Convolutional Neural Networks for Distant Speech Recognition

Pawel Swietojanski, Student Member, IEEE, Arnab Ghoshal, Member, IEEE, and Steve Renals, Fellow, IEEE

Abstract—We investigate convolutional neural networks (CNNs) for large vocabulary distant speech recognition, trained using speech recorded from a single distant microphone (SDM) and multiple distant microphones (MDM). In the MDM case we explore a beamformed signal input representation compared with the direct use of multiple acoustic channels as a parallel input to the CNN. We have explored different weight sharing approaches, and propose a channel-wise convolution with two-way pooling. Our experiments, using the AMI meeting corpus, found that CNNs improve the word error rate (WER) by 6.5% relative compared to conventional deep neural network (DNN) models and 15.7% over a discriminatively trained Gaussian mixture model (GMM) baseline. For cross-channel CNN training, the WER improves by 3.5% relative over the comparable DNN structure. Compared with the best beamformed GMM system, cross-channel convolution reduces the WER by 9.7% relative, and matches the accuracy of a beamformed DNN.

Index Terms—distant speech recognition, deep neural networks, convolutional neural networks, meetings, AMI corpus

I. INTRODUCTION

DISTANT speech recognition (DSR) [1] is a challenging task owing to reverberation and competing acoustic sources. DSR systems may be configured to record audio data using a single distant microphone (SDM), or multiple distant microphones (MDM). Current DSR systems for conversational speech are considerably less accurate than their close-talking equivalents, and usually require complex multi-pass decoding schemes and sophisticated front-end processing techniques [2]–[4]. SDM systems usually result in significantly higher word error rates (WERs) compared to MDM systems.

Deep neural network (DNN) acoustic models [5] have extended the state-of-the-art in acoustic modelling for automatic speech recognition (ASR), using both hybrid configurations [6]–[11] in which the neural network is used to estimate hidden Markov model (HMM) output probabilities and posteriorgram configurations [12]–[15] in which the neural network provides discriminative features for an HMM. It has also been demonstrated that hybrid neural network systems can significantly increase the accuracy of conversational DSR [16]. An advantage of the hybrid approach is the ability to use frequency domain feature vectors, which provide a small but consistent improvement over cepstral domain features [17].

Convolutional neural networks (CNNs) [18], which restrict the network architecture using local connectivity and weight sharing, have been applied successfully to document recognition [19]. When the weight sharing is confined to the time dimension, the network is called a time-delay neural network and has been applied to speech recognition [20]–[22]. CNNs have been used for speech detection [23], directly modelling the raw speech signal [24], and for acoustic modelling in speech recognition in which convolution and pooling are performed in the frequency domain [25]–[27]. Compared to DNN-based acoustic models, CNNs have been found to reduce the WER on broadcast news transcription by an average of 10% relative [26], [27].

Here we investigate weight sharing and pooling techniques for CNNs in the context of multi-channel DSR, in particular cross-channel pooling across hidden representations that correspond to multiple microphones. We evaluate these approaches through experiments on the AMI meeting corpus [28].

II. CNN ACOUSTIC MODELS

Context-dependent DNN–HMM systems use DNNs to classify the input audios into classes corresponding to the HMM tied states. After training, the output of the DNN provides an estimate of the posterior probability \( P(s | \alpha_t) \) of each HMM state \( s \) given the acoustic observations \( \alpha_t \) at time \( t \), which may be used to obtain the (scaled) log-likelihood of state \( s \) given observation \( \alpha_t \): \( \log P(s | \alpha_t) - \log P(s) \) [6], [8], [29], where \( P(s) \) is the prior probability of state \( s \) calculated from the training data.

A. Convolutional and pooling layers

The structure of feed-forward neural networks may be enriched through the use of convolutional layers [19] which allows local feature receptors to be learned and reused across the whole input space. A max-pooling operator [30] can be applied to downsample the convolutional output bands, thus reducing variability in the hidden activations.

Consider a neural network in which the acoustic feature vector \( \mathbf{V} \) consists of filter-bank outputs within an acoustic context window. \( \mathbf{V} = [v_1, v_2, \ldots, v_k, \ldots, v_B] \in \mathbb{R}^{B \times 2} \) is divided into \( B \) frequency bands with the \( b \)-th band \( v_b \in \mathbb{R}^2 \) comprising all the \( Z \) relevant coefficients (statics, \( \Delta \), \( \Delta^2 \), \ldots) across all frames of the context window. The \( k \)-th hidden convolution band \( \mathbf{h}_k = [h_{1,k}, \ldots, h_{j,k}, \ldots, h_{J,k}] \in \mathbb{R}^J \) is then composed of a linear convolution of \( J \) weight vectors (filters) with \( F \) consecutive input bands \( \mathbf{u}_k = [v_{k+1}, \ldots, v_{(k-1)L+F}] \in \mathbb{R}^{F \times 2} \), where \( L \in \{1, \ldots, F\} \) is the filter shift. Fig 1 gives
an example of such a convolution with a filter size and shift of \( F = 3 \) and \( L = 1 \) respectively. This may be extended to \( S \) acoustic channels \( V^1,...,V^S \) (each corresponding to a microphone), in which the hidden activation \( h_{j,k} \) can be computed by summing over the channels:

\[
    h_{j,k} = \sigma \left( b_{j,k} + \sum_{s=1}^{S} w_{j,k}^{s} \ast u_{s}^{k} \right),
\]

where \( \sigma(\cdot) \) is a sigmoid nonlinearity, \( \ast \) denotes linear valid convolution\(^1\), \( w_{j,k}^{s} \in \mathbb{R}^{F \times Z} \) is a weight vector of the \( j \)-th filter acting on the local input \( u_{s}^{k} \) of the \( s \)-th input channel, and \( b_{j,k} \) is an additive bias for the \( j \)-th filter and \( k \)-th convolutional band. Since the channels contain similar information (acoustic features shifted in time) we conjecture that the filter weights may be shared across different channels. Nevertheless, the formulation and implementation allow for different filter weights in each channel. Similarly, it is possible for each convolutional band to have a separate learnable bias parameter instead of the biases only being shared across bands [25, 26].

The complete set of convolutional layer activations \( h \in [h_1,\ldots,h_K] \in \mathbb{R}^{K \times J} \) is composed of \( K = (B - F)/L + 1 \) convolutional bands obtained by applying the (shared) set of \( J \) filters across the whole (multi-channel) input space \( V \) (as depicted in Fig 1). In this work the weights are tied across the input space (i.e. each \( u_{s}^{k} \) is convolved with the same filters); alternatively the weights may be partially shared, tying only those weights spanning neighbouring frequency bands [25]. Although limited weight sharing was reported to bring improvements for phone classification [25] and small LVSR tasks [32], a recent study on larger tasks [27] suggests that full weight sharing with a sufficient number of filters can work equally well, while being easier to implement.

A convolutional layer is usually followed by a pooling layer which downsamples the activations \( h \). The max-pooling operator [30] passes forward the maximum value within a group of \( R \) activations. The \( m \)-th max-pooled band is composed of \( J \) related filters \( p_{m} = [p_{1,m},\ldots,p_{J,m}] \in \mathbb{R}^{J} \):

\[
    p_{j,m} = \max_{r=1}^{R} \left( h_{j,(m-1)N+r} \right), \tag{2}
\]

\(^1\)The convolution of two vectors of size \( X \) and \( Y \) may result either in the vector of size \( X+Y-1 \) for a full convolution with zero-padding of non-overlapping regions, or the vector of size \( X-Y+1 \) for a valid convolution where only the points which overlap completely are considered [31].

where \( N \in \{1,\ldots,R\} \) is a pooling shift allowing for overlap between pooling regions when \( N < R \) (in Fig 1, \( R = N = 3 \)). The pooling layer decreases the output dimensionality from \( K \) convolutional bands to \( M = (K-R)/N+1 \) pooled bands and the resulting layer is \( p = [p_{1},...,p_{M}] \in \mathbb{R}^{M \times J} \).

B. Channel-wise convolution

Multi-channel convolution (1) builds feature maps similarly to the LeNet-5 model [19] where each convolutional band is composed of filter activations spanning all input channels. We also constructed feature maps using max-pooling across channels, in which the activations \( h_{j,k}^{s} \) are generated in channel-wise fashion and then max-pooled (4) to form a single cross-channel convolutional band \( c_{k} = [c_{1,k},\ldots,c_{J,k}] \in \mathbb{R}^{J} \) (Fig 1 (right)):

\[
    h_{j,k}^{s} = \sigma \left( b_{j,k} + w_{j,k}^{s} \ast u_{s}^{k} \right), \tag{3}
\]

\[
    c_{j,k} = \max_{s=1}^{S} \left( h_{j,k}^{s} \right). \tag{4}
\]

Note that here the filter weights \( w_{j,k}^{s} \) need to be tied across the channels such that the cross-channel max-pooling (4) operates on activations for the same feature receptor. The resulting cross-channel activations \( c = [c_{1},\ldots,c_{K}] \in \mathbb{R}^{K \times J} \) can be further max pooled along frequency using (2). Channel-wise convolution may also be viewed as a special case of 2-dimensional convolution, where the effective pooling region is determined in frequency but varies in time depending on the actual time delays between the microphones.

C. Fully-connected layers

The complete acoustic model is composed of one or more CNN layers, followed by a number of fully-connected layers, with a softmax output layer. With a single CNN layer, the computation performed by the network is as follows:

\[
    h^{l} = \sigma(W^{l}h^{l-1} + b^{l}), \quad \text{for } 2 \leq l < L \tag{5}
\]

\[
    a^{L} = W^{L}h^{L-1} + b^{L},
\]

\[
    P(s|o_{l}) = \frac{\exp(a^{L}(s))}{\sum_{s'} \exp(a^{L}(s'))}, \tag{6}
\]

where \( h^{l} \) is the input to the \((l+1)\)-th layer, with \( h^{l} = p \); \( W^{l} \) is the matrix of connection weights and \( b^{l} \) is the additive bias vector for the \( l \)-th layer; \( \sigma(\cdot) \) is a sigmoid nonlinearity that operates element-wise on its input vector; \( a^{L} \) is the activation at the output layer.

III. EXPERIMENTS

We have performed experiments using the AMI meeting corpus [28] (http://corpus.amiproject.org/) using an identical training and test configuration to [16]. The AMI corpus comprises around 100 hours of meetings recorded in instrumented meeting rooms at three sites in the UK, the Netherlands, and Switzerland. Each meeting usually has four participants and the language is English, albeit with a large proportion of non-native speakers. Multiple microphones were used, including individual headset microphones (IHM), lapel microphones, and one or more microphone arrays. Every recording used a
primary 8-microphone uniform circular array (10 cm radius), as well as a secondary array whose geometry varied between sites. In this work we use the primary array for our MDM experiments, and the first microphone of the primary array for our SDM experiments. Our systems are trained and evaluated using the split recommended in the corpus release: an 80 hour training set, and development and test sets each of 9 hours. We use the segmentation provided with the AMI corpus annotations (v1.6). For training purposes we consider all segments (including those with overlapped speech), and the WERs of the speech recognition outputs are scored by the asclite tool [33] following the NIST RT recommendations for scoring simultaneous speech (http://nist.gov/speech/tests/rt/2009). WERs for non-overlapped segments only may also be produced by asclite, using the -overlap-limit 1 option. Here, we report results using the development set only: both development and test sets are relatively large, and we previously found that the best parameters selected for the development set were also optimal for the evaluation set [16].

All CNN/DNN models, unless explicitly stated otherwise, were trained on 40-dimensional log Mel filterbank (FBANK) features appended with the first and the second time derivatives [17]. Our distant microphone systems within this work remain unadapted to both speakers and sessions. Ascribing speakers to segments without diarisation is unrealistic while a small mismatch between training and evaluation acoustic environments makes feature-space maximum likelihood linear regression only moderately effective (less than 1% absolute reduction in WER) for session adaptation. Our experiments were performed using the Kaldi speech recognition toolkit [34], and the pylearn2 machine learning library [35].

Our experiments used a 50,000 word pronunciation dictionary [4]. An in-domain trigram language model (LM) was estimated using the AMI training transcripts (801k words). This was interpolated with two further trigram LMs – one estimated from the Switchboard training transcripts (3M words), and the other from the Fisher English transcripts (22M words) [36]. The LMs are estimated using modified Kneser-Ney smoothing [37]. The LM interpolation weights were as follows: AMI transcripts (0.73); Switchboard (0.05); Fisher (0.22). The final interpolated LM had 1.6M trigrams and 1.5M bigrams, resulting in a perplexity of 78 on the development set.

### IV. Results

We have tested the CNNs with both SDM and MDM inputs. In each case we compare the CNN to two baseline systems: (1) a Gaussian mixture model (GMM) system, discriminatively trained using boosted maximum mutual information (BMMI) [38], with mel-frequency cepstral coefficient (MFCC) features post-processed with linear discriminative analysis (LDA) and decorrelated using a semi-tied covariance (STC) transform [39]; and (2) a deep neural network (DNN) with 6 hidden layers, with 2048 units in each layer [16] trained using the same FBANK features as used for for the CNNs. We used restricted Boltzmann machine (RBM) pretraining [40] for the baseline DNN systems, but not for the CNN systems. The CNN results are reported for the networks composed with a single CNN layer followed by 5 fully-connected layers. The CNN hyperparameters are as follows: number of filters \( J = 128 \), filter size \( F = 9 \), and filter shift \( L = 1 \).

#### A. Single Distant Microphone

We applied two CNN approaches to the SDM case, in which acoustic data from a single channel only is used. In the first approach the same bias terms were used for each band [26] (section II-A), and the results of the single channel CNN can be found in Table I. The first two rows are the SDM baselines (reported in [16])\(^2\). The following three lines are results for the CNN using max-pool sizes (PS) of \( R = N = 1, 2, 3 \). By using CNNs we were able to obtain 3.4% relative reduction in WER with respect to the best DNN model and a 19% relative reduction in WER compared with a discriminatively trained GMM-HMM. Note, the total number of parameters of the CNN models vary here as \( R = N \) while \( J \) is kept constant across the experiments. However, the best performing model had neither the highest nor the lowest number of parameters, which suggests it is due to the optimal pooling setting.

#### B. Multiple Distant Microphones

For the MDM case we compared a delay-sum beamformer with the direct use of multiple microphone channels as input to the network. For beamforming experiments, we follow noise cancellation using a Wiener filter with a delay-sum beamforming on 8 uniformly-spaced array channels using BeamformIt [41]. The results are summarised in Table II. The first block of Table II presents the results for the case in which the models were trained on a beamformed signal from 8 microphones. The first two rows show the WER for the baseline GMM and DNN acoustic models as reported in [16]. The following three rows contain the comparable CNN structures with different pooling sizes (PS) \( R = N = 1, 2, 3 \). The best model (pool size \( R = 1 \), equivalent to no max-pooling) scored 46.3% WER which is 6.4% relative WER better than the best DNN network and a relative improvement in WER of 16% compared with a discriminatively trained GMM-HMM system.

The second part of Table II shows WERs for the models directly utilising multi-channel features. The first row is a baseline DNN variant trained on 4 concatenated channels [16]. Then we present the CNN models with MDM input convolution performed as in equation (1) and pooling size of 2, which was optimal for the SDM experiments. This

<table>
<thead>
<tr>
<th>System</th>
<th>with overlap</th>
<th>no overlap</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMMI GMM-HMM (LDA+STC)</td>
<td>63.2</td>
<td>55.8</td>
</tr>
<tr>
<td>DNN +RBM (FBANK)</td>
<td>53.1</td>
<td>42.4</td>
</tr>
<tr>
<td>CNN (( R = 3 ))</td>
<td>53.3</td>
<td>42.8</td>
</tr>
<tr>
<td>CNN (( R = 2 ))</td>
<td>51.3</td>
<td>40.4</td>
</tr>
<tr>
<td>CNN (( R = 1 ))</td>
<td>52.5</td>
<td>40.9</td>
</tr>
</tbody>
</table>

\( ^2 \)DNN baseline WERs are lower than [16] due to the initial values chosen for the hyper-parameters.
are learned for each channel. Sharing the filter weights across
the different channels even when cross-channel pooling is
not used. Each channel contains the same information, albeit
shifted in time, hence using the same feature detectors for each
channel is a prudent constraint to learning. The first two rows
of Table IV show the results when a separate set of filters
for each channel is tied. The results are given in Table IV
when a separate set of filters are tied across channels. The
second block of Table IV shows the effect of training
the separate set of filters for each channel.

When using multiple distant microphones directly as input
to a CNN, we posit that the same filters should be used across
the different channels. The results presented here suggest that
channel-wise pooling works better for more input
channels: conventional convolution on 4 channels achieves
50.4% WER, compared to 50.0% for the 2-channel case. These
results indicate that picking the best information (selecting
the feature detectors with maximum activations) within the
channels is crucial when doing model-based combination of
multiple microphones.

C. Individual Headset Microphones

We observe similar relative WER improvements between
DNN and CNN for close talking speech experiments (Table
III) as were observed for the DSR experiments (Tables I
and II). The CNN achieves 3.6% WER reduction relative to
the DNN model. Both DNN and CNN systems outperform
a BMMI-GMM system trained in a speaker adaptive (SAT)
fashion by 9.4% and 12.9% relative WER respectively. We
did not see any improvements by increasing pooling size. [26]
has previously suggested that pooling may be task dependent.

D. Different weight-sharing techniques

When using multiple distant microphones directly as input
to a CNN, we posit that the same filters should be used across
the different channels even when cross-channel pooling is
not used. Each channel contains the same information, albeit
shifted in time, hence using the same feature detectors for each
channel is a prudent constraint to learning. The first two rows
of Table IV show the results when a separate set of filters
are learned for each channel. Sharing the filter weights across
channels improves the WER by 0.7% absolute (comparing
with the 2 channel CNN, Table II).

The second block of Table IV shows the effect of training
a separate bias parameter for each of the \( K \) convolutional
bands for the SDM system of Table I. These results are
generated for non-overlapping pools of size 3 and 2. If
the pooling size is too large, we observe that the WER increases.
This increase in WER is mitigated by using a band-specific
bias. We hypothesise that, under noisy conditions, the max-
pooling operator, which may be interpreted as a local hard-
decision heuristic, selects non-optimal band activations, while
the not-tied bias can actually “boost” the meaningful frequency
regions (on average). A band-specific bias does not lead to
further improvements: e.g., when \( R = 2 \), the overlapped
speech CNN with tied biases had a WER of 51.3% compared
to 51.9% for the not-tied version.

V. Discussion

We have investigated using CNNs for DSR with single
and multiple microphones. A CNN trained on a single distant
microphone is found to produce a WER approaching these of
a DNN trained using beamforming across 8 microphones. In
experiments with multiple microphones, we compared CNNs
trained on the output of a delay-sum beamformer with those
trained directly on the outputs of multiple microphones. In
the latter configuration, channel-wise convolution followed by
a cross-channel max-pooling was found to perform better than
multi-channel convolution.

A beamformer uses time-delays between microphone pairs
whose computation requires knowledge of the microphone
array geometry, while these convolutional approaches need no
such knowledge. CNNs are able to compensate better for the
confounding factors in distant speech than DNNs. However,
the compensation learned by CNNs is complementary to
that provided by a beamformer. In fact, when using CNNs
with cross-channel pooling, similar WERs were obtained by
changing the order of the channels at test time from the order
in which they were presented at training time, suggesting that
the model is able to pick the most informative channel.

Early work on CNNs for ASR focussed on learning shift-
invariance in time [20], [42], while more recent work [25],
[26] have indicated that shift-invariance in frequency is more
important for ASR. The results presented here suggest that
recognition of distant multichannel speech is a scenario where
shift-invariance in time between channels is also important,
thus benefitting from pooling in both time and frequency.