1. LSTM and the proposed LSTCM. LSTM applies an activation function to the hidden state. On the other hand, the proposed LSTCM applies the activation function to the cell state. Thus, the proposed LSTCM cell can maintain the same level of the complexity in time steps and while also transferring larger gradient to the next layer.

Both LSTM and GRU, unlike a vanilla RNN, use a gate based on sigmoid function to pass information to the next time step. Due to the gate concept, LSTM and GRU can represent complex cells and solve the gradient vanishing problem. However, when we stack multiple layers using LSTM or GRU to solve complex problems, the gradient vanishing problem arises. This occurs because the weights are multiplied iteratively when we train RNNs and the activation functions used in RNNs are usually hyperbolic tangent and sigmoid functions which disturb the learning process given that the derivatives of the hyperbolic tangent and sigmoid functions are small.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3040405, IEEE Access

To solve the gradient vanishing problem in RNNs, several approaches have been developed. Gulcehre *et al.* proposed hard-sigmoid and hard-tanh activation functions to reduce gradient vanishing in RNN [13]. Le *et al.* and Li *et al.* proposed IndRNN, which uses ReLU instead of a hyperbolic tangent as an activation function in RNNs [14], [15]. Le *et al.* proposed IRNN, which uses an identity matrix and scaled weight initialization to apply ReLU to RNNs [14]. Li *et al.* proposed the independently RNN (IndRNN), which uses the Hadamard product for independently learned neurons and which also enables ReLU as an activation function [15]. Gonnet and Deselaers proposed independently long short-term memory (ILSTM) which applies the IndRNN concept to LSTM, resulting in better performance while also avoiding the overfitting issue [16]. LSTM is expressed as follows:

$$\begin{aligned} f_t &= \sigma(W_x^f x_t + W_h^f h_{t-1} + b^f) \\ i_t &= \sigma(W_x^i x_t + W_h^i h_{t-1} + b^i) \\ o_t &= \sigma(W_x^o x_t + W_h^o h_{t-1} + b^o) \\ j_t &= \lambda_j(W_x^j x_t + W_h^j h_{t-1} + b^j) \\ c_t &= c_{t-1} \odot f_t + i_t \odot j_t \\ h_t &= \lambda_h(c_t) \odot o_t \end{aligned} \tag{2}$$

where x_t and h_t are correspondingly the input and hidden state at time step t; $\odot$ represents the Hadamard product; σ, λ_j, and λ_h are the sigmoid function and activation functions used to calculate j_t and h_t, respectively; and W and b are learned parameters of LSTM cells. However, despite the superior performance of ILSTM, the gradient vanishing problem can also occur in this case because it uses a hyperbolic tangent for the activation functions of λ_j and λ_h.

In this paper, we propose two different novel activation techniques for RNNs. In the first, we newly locate the activation function λ_c for the cell state instead of λ_h for the hidden state, as shown in Figure 1. The newly applied position for the activation function makes the proposed cell transfer larger gradients to the next layer and retain the complexity in time steps. Thus, the proposed cell is referred to as long short-time complex memory (LSTCM). With this new activation technique, the proposed LSTCM cell reduces the gradient vanishing problem in the layers, thus creating and training a deeper network for complex problems. The second technique is the novel application of a sinusoidal function as an activation function for RNNs. Sitzmann *et al.* proposed a sinusoidal function as an activation function for CNNs with well initialized weights in implicit neural representations such as natural images and 3D shapes [17]. Thus, we apply the sinusoidal function as an activation function for LSTM and the proposed LSTCM instead of a hyperbolic tangent. Experiments on various tasks demonstrated that the proposed LSTCM cell outperforms the LSTM cell in a deeper network. Moreover, when using the sinusoidal function as an activation function for LSTM and LSTCM cells, they outperform the traditional hyperbolic tangent activation function.

This paper is organized as follows. Section II proposes the new activation techniques for RNNs including LSTCM cells and a sinusoidal activation function. In Section III, the experiments conducted here, in this case a language modeling task and a machine translation task, are described and the results are discussed. Finally, concluding remarks follow in Section IV.

II. METHODS

In this section, we propose novel activation techniques for LSTM, including the LSTCM cell and sinusoidal activation function. First, we explain backpropagation through time in LSTM, which is the basis of the proposed LSTCM cell. Then, the proposed LSTCM cell is explained in details. We also explain how to apply the sinusoidal function as an activation function for LSTM and LSTCM instead of the hyperbolic tangent function.

A. BACKPROPAGATION THROUGH TIME IN LSTM

Despite the fact that LSTM has long been studied, the vanishing gradient problem remains associated with it. Equation (3) expressed backpropagation through time (BPTT) of LSTM at time step t. In Equation (3), δx refers to $\partial L/\partial x$ where L is the loss function; Δ is the cumulative gradient from the layers above calculated from all gradients of each state vector; λ' and σ' are derivations of activation functions; and $\{\bar{o_t}, \bar{i_t}, \bar{j_t}, \bar{f_t}\}$ are state vectors before the activation functions. As shown in Equation (3), the gradients are calculated through multiplication with λ', which is less than 1, recurrently in time steps until $t = 0$, at which point the gradient vanishing problem can occur. Additionally, all RNN architectures stack multiple cells as layers to create a deep network [18], implying that $h_{n,t}$, which is the hidden state of the n-th layer, becomes $x_{n+1,t}$, which is the input to the $n+1$-th layer and $h_{n,t}$ is calculated using $c_{n,t}$ after the activation function λ_h. Thus, when we backpropagate the gradients in LSTM, the small value λ' is propagated through the layers and the gradient decaying process accelerates.

$$\begin{aligned} \delta h_t =& \Delta + W_h^j \delta j_{t+1} + W_h^o \delta o_{t+1} \\ &+ W_h^i \delta i_{t+1} + W_h^f \delta f_{t+1} \\ \delta c_t =& \delta h_t \odot o_t \odot \lambda_h'(c_t) + \delta c_{t+1} \odot f_{t+1} \\ \delta j_t =& \delta c_t \odot i_t \odot \lambda_j'(\bar{j_t}) \\ \delta o_t =& \delta h_t \odot \lambda_h(c_t) \odot \sigma'(\bar{o_t}) \\ \delta i_t =& \delta c_t \odot j_t \odot \sigma'(\bar{i_t}) \\ \delta f_t =& \delta c_t \odot c_{t-1} \odot \sigma'(\bar{f_t}) \end{aligned} \tag{3}$$

B. THE PROPOSED LSTCM CELL

The proposed LSTCM cell applies an activation function to the cell state instead of the hidden state. This is described as follows:

$$
\begin{aligned}
f_t &= \sigma(W_x^f x_t + W_h^f c_{t-1} + b^f) \\
i_t &= \sigma(W_x^i x_t + W_h^i c_{t-1} + b^i) \\
o_t &= \sigma(W_x^o x_t + W_h^o c_{t-1} + b^o) \\
j_t &= \lambda_j(W_x^j x_t + W_h^j h_{t-1} + b^j) \\
c_t &= \lambda_c(c_{t-1}) \odot f_t + i_t \odot j_t \\
h_t &= c_t \odot o_t
\end{aligned} \tag{4}
$$

As shown in Equation (4), the difference between LSTM and the proposed LSTCM is that LSTCM applies the activation function λ_c instead of λ_h. When we stack multiple LSTCM cells as layers to create a deep network, $h_{n,t}$ is calculated using $c_{n,t}$ without the activation function λ. Thus, the backpropagated gradients through f_t, i_t, and j_t exceed those of LSTM.

Equation (5) shows the BPTT of the proposed LSTCM.

$$
\begin{aligned}
\delta h_t =& \Delta + W_h^j \delta j_{t+1} + W_h^o \delta o_{t+1} \\
&+ W_h^i \delta i_{t+1} + W_h^f \delta f_{t+1} \\
\delta c_t =& \delta h_t \odot o_t + \lambda_c'(c_{t+1}) \odot f_{t+1} \odot \delta c_{t+1} \\
\delta j_t =& \delta c_t \odot i_t \odot \lambda_j'(\bar{j}_t) \\
\delta o_t =& \delta h_t \odot c_t \odot \sigma'(\bar{o}_t) \\
\delta i_t =& \delta c_t \odot j_t \odot \sigma'(\bar{i}_t) \\
\delta f_t =& \delta c_t \odot \lambda_c(c_{t-1}) \odot \sigma'(\bar{f}_t)
\end{aligned} \tag{5}
$$

In Equation (5), δx refers to $\partial L/\partial x$ where L is the loss function; Δ is the cumulative gradient from the layers above as calculated from all gradients of each state vector; λ' and σ' are derivatives of the activation functions; and $\{\bar{o}_t, \bar{i}_t, \bar{j}_t, \bar{f}_t\}$ are state vectors before the activation functions. The greatest difference compared to LSTM is δc_t. In the BPTT of LSTM, δc_t is calculated by multiplying $\lambda_h'(c_t)$ by δh_t, whereas in the BPTT of the proposed LSTCM, δc_t is calculated by multiplying $\lambda_c'(c_{t+1})$ by δc_{t+1}. Thus, in the proposed LSTCM cell, the gradient through c_{t+1} becomes smaller, but the gradient through h_t becomes larger. Consequently, the proposed LSTCM cell backpropagates a larger gradient through the layers than LSTM and shows better performance in deeper networks.

C. RECURRENT WEIGHT

In this subsection, we explain recurrent weights which are the most important part in RNNs to maintain the information from previous time steps to current one [19]. It means that the past state (h_t) and its gradient affect the current state (h_T) and its gradient in RNNs. Thus, for a stable learning of RNNs, the gradient must be in $[\epsilon, \gamma]$, i.e. $\epsilon \leq \frac{\partial J_T}{\partial h_t} \leq \gamma$ where J_T is an objective function to minimize in time step T. In this formulation, when the calculated gradient is less than ϵ, the gradient vanishing problem occurs, and when the calculated gradient is larger than γ, the gradient exploding problem occurs. Therefore, if we initialize the recurrent weights in a certain range to keep the gradient in $[\epsilon, \gamma]$, then RNNs are learned in stable manner without gradient vanishing or exploding problems. We conducted experiments to find out the proper initialization for recurrent weights in LSTCM cells and the results are explained in Section III-D.

D. USING AN ACTIVATION FUNCTION WITH A SINUSOIDAL FUNCTION

Several studies have use a periodic function as an activation function for deep neural networks [20], [21]. Particularly, Sitzmann *et al.* held that a periodic activation function was better than traditional activation functions in complicated signal problems such as natural images and 3D shapes [17]. The natural language problem is also a complicated signal problem, and LSTM and LSTCM as proposed in this paper also use activation functions, specifically λ_h and λ_c, respectively. Thus, we apply the sinusoidal function as an activation function for LSTM and LSTCM, λ_h and λ_c, instead of a hyperbolic tangent function. When we apply the sinusoidal activation function and train the network for machine translation tasks, the gradient exploding problem occurs. Therefore, we restrict the range of the sinusoidal activation function, with the final activation function then defined as follows:

$$
\lambda(x) = sin(x) \begin{cases} x = \pi, & \text{if } x > \pi \\ x = -\pi, & \text{if } x < -\pi \\ x, & \text{otherwise} \end{cases} \tag{6}
$$

III. EXPERIMENTS

In this section, we verify the proposed LSTCM cell outperforms LSTM on certain natural language tasks.

A. LANGUAGE MODELING TASK

We experimentally tested the proposed LSTCM cell on a language modeling task using Penn Treebank (PTB) dataset [22]. The purpose of the PTB dataset is to predict the next word based on a previous sequence of words. The training parameters were an initial weight of 0.1, an initial learning rate of 1.0 (decay by half at 1/2 epoch and 3/4 epoch), a batch size of 512, 1000 hidden neurons, and a dropout rate of 0.3. The experimental environment was based on https://github.com/KangSooHan/LSTCM. In addition, to prevent overfitting and to ensure stable learning, we applied dropout [23], gradient clipping [18], and warmup steps in the learning process.

To verify the effect of a deeper network on the language modeling task based on the PTB dataset, we compared one, three, and six layers of LSTM, ILSTM, LSTCM, and ILSTCM cells. We set 40, 80, and 120 epochs for the one-, three-, and six-layer models by considering the overfitting point when the training perplexity decreased but the validation perplexity increased. Each model underwent learned three times independently, and the final experimental results were calculated by averaging the perplexity of the three outcomes.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3040405, IEEE Access

TABLE 1. Results of word-level PTB for the proposed LSTCM model in comparison with basic LSTM, in terms of perplexity.

No. of Layers	Cell	Train	Val	Test
1	LSTM	56.918	96.196	**92.372**
	LSTCM	57.276	96.286	93.58
	ILSTM	60.318	100.508	95.166
	ILSTCM	62.691	101.831	96.481
3	LSTM	61.738	99.927	95.985
	LSTCM	62.967	100.102	**95.962**
	ILSTM	79.615	105.371	97.828
	ILSTCM	77.886	104.925	96.911
6	LSTM	73.268	111.175	109.571
	LSTCM	72.577	109.392	**107.956**
	ILSTM	82.486	114.102	110.716
	ILSTCM	83.772	110.098	108.285

Table 1 shows the results of the language modeling task based on the PTB dataset. As shown in the result, the simplest layer model shows the best performance. Because the language modeling task is relatively simple, the deeper network does not show an effect and the advantage of the proposed LSTCM, which transfers more gradients between the layers, is therefore not clear during the language modeling task.

B. MACHINE TRANSLATION TASK

Because the effect of the proposed LSTCM cell was not clear in the relatively simple task described above, we applied it to a more complex task, in this case a machine translation task. We used the IWSLT2015 English-Vietnamese dataset [24] and the WMT2014 English-German dataset for the machine translation task. The training parameters were as 10 epochs, an initial learning rate of 0.5 (decay by half at 1/2 epoch and 3/4 epoch), a batch size of 128, 512 hidden neurons, and a dropout rate of 0.3. The experimental environment found at https://github.com/tensorflow/nmt [25]. We used sequence-to-sequence models [6] and the Google Neural Machine Translation (GNMT) model [7] as the basis model, which consists of RNN cells and shows the best performance on the machine translation task.

Model. For the experiments, we used two backbone networks based on a sequence-to-sequence model. For the IWSLT 2015 English-Vietnamese dataset, we used a sequence-to-sequence model based on a basic encoder-decoder structure along with the Luong attention mechanism [6]. The encoder layer consists initially of bidirectional cells and then stacked unidirectional cells. The encoder layer calculates the attention for the result of the encoder cells using the Luong attention mechanism and then passes it to the decoder layer. The decoder layer consists of stacked unidirectional cells, and it predicts the next word based on the attention value from the encoder layer and input words. For the WMT 2014 English-German dataset, we used the GNMT model. The GNMT model is a deep LSTM network with encoder and decoder layers that also uses residual connections along with attention connections from the decoder to the encoder. The GNMT model calculates the attention value of each unidirectional cell in the encoder layer, and the decoder layer predicts the next word based on each previous attention value from encoder layer. To observe the effect of a deeper network on the machine translation task, we compared model with one, four, and seven layers using LSTM, ILSTM, LSTCM, and ILSTCM cells. The model with i layers refers to the setting of i layers for the encoder, with last layer as the bidirectional cell, while $i-1$ layers are set for the decoder without a bidirectional cell.

Datasets. The IWSLT data used here is from translated TED talks and contains 133K training sentence pairs. The dataset is provided by the IWSLT 2015 Evaluation Campaign [24]. We applied a data preprocessing method [26] and thus obtained 17.2K vocabulary items for English and 7.7K vocabulary for Vietnamese. We validated and tested the model using TED tst2012 and tst2013, respectively. The WMT dataset contains approximately 4M sentence pairs. Sentences were encoded by means of byte-pair encoding [27] involving the use of a shared resource target vocabulary of approximately 37K tokens.

Training Parameters. The training parameters were an initial weight of 0.1, an initial learning rate of 0.2, a batch size of 100, 512 hidden neurons, and a dropout rate of 0.3. Additionally, to prevent overfitting and to ensure stable learning, we applied dropout [23], gradient clipping [18], and warmup steps during the learning process. For the IWSLT 2015 English-Vietnamese dataset, which is a relatively small dataset, we utilized 60,000 training steps, and the learning rate decayed by half at 1/2 and 3/4 training steps. For the WMT 2014 English-German dataset, we utilized 350,000 training steps, and the learning rate decayed by half at every 17,500 steps after half of the training steps.

C. COMPARISON RESULT WITH LSTM

As shown in Section III-A, the proposed LSTCM cell did not show an advantage compared to the LSTM cell on the language modeling task because the language modeling task is relatively simple and does not require a deeper network. However, in a more complex task, in this case the machine translation task in Section III-B, the proposed LSTCM cell outperformed the LSTM cell. We used IWSLT2015 and WMT2014 datasets to compare the proposed LSTCM and LSTM cells on the machine translation task.

IWSLT 2015 English-Vietnamese. Table 2 shows the result of the sequence-to-sequence model with Luong attention using the proposed LSTCM and ILSTCM cells as well as LSTM, ILSTM, and GRU cells trained based on the IWSLT 2015 dataset. We utilized two, four, and seven layers to compare the performances according to the layer depth. Each

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3040405, IEEE Access

TABLE 2. Performance outcome of the proposed LSTCM on IWSLT 2015 English-Vietnamese dataset based on sequence-to-sequence model with Luong attention in terms of the BLEU score.

No. of Layers	Cell	Train ppl	tst2012	tst2013
2	LSTM	**4.098**	21.43	23.16
	LSTCM	4.196	20.89	22.82
	GRU	4.978	14.62	15.43
	ILSTM	4.927	**23.67**	**26.68**
	ILSTCM	5.016	23.59	26.49
4	LSTM	**3.768**	22.42	25.23
	LSTCM	3.859	22.67	24.97
	GRU	4.653	12.58	12.72
	ILSTM	4.787	**24.50**	26.71
	ILSTCM	4.967	24.37	**26.84**
7 without skip connection	LSTM	8.291	12.49	12.80
	LSTCM	7.726	16.92	16.28
	GRU	9.911	8.12	10.09
	ILSTM	5.877	22.64	24.64
	ILSTCM	**5.392**	**23.29**	**26.13**
7 with skip connection	LSTM	2.928	16.82	17.24
	LSTCM	**2.901**	20.71	21.57
	GRU	3.362	15.61	16.23
	ILSTM	3.105	23.58	25.93
	ILSTCM	3.062	**23.71**	**26.11**

model underwent three independent learning trials, and the final experimental results were calculated by averaging the perplexity and BLEU scores of these three trials.

As shown in Table 2, the four-layer model showed the overall best performance. The tst2012 BLEU scores of four layers using ILSTM and ILSTCM cells were 24.50 and 24.37, respectively, and the corresponding tst2013 BLEU scores were 26.71 and 26.84. Accordingly, there were no major differences between the ILSTM and ILSTCM cells. On the other hand, the tst2012 and tst2013 BLEU scores for seven layers without or with a skip connection using ILSTM and ILSTCM cells showed that the proposed ILSTCM cell outperformed the ILSTM cell at a meaningful level. Also, ILSTM and ILSTCM cells, which applied the aforementioned independent approach, outperformed vanilla LSTM and LSTCM cells, respectively, because the independent cells mitigated the overfitting problem.

WMT 2014 English-German. Table 3 shows the result of the GNMT model using the proposed LSTCM and ILSTCM cells, as well as LSTM and ILSTM cells trained based on the WMT 2014 dataset. We utilized two, four, and seven layers to compare the performance capabilities according to the layer depth. Each model underwent three independent learning trials, and the final experimental results were calculated by averaging the perplexity and BLEU scores of these three trials.

As shown in Table 3, the seven-layer model with a skip

TABLE 3. Performance outcome of the proposed LSTCM on WMT2014 English-German dataset based on GNMT model in terms of the BLEU score.

No. of Layers	Cell	Train ppl	tst2013	tst2014
2	LSTM	**6.542**	23.12	24.06
	ILSTM	7.656	**23.73**	**24.91**
	LSTCM	6.654	23.01	23.89
	ILSTCM	7.421	23.55	24.76
4	LSTM	6.271	23.41	24.68
	ILSTM	6.690	**24.04**	25.02
	LSTCM	**6.268**	23.39	24.53
	ILSTCM	6.702	23.94	**25.06**
7 without skip connection	LSTM	17.66	10.32	10.16
	ILSTM	20.98	9.43	8.20
	LSTCM	8.17	15.37	14.61
	ILSTCM	**10.39**	**19.31**	**19.75**
7 with skip connection	LSTM	6.193	22.55	23.98
	ILSTM	6.567	23.98	25.15
	LSTCM	**6.186**	22.83	24.11
	ILSTCM	6.632	**24.09**	**25.26**

connection using the ILSTCM cell showed the best performance overall in terms of both the tst2013 and tst2014 BLEU scores (24.09 and 25.26, respectively). Unlike the IWSLT 2015 dataset, the seven-layer model showed the best performance on the WMT 2014 dataset, confirming that a deeper network is better on complex and large datasets. As in the experiment with the IWSLT 2015 dataset, ILSTM and ILSTCM cells, which applied an independent approach, outperformed the vanilla LSTM and LSTCM cells, respectively, and the proposed LSTCM cell showed better performance than the LSTCM cell on a deeper network.

Additionally, Figures 2 and 3 depict the average gradient when we train the seven-layer GNMT model using ILSTCM and ILSTM cells without and with a skip connection for the WMT 2014 dataset, respectively. As shown in Figures 2 and 3, the average gradient for the ILSTCM cell is greater that for the ILSTM cell. This result verifies that the better performance by the proposed LSTCM cell stems from the greater level of gradient transference compared to that in the LSTM cell after applying the activation function to the cell state instead of the hidden state.

D. WEIGHT INITIALIZATION IN LSTCM

For the stable learning of RNNs, the recurrent weights need to be initialized properly to keep gradients in $[\epsilon, \gamma]$, i.e. $\epsilon \leq \frac{\partial J_T}{\partial h_t}$. Thus, we found the proper weight initialization range experimentally. Figure 4 shows the result of the sequence-to-sequence model with Luong attention using the proposed ILSTCM cell in change of weight initialization. As shown in Figure 4, when the weights were initialized greater than 0.22, the model was not learned because the gradient exploding occurred. In this case, the BLEU score was *NAN*, thus it is represented as 0 in the graph. Moreover, when the weights

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3040405, IEEE Access

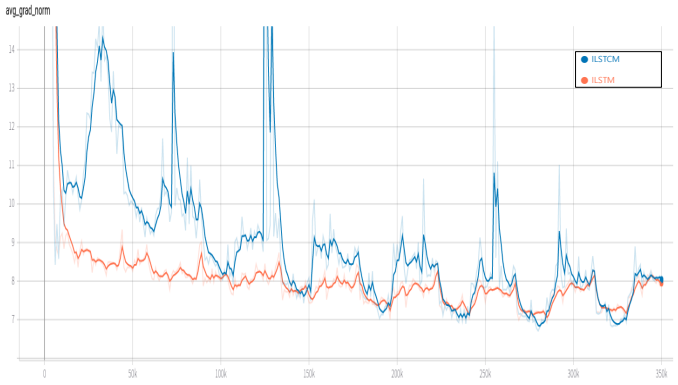


FIGURE 2. Average gradient of the seven-layer GNMT model using ILSTCM and ILSTM cells without a skip connection for the WMT 2014 dataset. The blue and orange lines represent the average gradient of the ILSTCM and ILSTM cells, respectively.

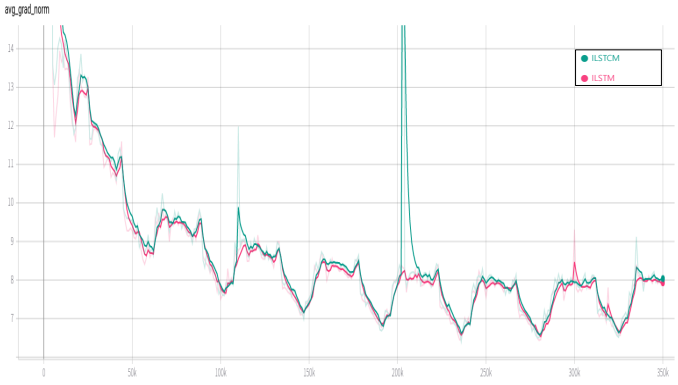


FIGURE 3. Average gradient of the seven-layer GNMT model using ILSTCM and ILSTM cells with a skip connection for WMT 2014 dataset. The green and magenta lines represent the average gradient of the ILSTCM and ILSTM cells, respectively.

were initialized less than 0.01, the BLEU score was very low compared to other weight initialization cases because the gradient vanishing occurred. Thus, we can conclude the proper weight initialization range for stable learning of LSTCM cells is $[0.01, 0.22]$.

E. SINUSOIDAL ACTIVATION FUNCTION PERFORMANCE

Table 4 shows the result of the sequence-to-sequence model with Luong attention using the proposed ILSTCM cell and the ILSTM cell with the sinusoidal activation function with training based on the IWSLT 2015 dataset. We compare the results between those with the sinusoidal activation function and those with the hyperbolic tangent activation function. As shown in Table 4, every layer combination with the sinusoidal activation function outperformed the corresponding cases with the hyperbolic tangent activation function. The proposed ILSTCM cell especially showed an improvement than ILSTM cell when we applied the sinusoidal activation function, and the overall best performance was achieved by the four-layer model using the ILSTCM cell with the sinusoidal activation function (tst2013 BLEU score: 24.71

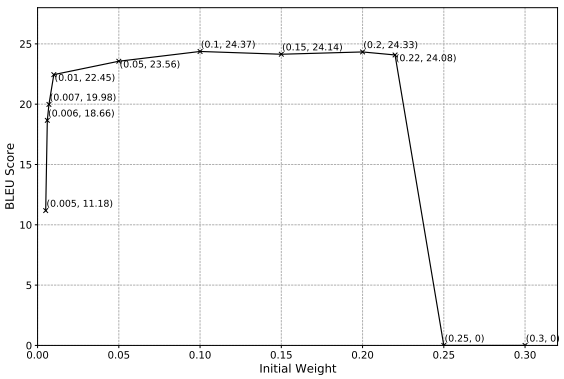


FIGURE 4. Performance of the proposed ILSTCM with different weight initializations on IWSLT 2015 English-Vietnamese dataset based on four-layer of sequence-to-sequence model with Luong attention in terms of BLEU score on tst2013.

TABLE 4. Performance outcome of the proposed LSTCM with the sinusoidal activation function on the IWSLT 2015 English-Vietnamese dataset based on the sequence-to-sequence model with Luong attention in terms of the BLEU score.

No. of Layers	Cell	Activation	Train ppl	tst2013	tst2014
4	ILSTM	tanh	**4.787**	24.50	26.71
	ILSTM	sin	5.091	24.68	27.43
	ILSTCM	tanh	4.967	24.37	26.84
	ILSTCM	sin	5.047	**24.71**	**27.96**
7 without skip connection	ILSTM	tanh	5.877	22.64	24.64
	ILSTM	sin	7.447	21.68	23.88
	ILSTCM	tanh	**5.392**	23.29	**26.13**
	ILSTCM	sin	6.336	**23.55**	26.12
7 with skip connection	ILSTM	tanh	3.105	23.58	25.93
	ILSTM	sin	3.532	24.17	26.92
	ILSTCM	tanh	**3.062**	23.71	26.11
	ILSTCM	sin	3.394	**24.36**	**27.10**

and tst2014 BLEU score: 27.96). Thus, we can conclude that the sinusoidal activation function is better than the hyperbolic tangent activation function on complicated networks.

F. DISCUSSION

In this subsection, we discuss the experimental results and advantages. The results show that a deeper network for relatively simple task, such as a language modeling task based on PTB datasets, showed low performance compared to those on a shallow network. On the other hand, for a more complex task, such as a machine translation task, a deeper network using four or seven layers showed better performance than those on a shallow network. More specifically, because the IWSLT2015 dataset contains fewer words and sentences than the WMT2014 dataset, the four-layer model outperformed the seven-layer model on the machine translation task based on the IWSLT2015 dataset, whereas the seven-layer model with a skip connection outperformed on the machine trans-

TABLE 5. Training time for the proposed LSTCM and and LSTM in terms of words per second.

Task	LSTM / LSTCM (wps)	ILSTM / ILSTCM (wps)
PTB	96000 / 95000	83000 / 81000
IWSLT(4layer)	176300 / 178400	158300 / 159600
WMT(4layer)	164200 / 166200	147200 / 148900

lation task based on the WMT2014 dataset. Additionally, ILSTM and ILSTCM, which apply independent concepts to LSTM and LSTCM, showed worse perplexity performance during the training phase. However, better perplexity during the training phase did not guarantee a model with better learning. On a language modeling task based on the PTB dataset, the difference between the perplexity level between the training and the test datasets was less when the ILSTM and ILSTCM were applied as compared to when LSTM and LSTCM were applied, meaning that the independent concept prevents the network overfitting problem. Moreover, on the machine translation task, the model using LSTM and LSTCM showed better perplexity in the training phase, whereas the model using ILSTM and ILSTCM showed a better BLEU score in the test phase. Thus, we can conclude that applying the independent concept to LSTM and LSTCM cells causes the network to train to the proper direction and prevents the overfitting problem.

The basic structure of proposed LSTCM cell is similar to LSTM cell. Thus, the well-studied approach for LSTM, especially distributed learning approach from multiple clusters or multi GPUs and performance improvement approach including dropout and layer normalization, also can be applied for LSTCM in the same manner. Moreover, as shown in Table 5, the training time for the proposed LSTCM and LSTM were not much different, therefore the existing applications of RNNs can use LSTCM cells instead of LSTM cells with ease.

IV. CONCLUSION

This paper proposed what is termed a long short-time complex memory (LSTCM) cell to solve the gradient vanishing problem in recurrent neural networks (RNNs) and long short-term memory (LSTM), especially when the network is deep. The proposed LSTCM cell applied an activation function to the cell state instead of the hidden state to transfer more of the gradient to the next layer. Moreover, we applied an sinusoidal function as an activation function of LSTCM cell instead of a hyperbolic tangent function. We conducted experiments on language modeling and machine translation tasks based on the PTB, IWLST2015, and WMT2014 datasets. The experimental results showed that the proposed LSTCM cell outperformed the LSTM cell on deeper networks for complex tasks. Furthermore, ILSTCM with the independent concept applied to LSTCM showed more stable training by preventing the overfitting problem.

REFERENCES

[1] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko, "Sequence to sequence-video to text," in Proceedings of the IEEE international conference on computer vision, pp. 4534–4542, 2015.

[2] A. Liu, Y. Qiu, Y. Wong, Y. Su, and M. Kankanhalli, "A fine-grained spatial-temporal attention model for video captioning," IEEE Access, vol. 6, pp. 68463–68471, 2018.

[3] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, et al., "Deep speech 2: End-to-end speech recognition in english and mandarin," in International conference on machine learning, pp. 173–182, 2016.

[4] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, et al., "Tacotron: Towards end-to-end speech synthesis," arXiv preprint arXiv:1703.10135, 2017.

[5] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," arXiv preprint arXiv:1409.0473, 2014.

[6] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," arXiv preprint arXiv:1508.04025, 2015.

[7] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," arXiv preprint arXiv:1609.08144, 2016.

[8] J. Xie, B. Chen, X. Gu, F. Liang, and X. Xu, "Self-attention-based bilstm model for short text fine-grained sentiment classification," IEEE Access, vol. 7, pp. 180558–180570, 2019.

[9] N. M. Sommer, S. Velipasalar, L. Hirshfield, Y. Lu, and B. Kakillioglu, "Simultaneous and spatiotemporal detection of different levels of activity in multidimensional data," IEEE Access, vol. 8, pp. 118205–118218, 2020.

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

[11] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," arXiv preprint arXiv:1412.3555, 2014.

[12] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," arXiv preprint arXiv:1611.01578, 2016.

[13] C. Gulcehre, M. Moczulski, M. Denil, and Y. Bengio, "Noisy activation functions," in International conference on machine learning, pp. 3059–3068, 2016.

[14] Q. V. Le, N. Jaitly, and G. E. Hinton, "A simple way to initialize recurrent networks of rectified linear units," arXiv preprint arXiv:1504.00941, 2015.

[15] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao, "Independently recurrent neural network (indrnn): Building a longer and deeper rnn," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5457–5466, 2018.

[16] P. Gonnet and T. Deselaers, "Indylstms: Independently recurrent lstms," arXiv preprint arXiv:1903.08023, 2019.

[17] V. Sitzmann, J. N. Martel, A. W. Bergman, D. B. Lindell, and G. Wetzstein, "Implicit neural representations with periodic activation functions," arXiv preprint arXiv:2006.09661, 2020.

[18] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in International conference on machine learning, pp. 1310–1318, 2013.

[19] S. Li, W. Li, C. Cook, Y. Gao, and C. Zhu, "Deep independently recurrent neural network (indrnn)," arXiv preprint arXiv:1910.06251, 2019.

[20] G. Parascandolo, H. Huttunen, and T. Virtanen, "Taming the waves: sine as activation function in deep neural networks," 2016.

[21] L. Munkhdalai, T. Munkhdalai, K. H. Park, H. G. Lee, M. Li, and K. H. Ryu, "Mixture of activation functions with extended min-max normalization for forex market prediction," IEEE Access, vol. 7, pp. 183680–183691, 2019.

[22] M. P. Marcus, B. Santorini, M. A. Marcinkiewicz, and A. Taylor, "Treebank-3," Linguistic Data Consortium, Philadelphia, vol. 14, 1999.

[23] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in Advances in neural information processing systems, pp. 1019–1027, 2016.

[24] M. Cettolo, N. Jan, S. Sebastian, L. Bentivogli, R. Cattoni, and M. Federico, "The iwslt 2016 evaluation campaign," in International Workshop on Spoken Language Translation, 2016.

[25] M. Luong, E. Brevdo, and R. Zhao, "Neural machine translation (seq2seq) tutorial," https://github.com/tensorflow/nmt, 2017.

[26] M.-T. Luong and C. D. Manning, "Stanford neural machine translation systems for spoken language domain," in International Workshop on Spoken Language Translation, (Da Nang, Vietnam), 2015.
[27] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," arXiv preprint arXiv:1508.07909, 2015.

SOO-HAN KANG received his B.S. degree in computer science and engineering in 2019 from Seoul National University of Science and Technology, Seoul, Korea, where he is currently pursuing a M.S. degree.

His research interests include machine learning and human-robot interaction.

JI-HYEONG HAN received her B.S. and Ph.D. degrees in electrical engineering from KAIST, Daejeon, Korea, in 2008 and 2015, respectively.

From 2015 to 2017, she was a Senior Researcher with the Electronics and Telecommunications Research Institute in Daejeon, Korea. Since 2017, she has been with Seoul National University of Science and Technology, Seoul, Korea, where she is currently an Assistant Professor. Her research interests include machine learning, human-centered intelligent robotics, and human-robot interaction.