creative commons
COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

M.S. THESIS

On-device Efficient Acoustic Modeling with Simple Gated Convolutional Networks

심플 게이티트 콘볼루션 네트워크를 이용한 효율적인 온-디바이스 음향 모델링

BY

LEE LUKAS

February 2019

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

M.S. THESIS

On-device Efficient Acoustic Modeling with Simple Gated Convolutional Networks

심플 게이티트 콘볼루션 네트워크를 이용한 효율적인 온-디바이스 음향 모델링

BY

LEE LUKAS

February 2019

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

On-device Efficient Acoustic Modeling with Simple Gated Convolutional Networks

심플 게이티트 콘볼루션 네트워크를 이용한 효율적인 온-디바이스 음향 모델링

지도교수 성 원 용

이 논문을 공학석사 학위논문으로 제출함

2019년 2월

서울대학교 대학원

전기 컴퓨터 공학부

이 루 카 스

이루카스의 공학석사 학위 논문을 인준함

2019년 2월

위 원 장: 최기영
부위원장: 성원용
위 원: 김남수

Abstract

Automatic speech recognition (ASR) is widely adopted for smartphones and many embedded devices in recent years, and neural network based algorithms show the best performance for ASR. While most of ASR systems are based on server-based processing, there is an increasing demand for on-device speech recognition because of privacy concern and low latency processing. Reducing the power consumption is especially important for on-device speech recognition to lengthen the battery life.

Among several neural network models, recurrent neural network (RNN) based algorithms are mostly used for speech recognition, and long short-term memory(LSTM) RNN is most popular because of its superior performance over the other ones. However, executing LSTM RNN demands many DRAM accesses because the cache size of embedded devices is usually much smaller than the parameter size of RNN. Multi-time step parallelization technique computes multiple output samples at a time by fetching one set of parameters, and thus it can reduce the number of DRAM accesses in proportional to the number of time steps computed at a time. However, LSTM RNN does not permit the multi-time step parallelization because of complex feedback structure of the model.

This thesis presents neural network models that support efficient on-device speech recognition. First, a few models that permit multi-time step parallel processing are evaluated. The models evaluated include Gated ConvNet, Diagonal LSTM, and QRNN (quasi RNN). Since the performance of these models are not as good as the LSTM, one-dimensional depthwise convolution is added to improve the performance. The one-dimensional convolution helps finding the temporal patterns of speech signal. Second, Simple Gated Convolution Network (Simple Gated ConvNet) is proposed for improved performance when the parameter count is very small. The Simple Gated ConvNet employs the simplest form of Gated ConvNet. Instead it relies on one-dimensional

convolution for temporal observation. Simple Gated ConvNet supports low-power on-device speech recognition because it can be executed employing multi-time step parallelization. The Simple Gated ConvNet under 3 million even shows better performance than the LSTM with 10 million parameters. In addition, the execution speed in ARM CPU can be increased more than ten-times compared with the LSTM RNN through multi-time step parallelization.

keywords: Speech Recognition, Sequence Modeling, Recurrent Neural Networks(RNN), Embedded Devices

student number: 2017-27205

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 On-device speech recognition: advantages and challenges

Today, most of automatic speech recognition (ASR) is based on neural network based algorithms. Especially, end-to-end speech recognition through neural networks is widely used as a effective method for speech recognition [1, 2, 3, 4]. The end-to-end approach does not need any previous one-to-one mapping between labels and features before training neural networks.

Meanwhile, the demand for on-device speech recognition is increasing. On-device speech recognition here refers to automatic speech recognition (ASR) in which speech recognition is solely performed in users' devices and the user information is not transmitted to the server. While speech recognition is popularly used in embedded devices and smartphones, most of speech recognition is carried out in the servers of service providers after user's speech is delivered. However, this server-based speech recognition causes a concern for security and privacy. This is because user's speech is transmitted to the server. It can be vulnerable to external attack and personal information can leak to the outside.

On-device speech recognition can be a solution in a significant part in security and privacy issues. Privacy invasion can be avoided with on-device speech recognition

because user's speech is not needed to be stored in servers. Moreover, as most of the speech recognition is achieved on local devices, propagation latency due to the network is mostly eliminated. Furthermore, service provider can reduce the need to run numerous servers at great expense and power consumption can be greatly decreased.

However, there exist challenges to implement on-device speech recognition. First, Speech recognition on-device is heavily dependent on the limitations of hardware specification. As neural network models require parameters which claim millions of megabytes (MB). Parameter size needs to be greatly reduced in order to implement speech recognition which mainly utilizes resources of user's local devices.

Second, the number of parameter fetch from main memory should be decreased. As neural networks models require parameters of large sizes, most of the parameters in the models can not be stored in cache memory, however, are stored in main memory. Frequent accesses to the main memory and fetching data lead to the reduced speed of inference in neural networks. Increased power consumption should be under consideration as well.

In server-based ASR, the number of fetch from main memory can be reduced by parallelization using the batch method. Each user corresponds to one batch in this scheme. Neural network operations can be conducted in parallel so that speech recognition is carried out for multiple people at once in a inference stage. In this process, parameters needed are fetched from main memory at once. However, this approach cannot be applied in on-device speech recognition. In on-device speech recognition, there only exists one independent user, therefore, batch-approach does not work well for the number of reduction of memory access.

This problem can be solved by multiple time-step parallelization. A neural network model for sequence modeling essentially require computations over every time-step. We can parallelize the operations at each time-step. Then, the parameters needed for each time-step operation can then be fetched from main memory at once. If sequential processing is not parallelized, main memory must be accessed every time a sample

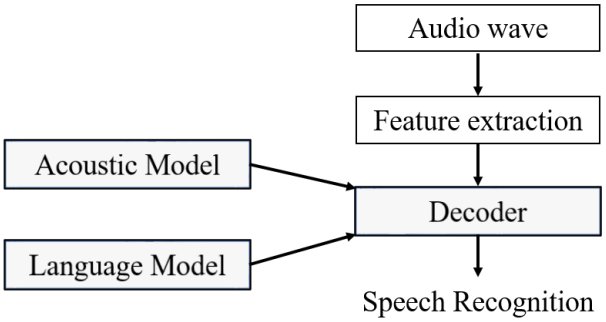


Figure 1.1: The overall procedure of automatic speech recognition.

from each time-step is computed.

In this thesis, we propose a neural network model which consumes low parameters while parallelization is employed in order to reduce the number of memory fetch. It also leads to increased speed of speech recognition. Before we address the proposed model, we explain the components of end-to-end speech recognition first.

1.2 The components of speech recognition

End-to-end speech recognition consists of acoustic model, language model and optional dictionary. Acoustic model represent probability distribution of acoustic features given the labels or transcriptions. Language models expresses the probability distribution between words or characters. Automatic speech recognition(ASR) is a process which finds out correct mapping between acoustic feature vectors and defined tokens. Acoustic model and Language model are ingredient for ASR. Especially, the acoustic models claims the core part of speech recognition as it expresses the relations between acoustic features and human-readable characters or words. In neural network approach, acoustic model is trained neural networks. Figure 1.1 illustrates the overall procedure of ASR.

Suggesting a novel structure for acoustic modeling, we implement on-device ef-

ficient speech recognition. To be specific, we propose the acoustic model which is suitable for on-device speech recognition in terms of parameter size and speed. Performance of the proposed model was better than previous models. Parameter size is much reduced. The reduction of the number of main memory access is achieved by algorithm-level parallelization. The proposed algorithm handles the computation of multiple-time steps in parallel.

1.3 The downsides of RNN based acoustic models

Recurrent neural networks(RNN) has been often employed in training acoustic models and it has been considered as a clever tool for processing sequence information. Among various RNNs, Long short-term memory (LSTM) has been usually employed for sequence modeling [5]. This is because gradient exploding and vanish problem can be avoided to some extent with the introduction of additional state and gates [6].

$$
\begin{aligned}
\mathbf{f}_t &= \sigma(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f), \\
\mathbf{i}_t &= \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i), \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o), \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh\left(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c\right), \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t).
\end{aligned}
\tag{1.1}
$$

However, LSTM RNN is not suitable for multi-time step parallelization, which computes the information of multiple time steps in parallel. Note the equation 1.1. As previous hidden states affect the current state in major computations, samples from multiple time steps can not be computed at once. It should be calculated sequentially. The dependency between current and past states hinders parallel computation. In this case, the memory should be accessed in every time-step. It leads to the speed drop and increased power consumption. Here, there is a need for a parallel-friendly algorithm

to replace LSTM RNN.

1.4 Exploration of efficient on-device acoustic modeling with neural networks

We explore on-device friendly acoustic models which can replace LSTM RNN. This is to solve the problem of LSTM which is difficult to perform multiple time step parallelization in embedded device. In addition, acoustic models for embedded devices should be higher in performance than LSTM RNN in terms of recognition accuracy. We utilize gated convolutional networks (Gated ConvNet), diagonal LSTM and quasi RNN (QRNN) as candidates for the purpose of on-device efficient acoustic modeling [7, 8, 9]. Unlike LSTM, the above models are advantageous for multiple time step parallelization.

Adopting the suggested models, We carry out CTC-based end-to-end speech recognition. when we solely make use of the models, the performance of the models is lower than than of LSTM RNN. Therefore we added the 1-D depthwise convolution to each model's layer. As for QRNN, it was proved that 1-D depthwise convolution with QRNN gives rise to performance jump[10]. The parameter size of 1-D depthwise convolution doesn't matter. The parameter increase from 1-D depthwise convolution is small enough to be stored in cache memory.

Introducing 1-D deptwise convolution leads to the performance grow for all the selected models. Among the models, Gated ConvNet with 1-D depthwise convolution shows the best performance in greedy decoding. Employing deep structures and 1-D depthwise convolution were keys to performance soaring. We further conducted frame-wise phoneme recognition tasks and the performance result of the selected models were verified again. Gated ConvNet with 1-D depthwise convolution shows the best performance in the phoneme classification tasks as well.

We present a execution result of the acoustic models on ARM CPU. By employing

multi-time step parallelization, the execution time of the suggested models shows that at least five times speed-up was achieved. We experimentally showed that it is worth of using on-device efficient acoustic models that we propose.

1.5 Simple Gated ConvNet for small footprint acoustic modeling

We further suggest convolutional neural network(CNN) based algorithms for sequence modeling replacing LSTM. To be specific, we propose simple Gated Convolutional Networks(Simple Gated ConvNet), which is the Simplest form of Gated Convolutional Networks (Gated ConvNet). We utilize Simple Gated ConvNet in training acoustic models. Our model can deal with the limitations for on-device speech recognition described above.

Simple Gated ConvNet is based on Gated ConvNet. Gated ConvNet was originally suggested for the purpose of training language models [7]. Gated ConvNet was also employed in acoustic modeling [11, 12]. However previous usage on Gated ConvNet consumed fairly large parameters for acoustic modeling, which is more than 50 M. Time delay networks which is used in Gated ConvNet adopt the length T. T plays a role of looking at sequential information of a certain length. Therefore it enables the network to process the sequence information. However, the filter length T is also be the main source of the increase in parameter sizes.

In Simple Gated ConvNet, this length T was reduced to the maximum, which is one. Therefore the parameter sizes can be greatly reduced. Of course, this can be a good virtue for on-device speech recognition. After T becomes 1, the model lacks the ability to see the neighboring context. In order to compensate for the performance drop, we additionally employed 1-D depthwise convolution which plays a role of looking at neighboring context. 1-D depthwise convolution is much lighter than original convolution operation in Gated ConvNet in terms of parameter sizes.

The proposed model take advantages over LSTM RNN and conventional Gated ConvNet with several aspects. First, Simple Gated ConvNet outperforms LSTM RNN with respect to word error rates(WER) and character error rate(CER). In other words, Simple Gated ConvNet shows more accurate speech recognition results than LSTM RNN. Second, the parameter size is much reduced. Simple Gated ConvNet with smaller parameter sizes shows better performance than that of LSTM RNN and Gated ConvNet. Third, parallelization over multiple time steps is achieved in Simple Gated ConvNet. This is because there is no dependency between multiple time step computation. Thanks to multiple time step parallelization, the decoding speed becomes much faster experimentally.

1.6 Outline of the thesis

The rest of the thesis is organized as follows. In Chapter 2, the result of diffrent acoustic models are explored. Especially, comparison between acoustic models are performed which can be suitable for on-device devices in terms of multi-time step parallelization. Chapter 3 addresses the details of Simple Gated ConvNet. The structure of Simple Gated ConvNet is illustrated. The performance of Simple Gated ConvNet is compared with LSTM RNN and conventional Gated ConvNet. Chapter 4 concludes the thesis.

Chapter 2

Exploration of Efficient On-device Acoustic Modeling with Neural Networks

There are various acoustic models that can be suitable for on-device speech recognition in terms of reducing the main memory accesses through parallelization. In chapter 2, we explore various acoustic models with neural networks which can be used for on-device speech recognition. The quasi RNNs (QRNNs), Gated ConvNets, and diagonalized LSTMs are considered here. Multiple time-step parallelziation can be carried out in all the listed models. 1-D depthwise convolution is also combined to all the acoustic models in order to achieve performance rise. The performance of these models are compared in Section 2.2. Speed-up of the execution is achieved experimentally as well.

2.1 Acoustic Modeling Algorithms

2.1.1 Diagonal LSTM RNN

In diagonal LSTM RNN, all the recurrent matrix become much lighter than conventional LSTM RNN. To be specific, all the components in $\mathbf{U}_f$, $\mathbf{U}_i$, $\mathbf{U}_o$ and $\mathbf{U}_c$ becomes zero except for diagonal terms [8, 13]. Therefore, matrix multiplication between re-

current weight matrix and input matrix converts to the element-wise multiplication between them. Equation 2.1 shows the operation of diagonal LSTM. $\odot$ denotes the element-wise multiplication.

$$\begin{aligned}
\mathbf{f}_t &= \sigma(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \odot \mathbf{h}_{t-1} + \mathbf{b}_f), \\
\mathbf{i}_t &= \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \odot \mathbf{h}_{t-1} + \mathbf{b}_i), \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \odot \mathbf{h}_{t-1} + \mathbf{b}_o), \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh\left(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \odot \mathbf{h}_{t-1} + \mathbf{b}_c\right), \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t).
\end{aligned} \tag{2.1}$$

We can take advantage of multiple time step parallelization using diagonal LSTM in embedded implementations. As diagonal terms are extracted from reccurent weight matrix, recurrent weights become much smaller. In LSTM RNN, heavy weight matrix for the recurrent function was the major obstacle for the multi-time step parallelization. However, as for diagonal RNN, lighter diagonal terms can be stored in cache memory. There is no need to access the main memory when considering recurrent terms. Therefore, samples of multiple time steps can be fetched in parallel.

$$\begin{aligned}
\hat{\mathbf{x}}_t &= \tanh\left(\mathbf{W}_z \mathbf{x}_t + \mathbf{b}_z\right), \\
[\mathbf{f}_t, \mathbf{i}_t, \mathbf{o}_t] &= \sigma([\mathbf{W}_f, \mathbf{W}_i, \mathbf{W}_o]\mathbf{x}_t + [\mathbf{b}_f, \mathbf{b}_i, \mathbf{b}_o]), \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \hat{\mathbf{x}}_t, \\
\mathbf{h}_t &= \mathbf{o}_t \odot \mathbf{c}_t + (1 - \mathbf{o}_t) \odot \mathbf{x}_t.
\end{aligned} \tag{2.2}$$

2.1.2 QRNN

QRNN does not adopt the feedback computation. In other words, weight matrix for recurrent layer is not used in QRNN [9, 14]. Instead, k input samples, $\mathbf{x}_t, \mathbf{x}_{t-1}, \ldots, \mathbf{x}_{t-k+1}$ are computed in order to consider sequence information. Note that LSTM RNN only utilize one imput sample $\mathbf{x}_t$. As no recurrent weight matrix is used, multiple time-step parallelization can be achieved.

Equation 2.2 shows the simplest form of QRNN. We employ one as a size of k. This is because the parameter size increases linearly proportional to the size k. We try to reduce the parameter size to the maximum.

2.1.3 Gated ConvNet

Gated ConvNet makes use of convolutional neural networks(CNN) for sequence modeling. One layer of Gated ConvNet adopts two separate convolution filters and these filters play a role of observing a certain length of context [7]. As Gated ConvNet employs no recurrent path, multiple time step parallelization can be easily applied. Equation 2.3 illustrates the operation of Gated ConvNet.

$$\mathbf{h}(\mathbf{X}) = (\mathbf{X} * \mathbf{W} + \mathbf{b}) \odot \sigma(\mathbf{X} * \mathbf{V} + \mathbf{c}) \tag{2.3}$$

where $*$ denotes convolution operation and $\odot$ represents element-wise multiplication. $\mathbf{W}, \mathbf{V} \in \mathbb{R}^{T \times D \times D}$ and $\mathbf{b}, \mathbf{c} \in \mathbb{R}^{D}$ are trainable variables where D is the channel of the network. $\mathbf{X} \in \mathbb{R}^{N \times D}$ is the input vector where N is the sequence length.

2.2 Experiments

The performance of various models were experimented on two acoustic modeling tasks. One is Connectionist Temporal Classification(CTC) algorithm based speech

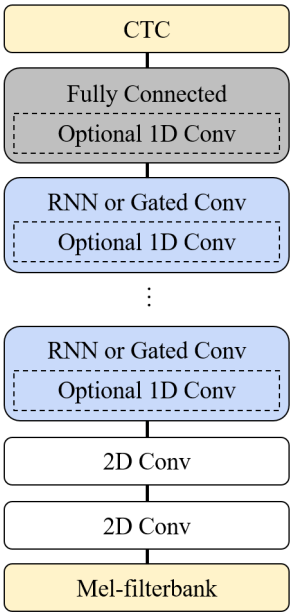


Figure 2.1: Acoustic modeling architecture for end to end speech recognition.

recognition, in which the unit of the labels is the character [1, 2, 15]. The other is frame-wise phoneme classification task, in which the unit of the labels is the phoneme. The main difference of these two task is whether pre-alignment between labels and acoustic features is needed. In CTC based algorithm tasks, pre-alignment between characters and acoustic features is not required. This is because the alignment is done automatically during the training stage. In frame-wise phoneme classification, alignment between labels and acoustic features should be conducted before the training stage. The Wall Street Journal (WSJ) corpus [16] was employed for CTC-based speech recognition tasks. As for the phoneme recognition task, TIMIT dataset was used.

2.2.1 End-to-end speech recognition

For CTC-based end-to-end speech recognition, a 40-dimensional log Mel-frequency filter-bank feature was extracted from raw wav files of WSJ corpus. 25 ms-sized Ham-

Table 2.1: WER and CER (%) evaluated on WSJ eval92. The models are trained on SI-284.

Models	**Params.**	**CER**	**WER**
4x600 LSTM	11.5M	**7.29**	**24.88**
6x800 QRNN (k = 1)	11.5M	12.70	45.22
6x700 Diagnoal LSTM	11.5M	8.97	30.40
6x300 Gated ConvNet ($T = 15$)	16.2M	8.02	28.65
6x300 Gated ConvNet ($T = 7$)	7.7M	8.52	30.56

ming window is used to sample the feature vectors every 10 ms. Then, the extracted filter-bank features were employed for the inputs of two-layered 2-D CNN layer. The two layered 2-D Convolutional Neural Networks (CNN) is used for down-sampling the input frames by two. The down-sampling of feature frames contributes to reducing the arithmetic complexity. Batch normalization and variational dropout are also adopted for the two layered CNN for regularization [17, 18].

The output of two-layered CNN becomes the input of the various acoustic models such as Gated ConvNet, QRNN and LSTM. 1-D depthwise convolution is optionally added to the one layer of the after selected models. The one layer of the models with optional 1-D depthwise convolution becomes the one component of the multi-layered network. The fully connected is added at the end of the network. The overall network is used to train CTC loss function. Figure 2.1 illustrates the whole network.

Experimental results are shown in the following tables. Most of the experiments were conducted with WSJ SI-284, which has 81 hours of speech. WSJ-ALL, which contains 150 hours of speech was also used if needed. The number of layers and the width of each layer are also shown. For example, 4x400 LSTM denotes the four-layered LSTM which has the hidden dimension of 400. As for Gated ConvNet, T

Table 2.2: WER and CER (%) evaluated on WSJ eval92 with 1-D convolution.

Models	**Params.**	**CER**(%)	**WER**(%)
4x600 LSTM, 1-D conv	11.5M	6.95	23.57
6x700 QRNN ($k = 1$), 1-D conv	11.5M	5.26	19.07
6x700 Diagonal LSTM, 1-D conv	11.5M	7.57	23.90
6x300 Gated ConvNet ($T = 15$), 1-D conv	16.2M	7.58	27.00
6x300 Gated ConvNet ($T = 7$), 1-D conv	7.7M	6.57	24.20
20x300 Gated ConvNet ($T = 2$), 1-D conv	7.5M	5.55	19.70
30x300 Gated ConvNet ($T = 2$), 1-D conv	11M	**4.73**	**17.00**

denotes the number of context length that each Gated ConvNet layer observes. QRNN with k represents that k input samples, $\mathbf{x}_t, \mathbf{x}_{t-1}, \ldots, \mathbf{x}_{t-k+1}$, which were used in computation in QRNN. Performance was evaluated using character error rate(CER) and word error rate(WER).

In table 2.1, the results of greedy decoding are shown. Greedy decoding means that no language model is augmented for evaluation. The performance of acoustic models can be solely compared by greedy decoding. The results of LSTM, QRNN, Diagonal LSTM and Gatd convNet are shown in the table. 1-D depthwise convolution is not applied in this table. Among the selectd models, LSTM RNN yields the best performance. QRNN shows the worst performance. Note that QRNN never adopts any local feedback from previous the hidden state. It can be seen that the recurrent path plays an important role in performance.

In table 2.2, we add 1-D depthwise convolution to each selected model. We try to increase the performance of each model by adding 1-d depthwise convolution. The parameter increase is not significant when 1-d depthwise convolution is combined. The time-length of 1-D convolution filter is 15, from -7 to +7, which means that the

Table 2.3: WER and CER (%) on WSJ eval92. The models are trained on WSJ SI-ALL.

		Greedy		HCLM	
Models	**Params.**	**CER**	**WER**	**CER**	**WER**
4x600 LSTM, 1-D conv	11.5M	5.91	20.14	2.71	6.56
6x700 QRNN ($k = 1$), 1-D conv	11.5M	4.13	18.02	**1.51**	**3.73**
30x300 Gated ConvNet ($T = 2$), 1-D conv	11M	**3.30**	**11.60**	1.53	3.86
Deep Speech 2 [4]	100M	WER 3.60 with 5-gram LM			

filter consider seven time steps in the past, seven time steps in the future. When 1-D depthwise convolution is combined, the performance of LSTM RNN jumped from 24.88% to 23.57%. Compared to LSTM, Gated ConvNet, QRNN and diagonal LSTM shows remarkable performance rise.

As for diagonal LSTM, the result is similar to that of LSTM RNN with 1-D depthwise convolution. As multi-time step parallelization can be achieved in diagonal LSTM, diagonal LSTM with 1-D depthwise convolution is much more beneficial than conventional LSTM. The performance of QRNN goes through drastic change when it is augmented with 1-D deptwise convolution. The word error rate improved from 45.22 % to 19.70 % when 1-D depthwise is combined. Gated ConvNet also shows increase in performance.

Introduction of deep structure contributes to the performance skyrocketing. The WER in Gated ConvNet drops to 17.00 %. It marks the best performance. Although, layers get deeper, total parameter size is not that large. This is because the filter length of T of Gated ConvNet is reduced. Instead, 1-D convolution filter, whose length is 15, contributes to processing sequence information more.

The performance of the suggested models are also trained in larger dataset, WSJ-ALL. The result of trained end-to-end acoustic models are listed in table 2.3. All the

Table 2.4: Frame-wise phoneme classification accuracy (%) on TIMIT.

Models	**Params.**	**Accruacy**
2x256 LSTM	0.92M	72.00
4x256 QRNN ($k = 1$)	0.92M	52.74
4x256 Diagonal LSTM	0.92M	68.45
4x256 Gated ConvNet ($T = 2$)	0.91M	62.00
2x256 LSTM, 1-D conv	0.93M	75.30
4x256 Diagonal LSTM, 1-D conv	0.93M	74.60
4x256 QRNN ($k = 1$), 1-D conv	0.93M	76.48
4x256 Gated ConvNet ($T = 2$), 1-D conv	0.92M	76.30
10x128 Gated ConvNet ($T = 2$), 1-D conv	0.67M	**77.04**

models were evaluated on WSJ eval92 testset. Furthermore, the beam decoding was conducted with with the support of language model. Beam width was set as 128. As a language model, hierarchical character language model (HCLM) is chosen [19, 20]. The HCLM was trained using the training text of WSJ corpus. In the result of both QRNN and Gated ConvNet augmented with 1-D depthwise convolution, WER below 4% was aquired. Although we spent about 10 times less number of parameters, the result is close to the result of Deep speech 2 [4].

2.2.2 Phoneme classification

Frame-wise phoneme classification on TIMIT dataset was performed in order to verify whether the proposed models with 1-D convolution work well in the non-CTC speech recognition tasks. Training TIMIT dataset for phoneme classification takes much less time than CTC based end-to-end speech recognition.

Table 2.5: Frame-wise phoneme classification accuracy (%) of different Gated ConvNets on TIMIT.

Models	**Params**	**Accuracy**
4x256 Gated ConvNet ($T = 2$), 1D conv (-3,+3)	0.92M	76.44
4x256 Gated ConvNet ($T = 2$), 1D conv (-5,+5)	0.92M	76.30
4x256 Gated ConvNet ($T = 2$), 1D conv (-7,-7)	0.92M	73.90
10x128 Gated ConvNet ($T = 2$), 1D conv (-3,+3)	0.67M	**78.04**
10x128 Gated ConvNet ($T = 2$), 1D conv (-5,+5)	0.67M	77.04

We made use of a 40-dimensional log Mel-frequency filter-bank feature as we did in the end-to-end speech recognition task. Note that previous research on phoneme recognition utilized the Mel-Frequency Cepstral Coefficients (MFCCs) [21]. Total 61 classes of phonemes were employed for the training, and they were folded into 39 classes for evaluation. This way of evaluation was done in the same way as conducted in previous studies [22]. The parameter sizes were limited to 1 M to be compared with the previous results [21].

In table 2.4, the accuracy of frame classification is shown. 11 of the context length of 1-D depthwise convolution was chosen. The result of LSTM RNN is better than other models without 1-D depthwise convolution. However, after 1-D depthwise convolution is augmented, the performance of Gated ConvNet ourperforms LSTM RNN which is combined with 1-D depthwise convolution. QRNN with 1-D depthwise convolution shows similar performance as Gated ConvNet with 1-D depthwise convolution. The accuracy of LSTM doesn't improve much when combined with 1-D depthwise convolution. We can find out that 1-D depthwise convolution contributes to performance improvement. Moreover, gated ConvNet and QRNN With 1-D depthwise

Table 2.6: Execution time (sec) for models for 1 sec speech, function of T_P

Models	$T_P = 1$	$T_P = 4$	$T_P = 8$	$T_P = 16$
4x600 LSTM	1.46			
6x700 QRNN ($k = 1$), 1-D conv	1.21	0.39	0.20	0.15
6x700 Diagonal LSTM	1.55	0.47	0.25	0.18
30x300 Gated ConvNet, 1-D conv	1.85	0.47	0.29	0.20

convolution outperformed LSTM RNN. The performance trend of acoustic modeling on CTC-based acoustic modeling was maintained in frame-wise classification tasks as well. Table 2.5 shows the result of Gated convNet with various configurations on phoneme classification task. "1D conv(-3,3)" means that it shows the context of the past three time-steps and the future three time-steps.

2.2.3 Implementation Results on Embedded Systems

Multi-time step parallelization is difficult with LSTM RNN. Parameters for feedback to the previous state in LSTM is too large to be stored in cache-memory. It means that main memory should be accessed in every time-step. However, the proposed models such as QRNN, Gated ConvNet with 1-D depthwise convolution are suitable for the multi-time step parllelization.

Table 2.6 shows the result of multi-time step parallelzation on the proposed models. ARM Cortex-A57 based embedded system was used to measure the execution time. The result shown is the execution time of CTC-based end-to-end acoustic models. Multi-time parallel steps of 1, 4, 8, 16 were chosen. As LSTM is not suitable for multi-time step parallelization, execution time of sequential processing is represented.

The execution time of QRNN and Gated ConvNet was 7.30 times and 5.03 times faster than LSTM RNN each when the parallel steps of 8 is adopted.

The result shows that fast inference of acoustic model can be achieved in embedded devices using the multiple time step parallelization. In this respect, parallelization friendly algorithms such as diagonal RNN, Gated ConvNet and QRNN has advantages over LSTM RNN.

2.3 Concluding Remarks

LSTM RNN based acoustic modeling is in-efficient for speech recognition on embedded devices. DRAM should be accessed every time-step when using LSTM RNN so that model parameter is fetched from DRAM at every inference. Multiple time step parallelization can not be applied because each state in LSTM RNN has dependency on previous time steps in major computation. It is the main bottleneck for fast and energy-efficient on-device acoustic modeling.

In order to overcome the setbacks of LSTM-RNN, we explore various acoustic models which are suitable for multiple-time step parallelization. And it leads to more efficient acoustic modeling than LSTM-RNN. Diagonal LSTM, the quasi RNN(QRNN) and Gated ConvNet were employed as candidates for on-device friendly acoustic models. These models allows recurrent paths to be enough small so that cache-memory can afford them. In addition, introduction of 1-D depthwise convolution makes some of the suggest models outperforms conventional LSTM-RNN.

Experiments were conducted in end-to-end speech recognition and frame-wise phoneme classification tasks. In most of the cases, Gated ConvNet with 1-D deptwise convolution was better than any other models in terms of word error rate. The execution time of the presented models were measured in embedded devices. The suggested models present the speed-up of over 500 % when parallel step of 8 is adopted.

In this chapter, we explored the neural networks models which are more advanta-

geous than conventional LSTM RNN for sequence modeling with respect to on-device efficient speech recognition.

Chapter 3

Simple Gated Convolutional Networks for small footprint acoustic modeling

We propose Simple Gated ConvNet which outperforms LSTM RNN and which is suitable for multi-time step parallelization. The parameter size needed in acoustic models is also greatly reduced, which can be a benefit for efficient on-device speech recognition.

3.1 Simple Gated ConvNet

3.1.1 Gated ConvNet

Gated ConvNet combines stacked convolution with gating mechanism for sequence modeling [7]. The main structure of Gated ConvNet is represented as follows.

$$\mathbf{h}^l(\mathbf{X}^l) = (\mathbf{X}^l * \mathbf{W}^l + \mathbf{b}^l) \odot \sigma(\mathbf{X}^l * \mathbf{V}^l + \mathbf{c}^l), \tag{3.1}$$

where $\odot$ is element-wise multiplication, σ represents sigmoid function and $*$ indicates convolution operation. The network is composed of L layers, and $l = 0, 1, 2, 3, ..., L-$

1. $\mathbf{W}^l, \mathbf{V}^l \in \mathbb{R}^{T\times 1\times D\times D}$ and $\mathbf{b}^l, \mathbf{c}^l \in \mathbb{R}^D$ are learned parameter where T is the filter length, and D denotes the feature dimension.

Note that the shape of $\mathbf{W}^l$ and $\mathbf{V}^l$ is $[T, 1, D, D]$.The size of the model increases in proportional to the length T, given that D is fixed. T can be interpreted as the context length at each layer. Note that sequence learning with a finite range of context was also investigated in time-delay neural networks(TDNN) [23, 24]. Processing a certain length of sequences in Gated ConvNet can be thought like observing a temporal context in TDNN. Both view finite range of sequences through temporal convolution operations.

It is inevitable to employ a certain length of T in order to observe adjacent sequence information. For acoustic modeling, conventional Gated ConvNet also adopted a certain length of T. In [12], the length of T range from 13 to 28 was chosen. As we go to deeper layers, T increases gradually. However, employing a certain length of T is one of the major source for the number of parameters skyrocketing. It serves as a obstacle for on-device usage.

3.1.2 Simple Gated ConvNet

Simple Gated ConvNet set the length T to one. It reduces the number of parameters needed in each convolution filter $\mathbf{W}^l$ and $\mathbf{V}^l$ from TDD to DD. In other words, convolution filters $\mathbf{W}^l$, $\mathbf{V}^l$ in Gated ConvNet does not cover neighboring context at once anymore. As separate two convolution filter $\mathbf{W}^l$, and $\mathbf{V}^l$ are employed in Gated ConvNet, 2T times decrease in parameter size is achieved. The reduction of T gives rise to huge performance drop obviously, which is described in Section 3 experimentally. Since we do not count consecutive context at a time when T is one, we need an additional means so that networks consider neighboring context.

Therefore, 1-D depthwise convolution is adopted. The 1-D depthwise convolution plays the role of looking at a certain length of sequences while using the lighter parameters. Depth-wise convolution has been employed in the area of vision, machine

translation and speech recognition recently[25, 26, 27, 10]. The operation of 1D depth-wise convolution is represented as follows.

$$\mathbf{h}^l_{t,1,d} = \sum_{i=\lfloor -T/2 \rfloor}^{\lfloor T/2 \rfloor} \mathbf{F}^l_{i,1,d} \mathbf{X}^l_{t+i-1,1,d} \tag{3.2}$$

Eq. (3.2) shows the convolution operation at the t-th time step. $\mathbf{F}^l \in \mathbb{R}^{T \times 1 \times D}$ is the filter matrix of 1-D depthwise convolution. The number of parameters is now TD which is needed for the convolution filter $\mathbf{F}^l$. We put the 1D depthwise convolution before and after each original convolution layer whose parameter size is reduced from $2TDD$ to $2DD$. Therefore, Now the number of parameters needed in Simple gated ConvNet can be modeled as approximately as $2DD + TD$. Note that the proposed model reduced parameter size magnificently compared to the $2TDD$ of original Gated ConvNet.

Furthermore, we carry out convolution operation in a way to cover neighboring features at a time. We introduce additional operation in the direction of feature dimensions while performing 1-D depth-wise convolution. We keep the input and output sizes by putting zero pad in both ends of Feature dimension. The operation of 1-D depth-wise convolution with width K is represented as follows.

$$\mathbf{h}^l_{t,1,d} = \sum_{w=\lfloor -K/2 \rfloor}^{\lfloor K/2 \rfloor} \sum_{i=\lfloor -T/2 \rfloor}^{\lfloor T/2 \rfloor} \mathbf{F'}^l_{i,1,d,w} \mathbf{X}^l_{t+i-1,1,d+w} \tag{3.3}$$

The proposed method processes the input channels using filters with the width of K. By doing so, our model can consult the correlation between the neighboring features which are expressed as adjacent channels.

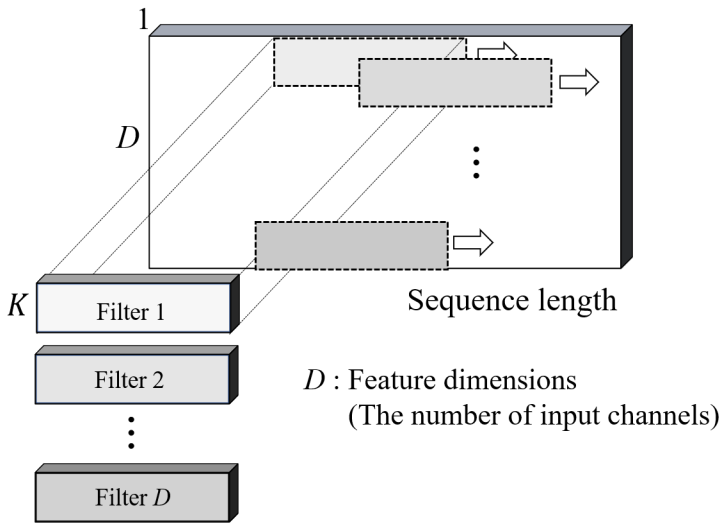
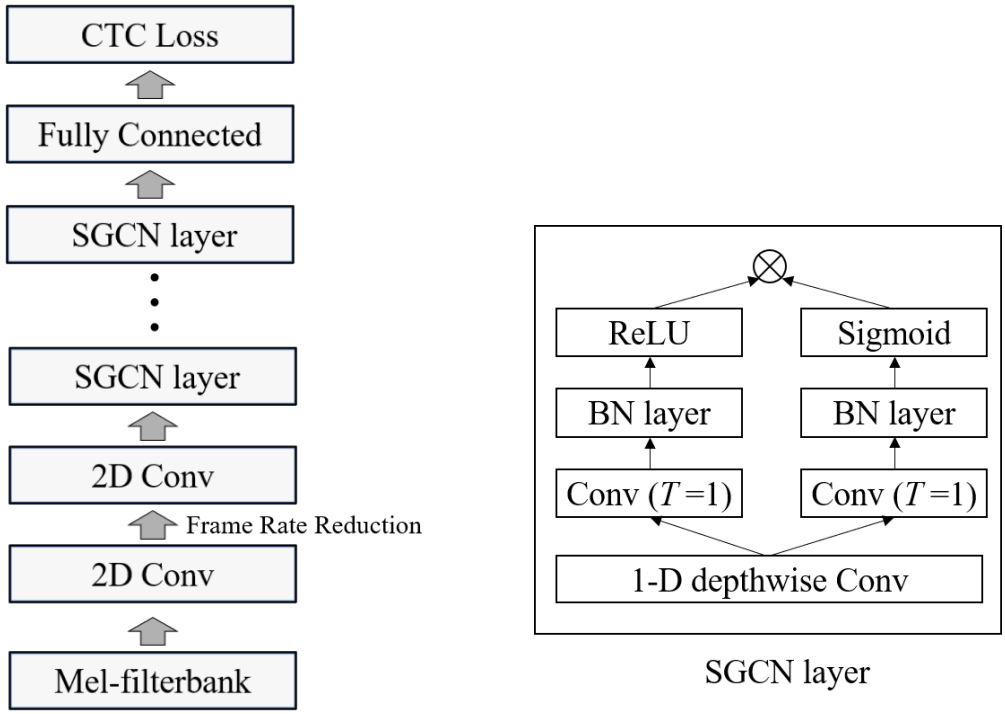


Figure 3.1: The operation of 1-D depthwise convolution with the width of K.

The parameter size of $\mathbf{F}'^{l}$ increases K times compared to the $\mathbf{F}^{l}$ of 1D depthwise convolution, which is from TD to TDK. We assign a number from 1 to 21 to K. At the moment, the the number of parameters required can be modeled as $2DD + TDK$. As the size of D is about several hundreds, parameter size growth is still slighter than TDD of original Gated ConvNet. Figure 3.1 shows the operation of 1-D depthwise convolution with the width of K.

Introduction of batch normalization [17] layers is another characteristic of our research. Previous research on Gated ConvNet rarely utilized batch normalization [7, 12, 11]. It turns out that batch normalization also contributes to the performance improvement in Gated ConvNet under our configurations. We arrange batch normalization right after convolution operations and before activation. This arrangement was mainly influenced by the concept of ResNet[28]. $\mathbf{X}^{l} * \mathbf{W}^{l} + \mathbf{b}^{l}$ of Gated ConvNet does not have an activation function. We added ReLU to $\mathbf{X}^{l} * \mathbf{W}^{l} + \mathbf{b}^{l}$. On the other hand, $\mathbf{X}^{l} * \mathbf{V}^{l} + \mathbf{c}^{l}$ already has activation function for gating mechanism.

We combine the above all elements into one layer. We also stack several layers



(a) Simple Gated ConvNet

(b) A Simple Gated ConvNet layer

Figure 3.2: The structure of Simple Gated ConvNet.

to complete the entire network structure. Residuals connection prevents gradient from becoming zero. Figure 3.2 addresses whole structure of Simple Gated ConvNet.

3.2 Experiments

3.2.1 Experiment Setups

Simple Gated ConvNet is adopted for training acoustic models with connectionist temporal classification (CTC) algorithm [2, 15]. We compared performance of Simple Gated ConNet with that of LSTM RNN and Gated ConvNet under various configurations. The Wall Street Journal (WSJ) Corpus was employed to train the acoustic

models [16].

Feature extraction for training was processed as follows. The initial inputs were raw wav files in WSJ corpus. In order to extract features, Discrete Fourier Transform (DFT) was performed every 10 ms with a 25 ms Hamming window. The samples extracted through DFT are passed through filter-banks. Then, we acquired a 40-dimensional log Mel-frequency filter-bank feature.

After features were extracted, we scaled down the sequence lengths by Convolutional Neural Networks (CNN). According to [4], down sampling on acoustic features by CNN contribute to performance rise with respect to training speed and error rates. We put this technique into practice. 40-dimensional log Mel-frequency filter-bank features becomes the inputs of the two-layered CNN. Dropout was adopted and batch Normalization layers are applied in each layer [29]. Sequence lengths were pooled down after passing two-layered CNN.

The output of this two-layered CNN was connected to Simple Gated ConvNet. On the same condition, LSTM RNN and Gated convNet were linked to the CNN layers for performance comparison. WSJ Si-284, the dataset of 81 hours of speech was mainly used for training. WSJ Si ALL, the dataset of 150 hours of speech was also adopted for the comparison of selected models. All the training results were evaluated on WSJ eval92 dataset. Because the CTC algorithm was used, pre-alignment between labels and features were not needed.

3.2.2 Experimental Results

The performance comparison between Simple Gated ConvNet and other models are shown in the following tables. CER(character error rate) and WER(Word Error Rate) was chosen for performance measurement. The model size is represented in the form of ($n_{layer} \times n_{dim}$). n_{layer} is the number of layers. n_{dim} denotes the size of a hidden dimension in LSTM and the width of a feature map in the case of Gated ConvNet and Simple Gated ConvNet. Two-layered CNN before the proposed models is not counted

in this notation. For example, 4x300 LSTM represents four-layered LSTM which has 300 hidden dimensions in each layer. 6x300 Simple Gated ConvNet represents six-layered Simple Gated ConvNet whose channel size is 300.

Table 3.1: WER and CER in percentage trained in WSJ Si-284 and evaluated on WSJ eval92 test set. SGCN is Simple Gated ConvNet. GCN is short of Gated ConvNet.

		Greedy	
Model	**Params.**	**CER**	**WER**
4x300 LSTM	2.95M	8.18	27.6
4x300 LSTM, 1-D depthwise	2.95M	7.13	24.7
4x575 LSTM, 1-D depthwise	10.01M	6.31	21.9
4x180 LSTM, Bidirectional	3.08M	5.61	19.3
6x300 GCN ($T = 15$)	16.38M	8.01	28.6
6x300 GCN ($T = 4$), 1-D depthwise	4.43M	6.37	23.1
6x300 GCN ($T = 2$), 1-D depthwise	2.25M	6.69	24.1
12x220 GCN ($T = 2$), 1-D depthwise	2.47M	5.67	20.1
6x300 SGCN	1.16M	7.03	25.5
12x300 SGCN	2.24M	**5.32**	**18.8**

In Table 3.1, The CER and WER performance of LSTM RNN, Gated ConvNet and Simple Gated ConvNet are shown. All the model were trained in WSJ Si-284 and evaluated in WSJ eval 92. Greedy decoding, without any support of language model, was performed. Simple gated ConvNet outperforms LSTM under the parameter 3M. WER differences is over 6%. Simple gated ConvNet surpasses LSTM when even 1-D depthwise convolution is combined with LSTM. Simple Gated ConvNet showed good performance even when the parameter size is relatively smaller than LSTM. Simple Gated ConvNet with 3M parameters defeats LSTM with 1-D depth-wise convolution whose parameter size is 10M. The performance of Simple gated convNet was better than other Gated ConvNet which has longer T, given similar number of parameters.

Note that the performance of Simple Gated ConvNet with 2.24M surpasses that of the Bidirectional LSTM with 3.08M. Considering bidirectional LSTM has latency which results from observing a whole sequence in advance, Simple Gated ConvNet can be a better option for real-time speech recognition.

Table 3.2: WER and CER in percentage with various width of 1-D depthwise convolution, trained in WSJ Si-284 and evaluated on WSJ eval92 test set.

		Greedy	
Model	**Params.**	**CER**	**WER**
12x300 SGCN ($K = 1$)	2.24M	5.32	18.8
12x300 SGCN ($K = 3$)	2.33M	5.09	17.8
12x300 SGCN ($K = 5$)	2.42M	5.14	18.3
12x300 SGCN ($K = 7$)	2.50M	5.10	17.8
12x300 SGCN ($K = 11$)	2.68M	4.93	17.5
12x300 SGCN ($K = 15$)	2.85M	4.96	17.7
12x300 SGCN ($K = 21$)	3.10M	**4.75**	**16.8**

In Table 3.2, improved performance in Simple Gated ConvNet is obtained by increasing the width K. Although performance does not increase linearly, higher K tends to contribute to performance improvement to some extent. It shows that we can take advantage of better acoustic modeling by viewing adjacent features at a time. When K is 21, the best WER is obtained. Considering the increase in the number of parameters, we adopt K of 5 or 11 in the following experiments.

In Table 3.3, models were trained in WSJ Si-ALL, which is larger dataset than WSJ Si-284. The 1-D filter width, K of 1 is adopted for LSTM with 1-D depthwise con-

Table 3.3: WER and CER in percentage trained in larger dataset (WSJ Si-ALL) and evaluated on WSJ eval92 testset.

	Greedy		HCLM	
Model	**CER**	**WER**	**CER**	**WER**
12x300 SGCN ($K = 11$)	3.74	13.7	1.62	4.04
4x300 LSTM, 1-D depthwise	5.07	17.7	2.40	5.76

Table 3.4: WER and CER in models with 1 M parameters, evalutated on WSJ eval92.

(a) Trained in WSJ Si-284.

		Greedy	
Model	**Params.**	**CER**	**WER**
4x160 LSTM	0.96M	10.78	37.1
4x160 LSTM, 1-D depthwise	0.97M	9.64	33.6
12x190 SGCN ($K = 5$)	0.99M	**6.11**	**22.0**

(b) Trained in larger dataset(WSJ Si-ALL).

		Greedy	
Model	**Params.**	**CER**	**WER**
4x160 LSTM	0.96M	8.64	31.2
4x160 LSTM, 1-D depthwise	0.97M	6.72	24.1
12x190 SGCN ($K = 5$)	0.99M	**5.11**	**18.8**

volution. Trained with larger data, Simple Gated ConvNet shows much better performance than LSTM with 1-D depthwise convolution. The decoding was also performed using language model. As a language model, Hierarchical Character Level Language Model(HCLM) was employed [19, 20]. The HCLM combines the advantages of word-level language model(WLM) and character-level language model(CLM). Even after language model is integrated, Simple Gated ConvNet is superior to 1-D depthwise convolution-augmented LSTM.

In Table 3.4, the model sizes are under 1M. The 1-D filter width, K of 1 is chosen for LSTM with 1-D depthwise convolution. The performance of Simple Gated ConvNet exceeds that of LSTM with 1-D convolution when the parmeter size is limited under 1 M. This trend is maintained in the larger dataset. Note that the performance drop in Simple Gated ConvNet is much smaller than that of LSTM when the parameter size is reduced to extremely low. Simple Gated ConvNet shows robust performance compared to LSTM RNN when the parameter size plunges.

Table 3.5: WER and CER in percentage with models for low latency tasks, trained in WSJ Si-284 and evalutated on WSJ eval92.

		Greedy	
Model	**Params.**	**CER**	**WER**
12x300 SGCN (100 ms latency)	2.68M	6.20	22.4
12x300 SGCN (200 ms latency)	2.68M	5.01	18.1
12x300 SGCN (1200 ms latency)	2.68M	4.93	17.5

Table 3.5 shows the performance of Simple Gated ConvNet on low latency tasks. Low latency model is achieved by using causal convolution, which only views past

Table 3.6: Execution time in second for processing 1 sec speech, function of T_p.

	T_p			
Model	1	4	8	16
4x275 LSTM	0.459	-	-	-
12x300 SGCN	0.363	0.095	0.061	0.043

time-steps [30, 31]. Employing causal convolution prevents latency as future time-steps are never seen by the operation of convolution. As for the proposed model, the filter size T of 1-D depthwise convolution is 11. the filter covers the input from t-5 to t+5, where t-5 denotes five-step past and t+5 represents five-step future. Therefore, one non-causal layer of Simple Gated ConvNet give rise to the latency of five steps. As one time-step is about 20ms, the latency of 100 ms occurs from one Simple Gated ConvNet layer. As we stack non-causal layers, the amount of latency escalates. The increased latency disturbs real-time speech recognition.

We reduce the amount of latency by replacing some non-causal layers with causal layers. The result of 1000 ms, 100ms and 200ms are shown in table 3.5. Note that one non-causal layer is the source of 100 ms delay. As we adopt 12-layered Simple Gated ConvNet, total 1200 ms delay occurs when all the layers are non-causal. The latency of total 100 ms is obtained by employing one non-causal layer and remaining layers are all causal. In the same way, we get the latency of 200 ms by choosing two non-causal layers and remaining all-causal layers. We can notice that the model of 200 ms delay shows small performance degradation compared to model of 1200 ms delay with all non-causal layers. The result of low latency tasks shows that Simple Gated ConvNet works well under low-latency configuration. And it can also be suitable for real-time recognition in terms of low latency.

The result of multi-time step parallelization is shown in table 3.6. As Simple Gated ConvNet adopts no recurrent path, the computation of samples from multiple time step

can be easily computed. Simple Gated ConvNet shows was 10 times faster than LSTM RNN When the parallel step of 16 was chosen.

3.3 Concluding Remarks

We propose Simple Gated ConvNet for end-to-end speech recognition. Simple Gated Net is on-device friendly compared to LSTM RNN. Simple Gated ConvNet outperforms LSTM based RNN in acoustic modeling. The proposed model significantly reduces the number of parameters needed compared to Gated ConvNet. Our model is also suitable for multi-time step parallelization as no recurrent path exists. These characteristics make the simple gated convNet advantageous for speech recognition over LSTM based RNN in embedded devices.

We show experimentally that Simple Gated ConvNet has much better performance than LSTM RNN and Gated ConvNet under 3 M parameters. Furthermore, Simple Gated ConvNet of 3 M shows superior performance than LSTM of 10 M. Simple Gated ConvNet shows robust performance in various tasks. When the parameter size was reduced to even 1M, simple Gated ConvNet shows affordable performance. Performance degradation of Simple Gated ConvNet was small even in the low latency tasks. Above all, 10 times speed-up over LSTM RNN is achieved through multi-time step parallelization.

Chapter 4

Conclusions

We tried to build an on-device automatic speech recognition (ASR) which can be performed in embedded devices independently. Major approach was finding neural network based acoustic models which is suitable for multiple time step parallelization. Conventional LSTM RNN is not suitable for multiple time step parallelization as it has large recurrent paths.

In chapter 2, we explored various acoustic models which can be on-device friendly. Gated Convolutional Networks(Gated ConvNet), the quasi RNN(QRNN) and diagonal LSTM are adopted. They have more advantages than LSTM RNN in execution time. Multiple time step parallelization technique allows these models to make at least 5 times speed-up in embedded devices. Introduction of 1-D depthwise was main source of performance rise and Gated ConvNet with 1-D depthwise convolution yields the best result in greedy decoding.

In chapter 3, we proposed Simple Gated Convolutional Network(Simple Gated ConvNet). We examined the structure and performance of Simple Gated ConvNet. We showed how this structure is more advantageous than LSTM RNN for on-device speech recognition. Parameter size was remarkably reduced compared to Gated ConvNet. The performance in Simple Gated ConvNet was much better than LSTM RNN and Gated ConvNet in terms of error rate. 10-times speed-up was obtained compared

to LSTM RNN.

The thesis was a journey to find out how to perform speech recognition efficiently in embedded device in algorithm level. Out method and the proposed model can be combined with other techniques like network compression in order to achieve more efficient on-device speech recognition.

Bibliography

[1] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE, 2013, pp. 6645–6649.

[2] Alex Graves and Navdeep Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *International Conference on Machine Learning*, 2014, pp. 1764–1772.

[3] Yajie Miao, Mohammad Gowayyed, and Florian Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*. IEEE, 2015, pp. 167–174.

[4] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al., "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International Conference on Machine Learning*, 2016, pp. 173–182.

[5] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[6] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio, "On the difficulty of training recurrent neural networks," in *International Conference on Machine Learning (ICML)*, 2013, pp. 1310–1318.

[7] Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier, "Language modeling with gated convolutional networks," in *International Conference on Machine Learning (ICML)*, 2017, pp. 933–941.

[8] Y Cem Subakan and Paris Smaragdis, "Diagonal RNNs in symbolic music modeling," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2017 IEEE Workshop on*. IEEE, 2017, pp. 354–358.

[9] James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher, "Quasi-Recurrent Neural Networks," *International Conference on Learning Representations (ICLR 2017)*, 2017.

[10] Jinhwan Park, Yoonho Boo, Iksoo Choi, Sungho Shin, and Wonyong Sung, "Fully neural network based speech recognition on mobile and embedded devices," in *Advances in Neural Information Processing Systems (NIPS)*, 2018.

[11] Jian Kang, Wei-Qiang Zhang, and Jia Liu, "Gated convolutional networks based hybrid acoustic models for low resource speech recognition," in *Automatic Speech Recognition and Understanding Workshop (ASRU), 2017 IEEE*. IEEE, 2017, pp. 157–164.

[12] Vitaliy Liptchinsky, Gabriel Synnaeve, and Ronan Collobert, "Letter-based speech recognition with gated ConvNets," *arXiv preprint arXiv:1712.09444*, 2017.

[13] Shuai Li, Wanqing Li, Chris Cook, Ce Zhu, and Yanbo Gao, "Independently recurrent neural network (indrnn): Building a longer and deeper RNN," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5457–5466.

[14] David Balduzzi and Muhammad Ghifary, "Strongly-typed recurrent neural networks," in *International Conference on Machine Learning*, 2016, pp. 1292–1300.

[15] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *International Conference on Machine Learning (ICML)*. ACM, 2006, pp. 369–376.

[16] Douglas B Paul and Janet M Baker, "The design for the Wall Street Journal-based CSR corpus," in *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, 1992, pp. 357–362.

[17] Sergey Ioffe and Christian Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning (ICML)*. JMLR, 2015, pp. 448–456.

[18] Yarin Gal and Zoubin Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Advances in neural information processing systems*, 2016, pp. 1019–1027.

[19] Kyuyeon Hwang and Wonyong Sung, "Character-level incremental speech recognition with recurrent neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 5335–5339.

[20] Kyuyeon Hwang and Wonyong Sung, "Character-level language modeling with hierarchical recurrent neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 5720–5724.

[21] Alex Graves and Jürgen Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[22] K-F Lee and H-W Hon, "Speaker-independent phone recognition using hidden markov models," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 11, pp. 1641–1648, 1989.

[23] Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[24] Vijayaditya Peddinti, Yiming Wang, Daniel Povey, and Sanjeev Khudanpur, "Low latency acoustic modeling using temporal convolution and LSTMs," *IEEE Signal Processing Letters*, vol. 25, no. 3, pp. 373–377, 2018.

[25] Francois Chollet, "Xception: Deep learning with depthwise separable convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 1800–1807.

[26] Lukasz Kaiser, Aidan N Gomez, and Francois Chollet, "Depthwise separable convolutions for neural machine translation," *arXiv preprint arXiv:1706.03059*, 2017.

[27] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *IEEE conference on computer vision and pattern recognition (CVPR)*, 2016, pp. 770–778.

[29] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfit-

ting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[30] Aäron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W Senior, and Koray Kavukcuoglu, "Wavenet: A generative model for raw audio.," in *SSW*, 2016, p. 125.

[31] Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al., "Conditional image generation with PixelCNN decoders," in *Advances in Neural Information Processing Systems*, 2016, pp. 4790–4798.

초 록

오늘날, 자동 음성 인식 시스템으로 인공신경망 기반의 알고리즘이 주요하게 활용되고 있다. 그런 가운데, 스마트폰이나 임베디드 장치에서 서버를 거치지 않고 진행되는 온-디바이스 음성 인식 시스템에 대한 수요가 증가하고 있다. 온-디바이스 음성 인식 시스템은 사용자의 음성이 서비스 제공자의 서버로 제공되지 않고, 음성인식이 사용자의 장치에서 독립적으로 이루어진다. 따라서, 프라이버시 침해와 보안에 대한 우려를 상당 부분 해소할 수 있다.

그러나, 인공신경망 기반의 음성 인식 시스템에서 주로 사용되는 LSTM 기반의 회귀신경망(RNN)은 온-디바이스 음성 인식에 효율적이지 않다. LSTM RNN은 시퀀스(sequence) 정보의 병렬화가 어렵다. 이는 LSTM RNN에는 현재의 시간 스텝(step)이 과거의 시간 스텝에 의존하는 되먹임(Feedback) 특성이 존재하기 때문이다. 또, 이 되먹임 정보는 너무 커서 캐시 메모리에 들어갈 수 없다. 따라서, 시퀀스 정보의 매 시간 스텝마다 DRAM에 접근하여 샘플을 불러와야 한다. 이 경우 매 시간 스텝마다 DRAM에 접근하여 전력소모가 증가할 뿐만 아니라, 실행 시간도 증가하게 된다.

우리는 이 논문에서 온-디바이스에 친화적인 인공신경말 모델을 제시한다. 이 모델들을 음향 모델링에 활용하여 LSTM RNN을 대체한다. 게이티드 콘볼루션 네크워크(Gated ConvNet), 대각성분 LSTM(Diagonal LSTM), QRNN(the quasi RNN)이 활용되었다. 이들 모델은 대부분의 연산에서 순서 의존성이 존재 하지 않아 시간 스텝별 병렬화가 가능하다.

이들 모델들은 자동 음성 인식에서 1차원 깊이 콘볼루션(1D depthwise Convolution)이 추가된 후에는 LSTM RNN의 성능을 훨씬 능가하였다. 특히 게이티드 콘볼루션 네트워크의 경우 깊은 구조를 채택하였을 때, 음향 모델 없이 가장 좋은 성능을 보여주었다. 무엇보다도 온-디바이스에 효율적인 인공신경망 모델들은 시퀀스의 시간 스텝별 병렬화를 통해 실제 임베디드 장치에서 LSTM RNN 대비 최소 5배의 실행 속도 증가를 보여주었다.

우리는 여기서 더 나아가, 심플 게이티드 콘볼루션 네트워크(Simple Gated ConvNet)을 제시한다. 심플 게이티드 콘볼루션은 게이티드 콘볼루션의 가장 단순화 된 형태에 기반을 둔 것으로, 파라미터의 수가 혁명적으로 감소한다. 이는 하드웨어 사양의 제한을 받는 온-디바이스 음성인식에 유리한 특성이다. 또한 심플 게이티드 콘볼루션 네트워크는 시간 스텝 별 순서 의존성이 존재하지 않기 때문에 시간 스텝 별 병렬화도 가능하다. 우리는 1차원 깊이 병렬화(1D depthwise convolution)을 여러 방향을 적용하여 성능 향상을 이끌어 내었다.

구체적으로, 우리는 심플 게이티드 콘볼루션 네크워크를 활용해 파라미터 사용량을 3 M 이하로 줄였다. 동일한 파라미터 수가 주어졌을 때 심플 게이티드 콘볼루션 네트워크는 자동 음성 인식에서 LSTM RNN이나 게이티드 콘볼루션 네트워크의 성능을 능가했다. 3 M 아래의 심플 게이티드 콘볼루션 네크워크는 10 M의 LSTM 보다 더 좋은 성능을 보여주기도 하였다. 또한, 시간 스텝 별 병렬화를 통해서 ARM CPU에서 LSTM RNN 대비 10 배의 실행 속도 증가를 얻어냈다.

주요어: 음성인식, 시퀀스 모델링, 회귀신경망(RNN), 임베디드 장치
학번: 2017-27205