Proceedings of ICSP2020

Jujubecake: An Extension of LSTM Considering Correlation among Input Blocks

Zeyu Zhao*, Wei-Qiang Zhang*, Jia Liu*, Feng Zhang†, Xiangjun Li†, Le Yu†
*Beijing National Research Center for Information Science and Technology
Department of Electronic Engineering, Tsinghua University, Beijing 100084
† China Mobile Information Security Center, Beijing 100053, China
Email: zzy17@mails.tsinghua.edu.cn, wqzhang@tsinghua.edu.cn

2020 15th IEEE International Conference on Signal Processing (ICSP) | 978-1-7281-4480-1/20/$31.00 ©2020 IEEE | DOI: 10.1109/ICSP48669.2020.9320915

***Abstract*—This paper proposes a new neural network unit for sequential input, called Jujubecake Unit or Jujubecake Cell. In the chain structure formed by this unit, not only the correlation among individual inputs but also the correlation among blocks consist of several inputs is considered so that that performance can be improved. In this paper, Jujubecake model and the multi-layer LSTM model are evaluated in two experiments: language modelling and audio event detection. The experimental results show that Jujubecake Unit proposed in this paper outperforms the traditional LSTM.**

***Keywords*—Long Short Term Memory, sequence modelling, language model, audio event detection**

I. Introduction

Life is full of sequence signals, such as the speech, the audio in the environment, and the sequence of words. In order to deal with sequence signals in deep learning, different methods have been proposed from different perspectives. In recent years, with the popularity of deep neural networks (DNN), the use of DNN, especially recurrent neural networks (RNN) to process sequence signals has yielded a lot of good results [1]–[3]. A simplest RNN has a structure as shown in Fig. 1. However, the problem of vanishing gradients [4], [5] or exploding gradients may occur, and as the number of iterations increases, the new unit would get less information from past ones. In order to solve this problem, Long Shot-Term Memory (LSTM) [6] has been proposed and been applied in various fields.

The idea of forming an RNN is intuitive. Each element of a sequence sequentially inputs into the unit of RNN, which mainly considers the correlation among input elements. However, this approach does not reflect a more advanced correlation. Namely, if the input sequence has some local features, we can treat adjacent elements of input sequence as one single block, so that information flows not only among the input elements but also among these small blocks consist of adjacent input elements.

For example, in the task of audio event detection [7], [8], we can apply an LSTM-RNN to predict a label for each input frame indicating whether a specific audio event happened in this frame of audio. However, this can be sometimes much unstable as we will see below in our experiments. The reason is that the information contained in a single frame is not sufficient to predict a label because of its short length. Thus, if we consider the adjacent frames at the same time, we may expect a more robust prediction. We believe that the information contained in these adjacent frames is much more efficient to give us a reliable output than only one single frame.

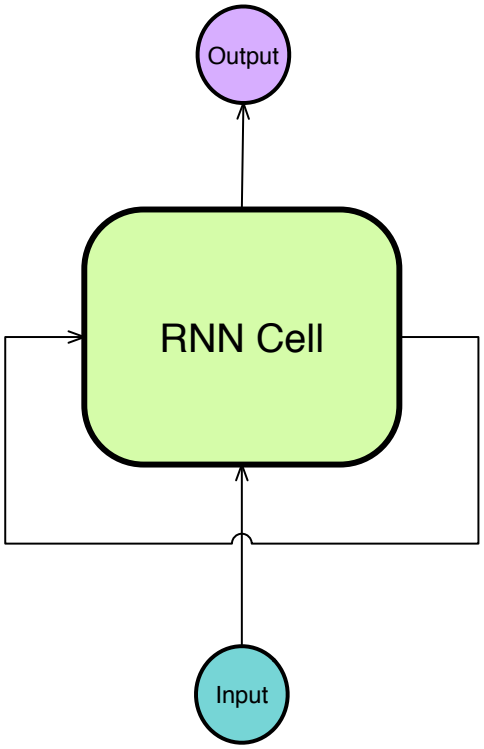

Fig. 1. A simplest RNN unit

Another instance is language modelling. In the problem of language modelling, a typical application is to predict the next word given an input sequence of words. A conventional method is also using LSTM-RNN or multi-layer LSTM-RNN, which takes each word from the input sequence as inputs and gives out a prediction of the next word. Similarly, in this process, the correlation between each word in the input

The corresponding author is Wei-Qiang Zhang.

This work was supported by the National Natural Science Foundation of China under Grant No. U1836219 and in part supported by Joint Foundation of the Ministry of Education of China and China Mobile under Grant No. MCM20180505.

sequence is considered explicitly. If we can, however, treat several adjacent words as a phrase and concern the correlation among phrases explicitly as well, we may expect a better performance.

In this paper, we call several adjacent elements of the input sequence a block. The primary motivation of this work, Jujubecake Cell, is to consider the correlation among blocks from the input sequence explicitly. To implement the above, we add some new components based on LSTM so that local features contained in blocks can be transferred and shared when processing the input sequence.

II. Related Work

Since its introduction in 1997, LSTM has been widely used in different fields, including the establishment of language models [2], the establishment of acoustic models [1], and speech recognition [3]. The idea of improving LSTM from a basic one is not novel. There are indeed improvements to basic LSTM that have proposed the concept of Nested LSTM [9]. Although both of them have got some good results on different tasks, they still have some shortcomings. As for [9], their direct purpose is to increase the complexity of the structure, and the authors expect to model the problem better. However, we think this approach lacks a clear physical explanation.

We propose a new RNN cell based on LSTM. In the proposed RNN cell, there is an intuitive physical explanation and an apparent nested structure.

III. Jujubecake Structure

In this section, we will first review the basic LSTM cell structure and then introduce our novel Jujubecake Cell and give some analysis in depth.

A. Basic LSTM cell structure

A basic LSTM [10] structure can be demonstrated as Fig. 2. In this figure, the blue part indicates the input to the cell, purple part represents the output, the yellow part represents the neural network layer, the pink part represents the element-wise operation, and the orange part represents the matrix operation (concatenation). More specifically, it can be expressed in accordance with eqs. (1) to (6) [11], where $\sigma(\cdot)$ represents the sigmoid function, and $\circ$ represents element-wise multiply.

$$\boldsymbol{f}_t = \sigma\left(\boldsymbol{x}_t\boldsymbol{W}_{xf} + \boldsymbol{h}_{t-1}\boldsymbol{W}_{hf} + \boldsymbol{b}_f\right) \tag{1}$$

$$\boldsymbol{o}_t = \sigma\left(\boldsymbol{x}_t\boldsymbol{W}_{xo} + \boldsymbol{h}_{t-1}\boldsymbol{W}_{ho} + \boldsymbol{b}_o\right) \tag{2}$$

$$\boldsymbol{i}_t = \sigma\left(\boldsymbol{x}_t\boldsymbol{W}_{xi} + \boldsymbol{h}_{t-1}\boldsymbol{W}_{hi} + \boldsymbol{b}_i\right) \tag{3}$$

$$\widetilde{\boldsymbol{c}}_t = \tanh\left(\boldsymbol{x}_t\boldsymbol{W}_{xc} + \boldsymbol{h}_{t-1}\boldsymbol{W}_{hc} + \boldsymbol{b}_c\right) \tag{4}$$

$$\boldsymbol{c}_t = \boldsymbol{c}_{t-1} \circ \boldsymbol{f}_t + \widetilde{\boldsymbol{c}}_t \circ \boldsymbol{i}_t \tag{5}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \circ \tanh\left(\boldsymbol{c}_t\right) \tag{6}$$

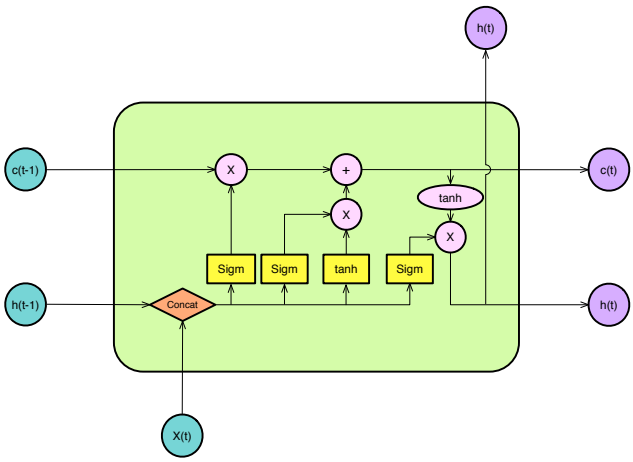

Fig. 2. A basic LSTMCell structure

B. Jujubecake Cell structure

Our novel structure is an extension of LSTM, as shown in Fig. 3, which consists of an external LSTM and several internal LSTMs, similar to the structure of a Chinese snack, Jujubecake, so we call it Jujubecake Cell. Note that the number of the internal LSTMs is configurable, and we can still use eqs. (1) to (6) to describe it with only several small modifications.

Now we will introduce the workflow of this Jujubecake Cell. For convenience, let us take a Jujubecake Cell with three internal LSTMs, as shown in Fig. 3 as an example. First, the inputs $\boldsymbol{x}_{T1}$, $\boldsymbol{x}_{T2}$, $\boldsymbol{x}_{T3}$ (denoted by Input1, Input2, Input3 in Fig. 3) are input to three internal LSTMs respectively and the outputs of them are concatenated together as the input of the external LSTM $\boldsymbol{X}_T$. Then, the output and hidden state of the previous Jujubecake Cell H_{T-1}, C_{T-1} ($H(T-1)$, $C(T-1)$ in Fig. 3) are input to the external LSTM to calculate the forgetting threshold $\boldsymbol{F}_T$, input threshold $\boldsymbol{I}_T$ and output threshold $\boldsymbol{O}_T$ according to eqs. (7) to (9). After that, we calculate the hidden states of internal LSTMs $\boldsymbol{c}_{T1}$, $\boldsymbol{c}_{T2}$, $\boldsymbol{c}_{T3}$ respectively using the output ($h(T-1)$ in Fig. 3) and the hidden state ($c(T-1)$ in Fig. 3) from the last internal LSTM of the last Jujubecake Cell and concatenate them to form the current input state of the external LSTM $\widetilde{\boldsymbol{C}}_T$ according to eq. (10). Then, we combine the current input state $\widetilde{\boldsymbol{C}}_T$ with the previous state $\boldsymbol{C}_{T-1}$ considering the input threshold $\boldsymbol{I}_T$ and the forgetting threshold $\boldsymbol{F}_T$ according to eq. (11) so that the current state $\boldsymbol{C}_T$ ($C(T)$ in Fig. 3) can be calculated. Finally, according to eq. (12), the output of current Jujubecake Cell $\boldsymbol{H}_T$ ($H(T)$ in Fig. 3) can be calculated given output threshold $\boldsymbol{O}_T$ and current state $\boldsymbol{C}_T$.

$$\boldsymbol{F}_T = \sigma\left(\boldsymbol{X}_T\boldsymbol{W}_{XF} + \boldsymbol{H}_{T-1}\boldsymbol{W}_{HF} + \boldsymbol{B}_F\right) \tag{7}$$

$$\boldsymbol{O}_T = \sigma\left(\boldsymbol{X}_T\boldsymbol{W}_{XO} + \boldsymbol{H}_{T-1}\boldsymbol{W}_{HO} + \boldsymbol{B}_O\right) \tag{8}$$

$$\boldsymbol{I}_T = \sigma\left(\boldsymbol{X}_T\boldsymbol{W}_{XI} + \boldsymbol{h}_{T-1}\boldsymbol{W}_{HI} + \boldsymbol{B}_I\right) \tag{9}$$

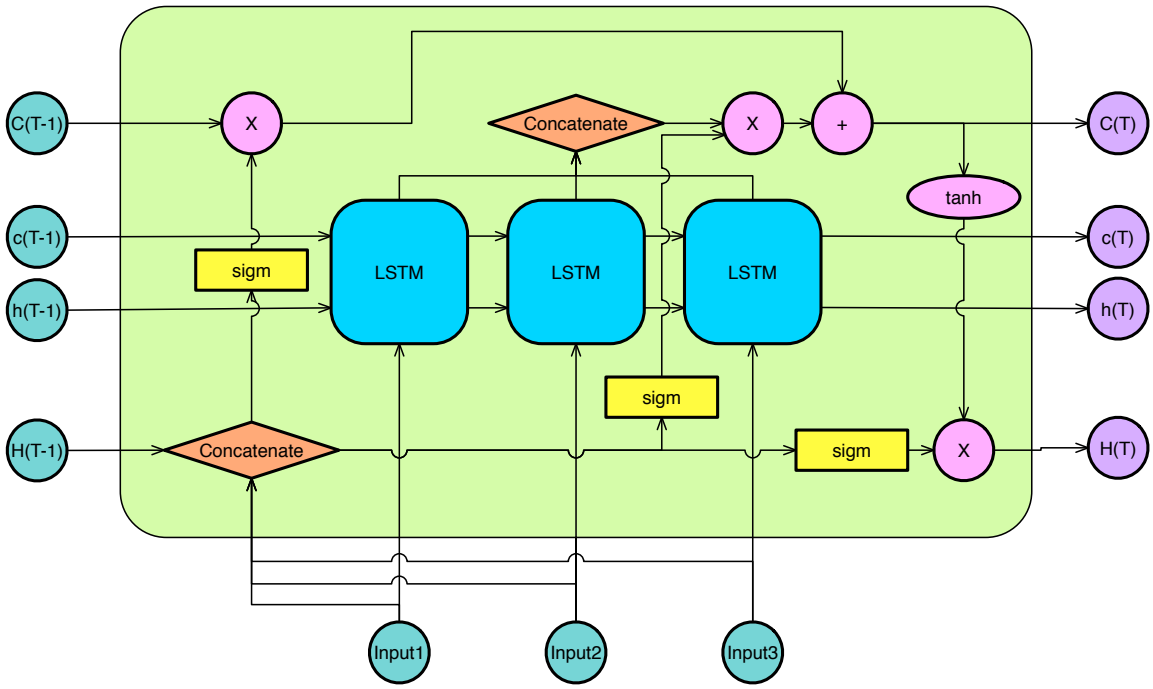

Fig. 3. Jujubecake Cell structure (3 internal LSTMs in this example)

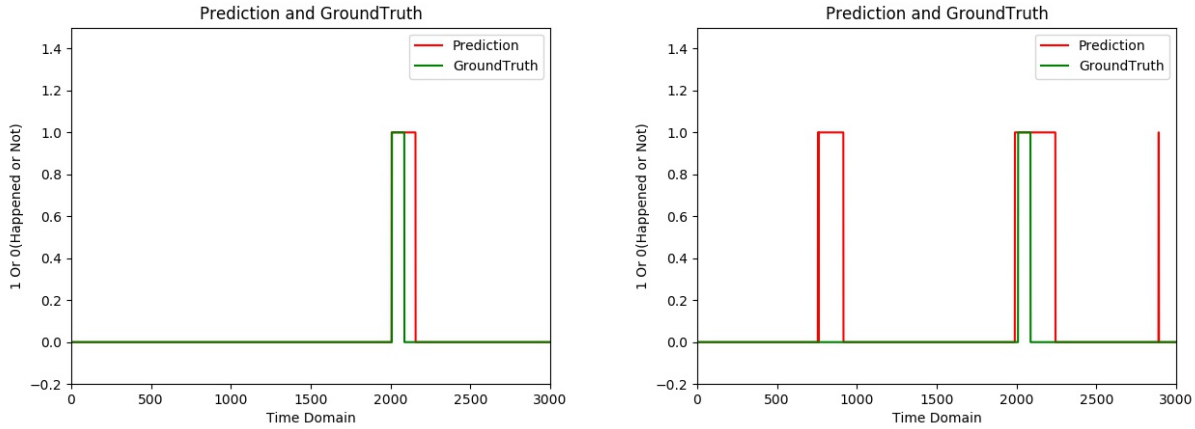

Fig. 4. Outputs of Jujubecake (left) and LSTM (right) for the same audio file

$$\widetilde{\boldsymbol{C}}_T = [\boldsymbol{c}_{T1}, \boldsymbol{c}_{T2}, \boldsymbol{c}_{T3}] \tag{10}$$

$$\boldsymbol{C}_T = \boldsymbol{C}_{T-1} \circ \boldsymbol{F}_T + \widetilde{\boldsymbol{C}}_T \circ \boldsymbol{I}_T \tag{11}$$

$$\boldsymbol{H}_T = \boldsymbol{O}_T \circ \tanh(\boldsymbol{C}_T) \tag{12}$$

C. Analysis of Jujubecake Cell

In our structure, we keep the original basic formula unchanged, and the only thing that needs to be changed is eq. (4). Specifically, we formulate Jujubecake Cell with eqs. (7) to (12). In eq. (10), $\boldsymbol{c}_{T1}$, $\boldsymbol{c}_{T2}$ and $\boldsymbol{c}_{T3}$ denote the states of each internal LSTM respectively, which implies that the external LSTM has access to all the information of the internal LSTMs. Another noticeable change in our Jujubecake Cell is that the output is directly determined by the state of the external LSTM, but not states of internal LSTMs. We should also note that involving this new structure does not break the chain of the internal LSTMs. In addition to the internal LSTM chain structure, we also have a new form of information flow. That is the external LSTM chain, which can transfer local features contained in blocks.

The number of internal LSTMs in one external LSTM is configurable and can be set manually indicating the range of

local features we need to concern. We can imagine that if we set the number of internal LSTMs just equal to the number of elements of the input sequence, the output of Jujubecake Cell will only contain the overall features of the input.

IV. Experiment

In this section, we will use experiments to evaluate the performance of the Jujubecake Cell, compared with multi-layer LSTM and other models in other papers. To illustrate the performance of Jujubecake Cell, we set the number of each model parameters equally.

One of the experiments is about word-level language modelling. In this paper, we use Penn Treebank [12] dataset. The other experiment is the detection of the glass-break event in DCASE2017 Task2. The dataset can be downloaded from the official website of DCASE2017. We have placed all the code needed for the experiments on Github[1].

All experiments in this paper are based on TensorFlow deep learning framework [13].

A. Language modeling

Penn Treebank has about 929k training words, 73k validation words, and 82k test words. The vocabulary is around 10k. We use Jujubecake Cell and 2-layer LSTM model to build the language model separately. The structure of the two models and the number of model parameters are shown in the Table I and Table II respectively.

TABLE I
MODEL STRUCTURE OF JUJUBECAKE MODEL

Layer (type)	Output Shape	Param
JujubeCake Cell	768	16259584
Dense	9999	15368463
Total params		36,747,535
Trainable params		36,747,535
Non-trainable params		0

TABLE II
MODEL STRUCTURE OF LSTM MODEL

Layer (type)	Output Shape	Param
BasicLSTM Cell	1024	8392704
BasicLSTM Cell	1024	8392704
Dense	9999	10248975
Total params		37,273,359
Trainable params		37,273,359
Non-trainable params		0

The training batch size is 32. Both models are trained no less than 50 epochs before convergence. The final model is evaluated on the validation set and the test set, and the results are shown in Table III. We also present some results from other models [10]. The metric here is perplexity, which is defined as eq. (13), where $p_{\mathrm{target_i}}$ denotes the probability output by the model for the target word in the ith sample and N denotes the total number of samples.

$$\mathrm{perplexity} = \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\ln p_{\mathrm{target_i}}\right) \tag{13}$$

TABLE III
MODEL PERFORMANCE ON TRAINING SET, VALIDATION SET AND TEST SET

Model	Training	Validation	Test
Jujubecake with dropout	111.4	198.8	177.2
2-layer LSTM model	242.2	338.0	298.7
non-regularized LSTM		120.7	114.5
Medium regularized LSTM		86.2	82.7
Large regularized LSTM		82.2	78.4

We note that Jujubecake model has a better performance than the 2-layer LSTM model. As for a reason, as mentioned in section III-C, Jujubecake Cell not only considers the relationship among words but among phrases as well.

Indeed, our Jujubecake Cell does not achieve state-of-the-art results in [10]. However, this is just a novel RNN cell, and it is at the very beginning that we have to do many optimisations about it in the future.

B. Audio event detection

In this experiment, there are about 3000 samples in the training set. Each sample is a piece of audio with a duration of 30 seconds. Within this 30 seconds, the glass-break event will occur randomly for a short period or not. We extract 20-dimensional MFCCs (Mel-frequency cepstral coefficients) features for each frame of audio. In the original annotation, the start and end time of the glass-break event is marked, and this annotation can be used to mark each frame whether the glass-break event occurs or not in this frame (if the event occurred, we denote it as 1, otherwise 0). Besides, the focal loss was used in our experiments [14].

We evaluate Jujubecake model on this task and compare it with some of the results in DCASE2017 Task2. The official metrics are applied, and their specific definitions are derived from [**?**]. The lower the error rate (ER) or, the higher the F score means, the better the overall performance achieve.

The structure and the number of the model parameter for Jujubecake model are shown in Table IV.

We set a batch size of 10 to train our model. After the training of 20 epochs, we got the converged model. We evaluate the model on the official evaluation set, which contains about 500 samples. The evaluation results are shown in Table V. Besides, we also list some other results to compare with our Jujubecake model.

To illustrate the problem in a more intuitive method, Fig. 4 shows the different results output by two models for the same piece of input audio. We can see that Jujubecake model (left in Fig. 4) is much stabler. This is because it considers the correlation among not only frames but also blocks that consist of multiple adjacent frames. Compared with 2-layer LSTM

[1]https://github.com/ZhaoZeyu1995/JujubeCakeCell

TABLE IV
MODEL STRUCTURE AND THE NUMBER OF MODEL PARAMETERS

Layer	Output Shape	Param
JujubeCake Cell	768	2193664
Reshape	$16 \times 16 \times 3$	0
Conv2D	$16 \times 16 \times 32$	2432
MaxPooling2D	$8 \times 8 \times 32$	0
Conv2D	$8 \times 8 \times 64$	51264
MaxPooling	$4 \times 4 \times 64$	0
Reshape	$4 \times 4 \times 64$	0
Dense	256	262400
Dense	2	514
Total params		2,510,274
Trainable params		2,510,274
Non-trainable params		0

TABLE V
THE RESULT OF GLASSBREAK EVENT DETECTION

Model	ER	F
Jujubecake model	0.26	86.9%
Lim_COCAI_task2_1 [15]	0.0480	97.6%
Wang_THU_task2_1 [8]	0.3560	81.0%
Ghaffarzadegan_BOSCH_task2_3 [16]	0.2320	87.9%

model (right in Fig. 4), our Jujubecake model is less likely to output a false positive like the rightmost in Fig. 4.

V. CONCLUSION

We propose a new RNN cell, Jujubecake Cell, which has a better performance than LSTM. Jujubecake Cell, first of all, is an extension of LSTM, with the original LSTM chain structure well preserved, so we can say that there is precisely no loss of the original information flow. Besides, due to the addition of the external LSTM in Jujubcake Cell, not only the association between individual inputs but also the association between blocks that consist of multiple inputs is considered. This improves the model stability. Thus, as demonstrated in the experiments, Jujubecake Cell has better performance than LSTM.

We have to admit that Jujubecake model does not get the best performance as for the experiment results. However, Jujubecake is a novel RNN cell, and it is currently at the very beginning that we have to do many optimisations about it in the future.

VI. ACKNOWLEDGEMENT

This work was supported by the National Natural Science Foundation of China under Grant No. U1836219 and in part supported by Joint Foundation of the Ministry of Education of China and China Mobile under Grant No. MCM20180505.

REFERENCES

[1] N. Kalchbrenner and P. Blunsom, "Recurrent continuous translation models," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 1700–1709.

[2] T. Mikolov *et al.*, "Statistical language models based on neural networks," *Presentation at Google, Mountain View, 2nd April*, vol. 80, p. 26, 2012.

[3] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proceedings of 2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 6645–6649.

[4] S. Hochreiter, "Untersuchungen zu dynamischen neuronalen Netzen," Master's thesis, Institut für Informatik, Technische Universität, Munchen, 1991.

[5] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[7] Y. Shen, K. He, and W. Zhang, "Home activity monitoring based on gated convolutional neural networks and system fusion," DCASE2018 Challenge, Tech. Rep., September 2018.

[8] J. Wang, W. Zhang, and J. Liu, "Transfer learning based DNN-HMM hybrid system for rare sound event detection," DCASE2017 Challenge, Tech. Rep., September 2017.

[9] J. R. A. Moniz and D. Krueger, "Nested LSTMs," in *Proceedings of Asian Conference on Machine Learning*, 2017, pp. 530–544.

[10] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.

[11] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[12] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of English: The Penn Treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[13] M. Abadi, A. Agarwal, P. Barham *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[14] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2980–2988.

[15] H. Lim, J. Park, and Y. Han, "Rare sound event detection using 1D convolutional recurrent neural networks," DCASE2017 Challenge, Tech. Rep., September 2017.

[16] S. Ghaffarzadegan, A. Salekin, S. Das, and Z. Feng, "Bosch rare sound events detection systems for DCASE2017 challenge," DCASE2017 Challenge, Tech. Rep., September 2017.