Time-Frequency Kernel-Based CNN for Speech Recognition

Tuo Zhao¹, Yunxin Zhao¹, Xin Chen²

¹Department of Computer Science, University of Missouri, Columbia, Missouri 65211, USA
²Pearson Knowledge Technologies, Menlo Park, CA 94025, USA

tz579@mail.missouri.edu, zhaoy@missouri.edu, xin.chen@pearson.com

Abstract

We propose a novel approach to generate time-frequency kernel based deep convolutional neural networks (CNN) for robust speech recognition. We give different treatments to shifting along the time and the frequency axes of speech feature representations in the 2D convolution, so as to achieve certain invariance in small frequency shifts while expanding time context size for speech input without smearing time positions of phone segments. The 2D-kernel approach allows easy implementation of deep CNNs. We present experimental results on speaker-independent phone recognition tasks of TIMIT and FFMTIMIT, where the latter was acquired using a far-field microphone and the speech data are noisy. Our results demonstrate that the proposed time-frequency kernel-based CNN gives consistent phone error reductions over frequency-domain CNN and DNN for both TIMIT and FFMTIMIT, with more benefits shown for recognizing noisy speech by using clean speech models.

Index Terms: time-frequency kernels, convolutional neural network, robust speech recognition

1. Introduction

In recent years, significant advancements have been made in training deep neural networks (DNNs) for continuous speech recognition. The superior performance of DNN over Gaussian mixture models (GMMs) for computing the state observation scores of hidden Markov models (HMM) has been attributed partially to its better handling of correlations in speech feature inputs. Good results are reported by processing relatively long time spans of speech input based on enlarged input context windows [1] and/or reduced frame rates [2].

These positive results are also supported by earlier findings from psychoacoustics and psycholinguistics highlighting the crucial role of syllable-length intervals (100-250 ms) in speech perception [3]. Further recognition performance improvements have been made with convolutional neural networks (CNNs). In CNN, most efforts are focused on convolving speech filter bank outputs along the frequency dimension to create a certain degree of invariance to speaker-induced frequency shifts [4,5]. When convolution and pooling was performed in the time dimension as in the frequency dimension, however, negligible benefits were reported [6,7], and the same was true when combining such time convolution with frequency convolution [7]. Recently, a successful effort [8] was reported on performing convolution along the time axis without incurring pooling induced position ambiguity as in [9], which was further improved by combining time- and frequency-domain convolution through a hierarchical subnetwork architecture [10].

On the other hand, research shows that human auditory system processes independent spectral-temporal features first, before merging them at some higher processing level [11]. Along this line, a recognizer that combines the phone- and syllable-scale information has been demonstrated to perform significantly better than the corresponding phone-based baseline in a small vocabulary recognition task on both clean and reverberant conditions [12]. Another approach that combines information in multiple spectral resolutions by supplementing multi-level sub-band features has also shown advantage over the single resolution baseline on a phone recognition task in broadband noise condition [13].

In the current work, we propose to use a 2-D time-frequency kernel to combine time- and frequency-domain convolution for CNN similar to [7]; but unlike [7], our goal of time convolution is not for achieving invariance to time shift but rather for effectively utilizing the time context for state posterior score computation as in [10]. To do so, in the time dimension we allow sub-sampling during convolution but we do not perform pooling. We have implemented the proposed 2D convolution for CNN based on the in-development Computational Network Toolkit (CNTK) released by Microsoft [14]. This approach allows us to train deep time-frequency CNNs which might not be easily implemented by the hierarchical subnetwork structure of [10]. Utilizing the flexibility offered by the 2D kernel, we further investigate the potential of integrating information over different time spans by convolving in time-domain with different context window sizes, kernel sizes and sub-sampling rates, and the potential of integrating information over different spectral resolutions by convolving in frequency-domain with acoustic features that are derived from different numbers of mel-frequency channels.

We evaluate the effectiveness of the proposed method on speaker-independent phone recognition tasks of TIMIT and FFMTIMIT. Unlike TIMIT data which were recorded by a close-talking microphone and the speech are clean, the FFMTIMIT data were recorded by a far-field microphone and the speech data have significant low-frequency noise due to the HVAC system and mechanical vibration in the recording booth floor. The FFMTIMIT has 17 fewer speakers than TIMIT, otherwise the two datasets have identical speakers and speech materials [15].

The rest of the paper is organized as the following: In Section 2, we describe the procedure and implementation of the proposed approach. In Section 3, we give details of experimental setups. In Section 4, we provide experimental results and findings. We draw a conclusion in Section 5 and give outlook of possible extensions in the future.

2. Time-frequency kernel-based CNN

As discussed in Section 1, the proposed approach differs from the 2D CNN approach in image processing [16] and that of [7] due to our different treatments on shifting along the time and the frequency dimensions of speech feature

[...]

1. Introduction

In recent years, significant advancements have been made in training deep neural networks (DNNs) for continuous speech recognition. The superior performance of DNN over Gaussian mixture models (GMMs) for computing the state observation scores of hidden Markov models (HMM) has been attributed partially to its better handling of correlations in speech feature inputs. Good results are reported by processing relatively long time spans of speech input based on enlarged input context windows [1] and/or reduced frame rates [2].

These positive results are also supported by earlier findings from psychoacoustics and psycholinguistics highlighting the crucial role of syllable-length intervals (100-250 ms) in speech perception [3]. Further recognition performance improvements have been made with convolutional neural networks (CNNs). In CNN, most efforts are focused on convolving speech filter bank outputs along the frequency dimension to create a certain degree of invariance to speaker-induced frequency shifts [4,5]. When convolution and pooling was performed in the time dimension as in the frequency dimension, however, negligible benefits were reported [6,7], and the same was true when combining such time convolution with frequency convolution [7]. Recently, a successful effort [8] was reported on performing convolution along the time axis without incurring pooling induced position ambiguity as in [9], which was further improved by combining time- and frequency-domain convolution through a hierarchical subnetwork architecture [10].

On the other hand, research shows that human auditory system processes independent spectral-temporal features first, before merging them at some higher processing level [11]. Along this line, a recognizer that combines the phone- and syllable-scale information has been demonstrated to perform significantly better than the corresponding phone-based baseline in a small vocabulary recognition task on both clean and reverberant conditions [12]. Another approach that combines information in multiple spectral resolutions by supplementing multi-level sub-band features has also shown advantage over the single resolution baseline on a phone recognition task in broadband noise condition [13].

In the current work, we propose to use a 2-D time-frequency kernel to combine time- and frequency-domain convolution for CNN similar to [7]; but unlike [7], our goal of time convolution is not for achieving invariance to time shift but rather for effectively utilizing the time context for state posterior score computation as in [10]. To do so, in the time dimension we allow sub-sampling during convolution but we do not perform pooling. We have implemented the proposed 2D convolution for CNN based on the in-development Computational Network Toolkit (CNTK) released by Microsoft [14]. This approach allows us to train deep time-frequency CNNs which might not be easily implemented by the hierarchical subnetwork structure of [10]. Utilizing the flexibility offered by the 2D kernel, we further investigate the potential of integrating information over different time spans by convolving in time-domain with different context window sizes, kernel sizes and sub-sampling rates, and the potential of integrating information over different spectral resolutions by convolving in frequency-domain with acoustic features that are derived from different numbers of mel-frequency channels.

We evaluate the effectiveness of the proposed method on speaker-independent phone recognition tasks of TIMIT and FFMTIMIT. Unlike TIMIT data which were recorded by a close-talking microphone and the speech are clean, the FFMTIMIT data were recorded by a far-field microphone and the speech data have significant low-frequency noise due to the HVAC system and mechanical vibration in the recording booth floor. The FFMTIMIT has 17 fewer speakers than TIMIT, otherwise the two datasets have identical speakers and speech materials [15].

The rest of the paper is organized as the following: In Section 2, we describe the procedure and implementation of the proposed approach. In Section 3, we give details of experimental setups. In Section 4, we provide experimental results and findings. We draw a conclusion in Section 5 and give outlook of possible extensions in the future.

2. Time-frequency kernel-based CNN

As discussed in Section 1, the proposed approach differs from the 2D CNN approach in image processing [16] and that of [7] due to our different treatments on shifting along the time and the frequency dimensions of speech feature
representation. The convolution and pooling strategy of the proposed time-frequency kernel-based CNN is shown in Figure 1, where for simplicity, the operations are shown only on one feature map in each layer, with the understanding that multiple time-frequency kernels are actually employed to generate multiple feature maps in each layer. Specifically, in the i-th convolution-pooling layer, the convolution kernel shifts along the time axis by \( T_i (T_i > 1) \) frames per step, but it shifts along the frequency axis by only one band per step; the pooling window shifts by \( S_j \) bands \( (S_j > 1) \) and perform pooling from \( P_j \) bands \( (P_j > 1) \) in frequency, but it shifts by only one frame and does not perform pooling in time. Accordingly, in the time-domain sub-sampling is performed with kernel shifts during the convolution process, while in the frequency-domain sub-sampling is performed with window shifts during the pooling process.

![Figure 1: Topology of convolution and pooling](image)

In the current work, the simple full weight sharing scheme is used in both convolution directions for ease of implementation, even though limited weight sharing in the frequency dimension [17] has previously shown better results on TIMIT [17,10]. As in the common practice of CNNs, nonlinear activation \( \sigma() \) is applied in between the convolution and pooling operations, and the biases are shared among all bands and frames [7]. Consider the convolution step that maps \( I \) input feature maps, \( F_1, F_2, \ldots, F_N \) to the \( j \)-th intermediate feature map \( Q_j \) with the \( I \) kernel weight matrices \( W_{l, r, \ldots, c} \), \( W_{l, r, \ldots, c} \), connecting \( F_1, F_2, \ldots, F_l \) to \( Q_j \), respectively. Let the size of the 2D convolution kernel be \( K_f \times K_f \) and its shift size be \( T \) along the time axis. Then the mapping operation is performed as

\[
q_{j, m, n} = \sigma \left( \sum_{i=1}^{I} \sum_{k=1}^{L} \sum_{l=1}^{K_f} \sum_{x=1}^{K_f} \sum_{y=1}^{K_f} f_{j, (m-1) \times T + k, (n-1) \times T + l} \cdot w_{i, j, k, l} + a_i \right)
\]

(1)

where \( f_{j, m, n} \) is the \((x,y)\)-th unit of \( F_j \), \( q_{j, m, n} \) is the \((x,y)\)-th unit of \( Q_j \), \( w_{i, j, k, l} \) is the \((x,y)\)-element of \( W_{i} \), and \( a_i \) is the bias added to \( Q_j \). In the subsequent pooling step, max pooling is performed in the frequency dimension on the \( j \)-th feature map \( Q_j \) to produce \( V_j \) as the \( j \)-th output feature map for the convolution-pooling layer, i.e.,

\[
v_{j, m, n} = \max_{p=1}^{P} q_{j, m, n, (p-1) \times S + r}
\]

(2)

where \( v_{j, m, n} \) is the \((x,y)\)-th unit of the \( j \)-th output feature map \( V_j \), \( P \) is the size of the pooling window, and \( S \) is the size of the pooling window shift, both in the frequency dimension.

3. Experimental Setups

3.1. Overview

Experiments were performed on speaker-independent phone recognition tasks of TIMIT and FFMTIMIT, and performances of the proposed method and several comparison cases were evaluated by phone error rate. For each dataset, a randomly selected 10% of training utterances were held out as the development set, and the 192 -utterance core set was used for testing. The standard 61-phone set was used in model training and decoding, which was then mapped into a 39-phone set for measuring phone recognition accuracies, following the common practice [18]. The language model for decoding contained 63 unigrams (including the utterance beginning and ending symbols) and 2282 bi-grams. HTK toolkit [19] was used for extracting front-end speech features, clustering context-dependent training targets and time-aligning training labels, while CNTK toolkit [14] was used for training neural networks and generating frame-wise acoustic scores.

The speech representation of log mel frequency filter-bank features with delta and acceleration were used in training the neural networks, where the features were generated every 10 ms in a window of 30 ms. A set of 1551 (for TIMIT) or 1549 (for FFMTIMIT) tied tri-phone states were used as the training targets of the neural networks. The rectified linear activation was used, and thus the deep neural networks were not pretrained. A single-pass Newbob scheme was used for modifying learning rates through the back-propagation iterations [20]. The initial learning rate was set as 0.8 for each of the 1024-frame mini-batches. Frame-level randomization was performed prior to each learning iteration. A dropout rate of 0.2 was set for each non-output layer.

3.2. Time-frequency kernel-based CNNs

To investigate the effect of incorporating information over different time spans, symmetrical context windows with 15, 21, and 27 frames were used respectively as the neural network input. To investigate the potential of incorporating information over different spectral resolutions, filter-bank features with 40, 60 and 80 mel-frequency channels were used, respectively. The context window size was fixed to 21 frames when varying the number of mel-frequency channels, and the number of mel-frequency channels was fixed to 60 when varying the context window size.

The proposed time-frequency-kernel based CNN consisted of 4 convolution-pooling layers, which were topped by 4 fully-connected hidden layers. No zero-padding was performed before the convolution. In order to limit the amount of trainable parameters to fit the capacity of our current computing platform, the number of hidden nodes in the convolution-pooling layers were changed gradually layer by layer, forming a bottleneck before reaching the fully connected hidden layer. The details are described below.

For each convolution-pooling layer, the number of output feature maps \( M_0 \) was set based on the number of input feature maps \( M_i \) and the kernel sizes in time and frequency dimensions \((K_f) \), where for convenience we also refer the frames in time as bands. Particularly,

\[
M_0 = M_i \times K_f \times K_f \times r
\]

(3)
where \( r (0 < r < 1) \) was a shrinkage factor applied recursively for each convolution-pooling layer so that the number of output nodes of the 4th layer was set to be identical with the number of output nodes of the 5th layer (the 1st fully-connected layer). In this way, the model would have relatively consistent "resolutions" of feature representation among different layers. For simplicity, the fully-connected layers, i.e., the 5th to 8th layers, used the same number of hidden nodes that matched the number of triphone tied-states.

For the 1st convolution-pooling layer, there were 3 input feature maps, containing the instantaneous, delta and double-delta mel-frequency filter coefficients of the speech features. For easier comparisons among different models, the kernel size, kernel shift, pooling window size and pooling window shift in the 1st convolution-pooling layer were set to vary with the context size and the spectral resolution so as to always produce a 48-band structure (3 in time axis by 16 in frequency axis) for each output feature map. To fix ideas, the parameter settings for the convolution-pooling layers (i.e., the 1st through the 4th layers) are summarized below in Tables 1 and 2 for CNNs with different context sizes and spectral resolutions. With these parameter settings, \( r \) was set to 0.494, 0.456 and 0.426 respectively for the 3 models of different input bands in time shown in Table 1, and to 0.506, 0.456 and 0.423 respectively for the 3 models of different input bands in frequency shown in Table 2.

Table 1. Parameter settings for convolution-pooling layers of time-frequency kernel-based CNNs with 3 different context sizes of 15/21/27 frames, where the number of mel-frequency channels was fixed to 60.

<table>
<thead>
<tr>
<th>Layer</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
</tr>
</thead>
<tbody>
<tr>
<td># Input Feature Maps</td>
<td>3</td>
<td>89/115/138</td>
<td>793/938/1063</td>
<td>592/427/454</td>
</tr>
<tr>
<td># Input Bands (Time)</td>
<td>15/21/27</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td># Input Bands (Freq.)</td>
<td>60</td>
<td>16</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Conv. Kernel Size (Time)</td>
<td>7/7/9</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Conv. Kernel Size (Freq.)</td>
<td>12</td>
<td>9</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Conv. Kernel Shift (Time)</td>
<td>5/7/9</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Conv. Kernel Shift (Freq.)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Pool. Window Size (Time)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Pool. Window Size (Freq.)</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Pool. Window Shift (Time)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Pool. Window Shift (Freq.)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2. Parameter settings for convolution-pooling layers of time-frequency kernel-based CNNs with 3 different spectral resolutions of 40/60/80 mel-frequency channels, where the context size was fixed to 21 frames.

<table>
<thead>
<tr>
<th>Layer</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
</tr>
</thead>
<tbody>
<tr>
<td># Input Feature Maps</td>
<td>3</td>
<td>85/115/142</td>
<td>766/938/1083</td>
<td>585/427/458</td>
</tr>
<tr>
<td># Input Bands (Time)</td>
<td>21</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td># Input Bands (Freq.)</td>
<td>40/60/30</td>
<td>16</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Conv. Kernel Size (Time)</td>
<td>7</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Conv. Kernel Size (Freq.)</td>
<td>10/16</td>
<td>9</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Conv. Kernel Shift (Time)</td>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Conv. Kernel Shift (Freq.)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Pool. Window Size (Time)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Pool. Window Size (Freq.)</td>
<td>3/4/5</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Pool. Window Shift (Time)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Pool. Window Shift (Freq.)</td>
<td>2/3/4</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

It should be noticed that in the last two layers of convolution (i.e., the 3rd and the 4th layers), the operations of pooling or sub-sampling were not performed, with significantly reduced number of output feature maps on the topmost layer (the 4th layer) before connecting to the fully connected upper layers. Different from the time-frequency CNN work of [10] where a bottleneck structure was only used in the time dimension, in our approach here the bottleneck structure was maintained for both time and frequency dimensions. This 2-D bottleneck structure allows us to work with reasonable sized models and number of parameters to carry out experiments within the limitations of our current computing resource. Because the approach of [10] would require a lot more computing resource beyond our current capabilities, we do not make direct comparisons with it in our experiments described below.

It should also be noticed that we performed 2 layers of convolution-pooling or convolution-sub-sampling in the beginning two layers of the CNN (the 1st and 2nd layers) to reduce the numbers of bands for the 3rd and 4th layers. This allowed us to maintain a sufficient number of feature maps through the bottleneck (i.e., the output of the 4th layer) while keeping the total amount of trainable parameters limited. In addition, since we used full weight sharing (FWS) for convolution, performing 2 layers of convolution-pooling in the frequency domain is desired as the strategy was previously shown to be superior than both the 1-layer FWS and its limited weight sharing (LWS) counterparts [6].

3.3. Evaluation Tasks

For each model configuration, we trained and evaluated one model on TIMIT and another model on FFMTIMIT. In order to examine the robustness performance under the mismatched condition of training with a near field microphone and testing with a far field microphone, we also evaluated phone error rates on the FFMTIMIT core test set when using the models trained by TIMIT to do recognition. In the case of matched training-test conditions, cepstral mean and variance normalizations were performed by applying the normalization parameters derived from a training set (as implemented in CNTK) to the corresponding training and test sets. In the case of mismatched training-test conditions, per utterance based cepstral mean normalization was performed on the training and test utterances separately. For purpose of comparison, in certain tasks a 1-D (time or frequency) kernel-based CNN as well as a fully-connected DNN were trained along with the time-frequency kernel-based CNNs. Configurations were shared among different kinds of CNNs and DNNs if possible, and the numbers of hidden nodes and/or feature maps were calculated using similar methods. The experimental results are presented below in Section 4.

4. Results and Discussion

4.1. Time-frequency kernel-based CNN

Tables 3 and 4 show the phone error rate (PER) results using different numbers of frames as the input context and different numbers of mel-frequency channels in speech features, respectively. Note that the 2nd column of results in Table 3 and Table 4 correspond to the same models in 2D kernel CNN and DNN. The key observations are summarized as the following:

- In the matched training and testing conditions, the time-frequency kernel-based CNN gave a relative PER reduction of 1.6--3.1% from the frequency kernel-based CNN for each context size, and a relative PER reduction of 5.3--8.5% from the time kernel-based CNN for each spectral resolution.

- In the mismatched training and testing conditions, the time-frequency kernel-based CNN gave a relative PER
CNN with time pooling was modified from the CNN without time pooling by reducing the time-domain kernel shift size from 7 to 1 and performing a time-domain pooling operation with a window size of 5 and a shift size of 5 for the 1st convolution-pooling layer (so as to generate 3 output bands in time axis for that layer). It can be seen from Table 8 that pooling in time in the 1st layer increased PER by up to 2% relative to the case of without pooling in time.

<table>
<thead>
<tr>
<th>Task</th>
<th>Method</th>
<th># Frames in Context</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>(matched)</td>
<td>18.42 18.34 18.46</td>
</tr>
<tr>
<td>B</td>
<td>(matched)</td>
<td>19.21 19.50 19.09</td>
</tr>
<tr>
<td>C</td>
<td>(matched)</td>
<td>20.33 20.49 20.75</td>
</tr>
</tbody>
</table>

Table 5. PERs (%) of time-frequency kernel-based CNN ensembles integrating multiple temporal spans. Number of mel-frequency channels was fixed to 60.

<table>
<thead>
<tr>
<th>Task</th>
<th># Mel-Frequency Channels</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>40 60 80 combined</td>
</tr>
<tr>
<td>B</td>
<td>40 60 80 combined</td>
</tr>
<tr>
<td>C</td>
<td>40 60 80 combined</td>
</tr>
</tbody>
</table>

Table 7. PERs (%) of time-frequency kernel-based CNN ensembles integrating multiple temporal spans and spectral resolutions.

4.2. Multi-scale time-frequency kernel-based CNN

Considering the relatively stable performance of CNNs shown in Table 3 and Table 4, we combined time-frequency kernel-based CNNs that were trained with different spans of input context by averaging the frame-wise log posterior probability scores provided by the different models. The decoding results are shown in Table 6. In a similar way, we also combined time-frequency kernel-based CNNs that were trained on feature sets with different spectral resolutions and the results are shown in Table 7.

The ensemble models integrating multiple temporal spans introduced relative PER reductions of 1.8% (with matched training and testing datasets) or 2.9% (with mismatched training and testing datasets) from the best results of the corresponding single models, while the ensemble models integrating multiple spectral resolutions also introduced 1.7–2.2% relative PER reductions from the best results of the corresponding single models. Combining all the 5 models for each task produced even better results, shown in Table 7.

<table>
<thead>
<tr>
<th>Task</th>
<th>Without Pooling</th>
<th>With Pooling</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>19.36</td>
<td>19.96</td>
</tr>
<tr>
<td>B</td>
<td>20.83</td>
<td>20.97</td>
</tr>
<tr>
<td>C</td>
<td>23.94</td>
<td>24.24</td>
</tr>
</tbody>
</table>

Table 8. PERs (%) of without and with pooling in time-frequency kernel-based CNNs. Number of context frames was fixed to 21. Number of mel-frequency channels was fixed to 60.

5. Conclusion

We have proposed a novel time-frequency kernel-based convolutional neural network method for speech recognition. We have investigated the performance of the proposed method with different input context spans and different frequency resolutions, in comparison with the conventional frequency-domain CNN and DNN, on the phone recognition tasks of TIMIT and FFMTIMIT. In each case, the time-frequency kernel based CNN consistently produced the lowest phone error rate, with the largest positive impact on the mismatched training-test conditions. We further combined the 2D-CNNs with different time spans and frequency resolutions and obtained additional improvements in phone accuracy performance. In future, a thorough investigation will be made on the 2D CNNs with larger amounts of trainable parameters, on larger speech recognition tasks, and in more challenging noisy conditions.

6. Acknowledgement

The authors would like to thank Dr. Dong Yu of Microsoft Research for his kindly answering questions about the CNTK toolkit through CodePlex.com.
References

[15] https://catalog.ldc.upenn.edu/docs/LDC96S32/FFMTIMIT.TXT