SPEAKER LOCATION AND MICROPHONE SPACING INVARIANT ACOUSTIC MODELING FROM RAW MULTICHANNEL WAVEFORMS

Tara N. Sainath, Ron J. Weiss, Kevin W. Wilson, Arun Narayanan, Michiel Bacchiani and Andrew Senior

Google, Inc., New York, NY, USA
{tsainath, ronw, kwwilson, arunnt, michiel, andrewsenior}@google.com

ABSTRACT

Multichannel ASR systems commonly use separate modules to perform speech enhancement and acoustic modeling. In this paper, we present an algorithm to do multichannel enhancement jointly with the acoustic model, using a raw waveform convolutional LSTM deep neural network (CLDNN). We will show that our proposed method offers $\sim$5% relative improvement in WER over a log-mel CLDNN trained on multiple channels. Analysis shows that the proposed network learns to be robust to varying angles of arrival for the target speaker, and performs as well as a model that is given oracle knowledge of the true location. Finally, we show that training such a network on inputs captured using multiple (linear) array configurations results in a model that is robust to a range of microphone spacings.

1. INTRODUCTION

While state-of-the-art speech recognition systems perform reasonably well in close-talking microphone conditions, performance degrades in conditions when the microphone is a large distance from the user. In such farfield cases, the speech signal is subject to degradation due to reverberation and additive noise. To improve recognition, these systems often use multiple microphones to enhance the speech signal and reduce the impact of reverberation and noise [1, 2].

Multichannel ASR systems often use two separate modules to perform recognition. First microphone array speech enhancement is applied, typically via beamforming. Then this enhanced signal is passed to an acoustic model [3, 4]. One widely used technique is delay-and-sum beamforming [2], in which signals from different microphones are first aligned in time to adjust for the propagation delay from the target speaker to each microphones. The time-aligned signals are then summed to enhance the signal from the target direction and to attenuate noise coming from other directions. Other techniques include Minimum Variance Distortionless Response (MVDR) beamforming [5, 6] and Multichannel Wiener Filtering (MWF) [1].

If the end goal is improved ASR performance, treating enhancement as disjoint from acoustic modeling might not lead to the optimal solution. To address this issue [7] proposed a likelihood-maximizing beamforming (LIMABEAM) technique which does beamforming jointly with acoustic modeling. This technique was shown to outperform conventional beamforming techniques such as delay-and-sum. Like most enhancement techniques, this is a model-based approach and requires an iterative acoustic model and/or enhancement model parameter optimization. Current acoustic models are generally neural network based where optimization is performed using a gradient learning algorithm. Combining model-based enhancement with an acoustic model that uses gradient learning can lead to considerable complexity (e.g. [8]).

In this paper, we extend the motivation of doing beamforming jointly with acoustic modeling [7], but do this within the context of a deep neural network (DNN) framework following [9]. DNNs are attractive because they have been shown to do feature extraction jointly with classification [10]. Since beamforming requires the fine time structure of the signal at different microphones, we model the raw time-domain waveform directly. Furthermore, we show experimentally that optimizing both the enhancement and acoustic model jointly in this framework is effective.

In [9] joint beamforming and acoustic model optimization was explored for a task exhibiting limited variation in target speaker location. Though the network was able to learn spatial filters and improve performance over a similar single channel waveform model, its performance remained slightly worse than a simple delay-and-sum beamforming frontend applied to a conventional log-mel acoustic model. Recently, we have shown that acoustic models trained directly on the single channel raw time-domain waveform [11] using a convolutional, long short-term memory, deep neural network (CLDNN) [12] can obtain identical accuracy to a similar CLDNN trained on log-mel features. In this paper, we extend [11] and explore multichannel raw waveform processing with CLDNNs, thus combining multichannel processing and acoustic modeling. We specifically look at a task with lower signal-to-noise ratio (SNR), longer reverberation times, and a wider variation in target speaker location across utterances than [9].

We conduct experiments on a 2,000 hour, multi-condition trained (MTR) English Voice Search task, with an average SNR of 12 dB and reverberation time (T60) of 600 ms. A main summary of our experiments and observations is as follows. First, we train a single channel raw waveform CLDNN and observe that it improves performance over a log-mel CLDNN on this data set. Next, we evaluate several multichannel variations of the raw waveform CLDNN. We compare a multichannel raw waveform CLDNN to (a) delay-and-sum (D+S) beamforming using oracle knowledge of the true time delay of arrival (TDOA) of the target passed into a single-channel CLDNN and (b) aligning the signals in each channel using the true speech TDOA and then passing the aligned signals into a multichannel raw waveform CLDNN (which we refer to as time-aligned multichannel or TAM). When trained and evaluated on matched microphone array configurations, we find that the raw waveform CLDNN learns filters which are steered in different directions similar to [9], and outperforms D+S and multichannel log-mel models [13], while matching the performance of TAM. Finally, to make our system robust to variation in array configuration we train our system on signals from multiple array geometries and find that the multi-geometry trained network is largely invariant to microphone spacing, can adapt to unseen channels, and outperforms traditional beamforming techniques.

The rest of this paper is organized as follows. Section 2 describes the architecture of multichannel raw waveform CLDNNs. The experimental setup is discussed in Section 3, while Section 4 presents results and analysis with the proposed architecture. Finally, Section 5 concludes the paper and discusses future work.

2. MULTICHANNEL RAW WAVEFORM CLDNN

The proposed multichannel raw waveform CLDNN is related to filter-and-sum beamforming, a generalization of delay-and-sum beamforming which filters the signal from each microphone using a finite impulse response (FIR) filter before summing them. Using similar notation to [7], filter-and-sum enhancement can be written as follows:

$$y[t] = \sum_{c=0}^{C-1} \sum_{n=0}^{N-1} h_c[n] x_c[t - n - \tau_c] \tag{1}$$

where $h_c[n]$ is the n^{th} tap of the filter associated with microphone c, $x_c[t]$, is the signal received by microphone c at time t, τ_c is the steering delay induced in the signal received by a microphone to align it to the other array channels, and $y[t]$ is the output signal generated by the processing. C is the number of microphones in the array and N is the length of the FIR filters.

Enhancement algorithms optimizing the model in Equation 1 will generally require an estimate of the steering delay τ_c obtained from a separate localization model and will obtain filter parameters by optimizing an objective, such as MVDR. In contrast, our aim is to have the network jointly estimate steering delays and filter parameters by optimizing acoustic modeling performance. Different steering delays are captured by using a bank of P filters. The output of filter $p \in P$ can be written as follows:

$$y^p[t] = \sum_{c=0}^{C-1} \sum_{n=0}^{N-1} h_c^p[n] x_c[t - n] \tag{2}$$

where the steering delay for each microphone is implicitly absorbed into the filter parameters $h_c^p[n]$.

The first layer in our raw waveform architecture models Equation 2 and performs a multichannel time-convolution with a FIR spatial filterbank $h_c = \{h_c^1, h_c^2, \dots h_c^P\}$ where $h_c \in \Re^{N \times P}$ for $c \in 1, \cdots, C$. Each filter h_c^p is convolved with the input for a specific channel x_c, and the output for each filter $p \in P$ is summed across all channels $c \in C$. The operation within each filter p can be interpreted as filter-and-sum beamforming, except it does not first shift the signal in each channel by an estimated time delay of arrival. As we will show, the network learns the steering delay and filter parameters implicitly.

However, because the goal of our work is to do multichannel processing jointly with acoustic modeling, this means that we want to produce an output that is invariant to perceptually and semantically identical sounds appearing at different time shifts. We have shown in [9] and [11] that we can reduce these temporal variations present in raw waveform by pooling the outputs after filtering, in an operation analogous to windowing in the short-time Fourier transform. Specifically, the output of the spatial filterbank is max-pooled across time to give a degree of short term shift invariance, and then passed through a compressive non-linearity.

As shown in [9, 11], when operating on single channel inputs these time-convolution layers implement a standard time-domain filterbank, such as a gammatone filterbank, which is implemented as a bank of time-domain filters followed by rectification and averaging over a small window. Since our set of time-convolutional layers can do this (and as we will show, does in fact do this), we will subsequently refer to the output of this layer as a “time-frequency” representation, and we will assume that the outputs of different convolutional units correspond to different “frequencies.” Therefore, the time-convolution layers do both spatial and spectral filtering.

Our multichannel time convolution layers are shown in the `tConv` block of Figure 1. First, we take a small window of the raw waveform of length M samples for each channel C, denoted as $\{x_1[t], x_2[t], \dots, x_C[t]\}$ for $t \in 1, \cdots, M$. Each channel c is convolved by a filter with N taps, and there are P such filters $h_c = \{h_c^1, h_c^2, \dots, h_c^P\}$. If we assume we stride the convolutional filter by 1 in time across M samples, the output from the convolution in each channel is $y_c[t] \in \Re^{(M-N+1) \times P}$. After summing the outputs $y_c[t]$ across channels c, we max pool the filterbank output in time (thereby discarding short term phase information), over the entire time length of the output signal $M - N + 1$, to produce $y[t] \in \Re^{1 \times P}$. Finally, we apply a rectified non-linearity, followed by a stabilized logarithm compression[1], to produce a frame-level feature vector at time t, i.e., $z[t] \in \Re^{1 \times P}$. We then shift the window around the waveform by 10ms and repeat this time convolution to produce a set of feature frames at 10ms intervals.

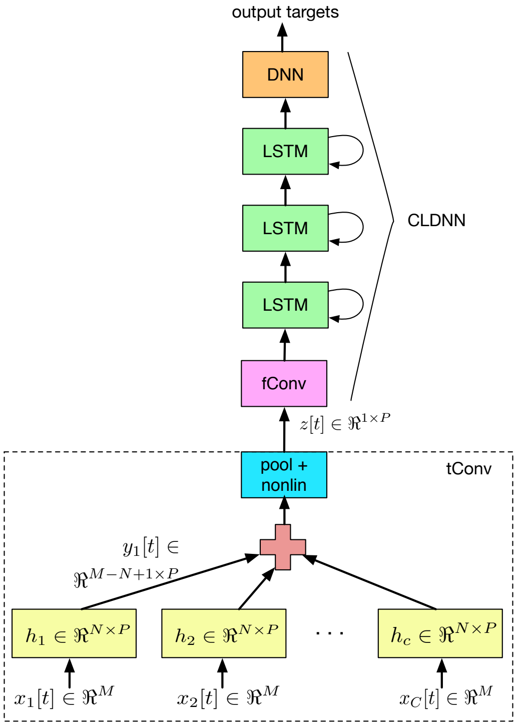
output targets
DNN
LSTM
LSTM
LSTM
CLDNN
fConv
$z[t] \in \Re^{1 \times P}$
tConv
pool + nonlin
$y_1[t] \in \Re^{M-N+1 \times P}$
$h_1 \in \Re^{N \times P}$
$h_2 \in \Re^{N \times P}$
...
$h_c \in \Re^{N \times P}$
$x_1[t] \in \Re^M$
$x_2[t] \in \Re^M$
$x_C[t] \in \Re^M$

Fig. 1: Multichannel raw waveform CLDNN architecture.

2.1. CLDNN

As shown in the CLDNN block of Figure 1, the output out of the time convolutional layer (`tConv`) produces a frame-level feature, denoted as $z[t] \in \Re^{1 \times P}$. This feature is then passed to a CLDNN model [12], which predicts context dependent state output targets.

[1] We use a small additive offset to truncate the output range and avoid numerical instability with very small inputs: $\log(\cdot + 0.01)$.

First, the `fConv` layer applies frequency convolution to $z[t]$, using an architecture similar to that proposed in [14]. Specifically, we use 1 convolutional layer, with 256 filters of size 1×8 in time-frequency. Our pooling strategy is to use non-overlapping max pooling along the frequency axis, with a pooling size of 3 [15].

After frequency convolution, the CNN output is passed to a stack of LSTM layers, which model the signal across long time scales. We use 3 LSTM layers, each consisting of 832 cells, and a 512 unit projection layer [16]. Finally, the output of the LSTM is passed to one fully connected DNN layer with 1,024 hidden units.

The time convolution layer is trained jointly with the rest of the CLDNN. During training, the raw waveform CLDNN is unrolled for 20 time steps for training with truncated backpropagation through time. In addition, the output state label is delayed by 5 frames, as we have observed that information about future frames improves prediction of the current frame [12].

3. EXPERIMENTAL DETAILS

3.1. Data

Our main experiments are conducted on about 2,000 hours of noisy training data consisting of 3 million English utterances. This data set is created by artificially corrupting clean utterances using a room simulator, adding varying degrees of noise and reverberation. The clean utterances are anonymized and hand-transcribed voice search queries, and are representative of Google's voice search traffic. Noise signals, which include music and ambient noise sampled from YouTube and recordings of "daily life" environments, are added to the clean utterances at SNRs ranging from 0 to 20 dB, with an average of about 12 dB. Reverberation is simulated using the image model [17] – room dimensions and microphone array positions are randomly sampled from 100 possible room configurations with T60s ranging from 400 to 900 ms, with an average of about 600 ms. The simulation uses an 8-channel linear microphone array, with inter-microphone spacing of 2 cm. Both noise and target speaker locations change between utterances; the distance between the sound source and the microphone array is chosen between 1 to 4 meters.

Our main evaluation set consists of a separate set of about 30,000 utterances (over 20 hours). The simulated set is created similarly to the training set under similar SNR and reverberation settings. Care was taken to ensure that the room configurations, SNR values, T60 times, and target speaker and noise positions in the evaluation set are not identical to those in the training set. The microphone array geometry between the training and simulated test sets is identical. Most of the results we report will be on this test set.

A second "Rerecorded" test set was obtained by first playing the evaluation set and the noises individually over speakers in a living room, using an 8-channel linear microphone array with microphones placed 4 cm apart. The target speaker was placed in broadside and off-broadside positions, with directions of arrival (DOAs) of 90 and 135 degrees, respectively. Noise sources were placed at 30, 75, 105, and 150 degrees. A final noise set was obtained by recording noise in a cafe during lunch-time using the same microphone array, and is used to evaluate performance of the system in diffuse noise conditions. The rerecorded utterances and noise are mixed at SNRs ranging 0 to 20 dB using the same distribution as used to generate the simulated evaluation set.

3.2. Acoustic model details

The CLDNN architecture and training setup follow a similar recipe to [11, 12]. The baseline log-mel features are computed with a 25ms window and a 10ms hop. The raw waveform features are computed with an identical filter size of 25ms, or $N = 400$ at a sampling rate of 16kHz. The input window size is 35ms ($M = 560$) giving a 10ms fully overlapping pooling window. Both log-mel and raw waveform features are computed every 10ms. We explore varying the number of time-convolution filters P, and the number of log-mel outputs.

Single channel models are trained using signals from channel 1, $C = 2$ channel models use channels 1 and 8 (14 cm spacing), $C = 4$ channel models use channels 1, 3, 6, and 8 (14 cm array span, with adjacent microphone spacing of 4cm-6cm-4cm). Unless otherwise indicated, all neural networks are trained with the cross-entropy (CE) criterion, using asynchronous stochastic gradient descent (ASGD) optimization [18]. The sequence-training experiments in this paper also use distributed ASGD [19]. All networks have 13,522 context-dependent (CD) output targets. The weights for all CNN and DNN layers are initialized using the Glorot-Bengio strategy [20], while those of all LSTM layers are randomly initialized using a uniform distribution between -0.02 and 0.02. We use an exponentially decaying learning rate, initialized to 0.004 and decaying by 0.1 over 15 billion frames.

4. RESULTS

4.1. Single channel

Our first set of experiments compares raw waveform and log-mel CLDNN for a single channel. Table 1 shows WER results for both features as a function of the number of filters P. Notice that the raw waveform models consistently outperform the corresponding log-mel models. In addition, for larger numbers of filters, the raw waveform shows even larger improvements over log-mel than with the standard 40 filters.

# of Filters (P)	log-mel	raw waveform
40	25.2	24.7
84	25.0	23.7
128	24.4	23.5

Table 1: WER for single channel models.

To analyze this further, Figure 2 shows the WER for the above systems, as a function of SNR, reverberation time and target to mic distance. For the same number of filters, the raw waveform shows small but consistent improvement over log-mel at all SNRs, and increasing benefits over log-mel as the reverb time and the target to microphone distance increases. In [11] we found that log-mel and raw waveform CLDNN performance matched for a single channel. However, that data set had a much higher average SNR of about 20dB, and had less reverberation. The figure shows the benefits of the raw waveform over log-mel are apparent under more adverse conditions.

Analyzing the center frequencies (bin index containing the peak magnitude response) of the learned and log-mel filters in Figure 3, we see a similar trend to [9, 11] where the raw waveform model puts more filters with low center frequency compared to the mel scale. It appears that giving more resolution in lower frequencies is beneficial, particularly in noisy and reverberant conditions.

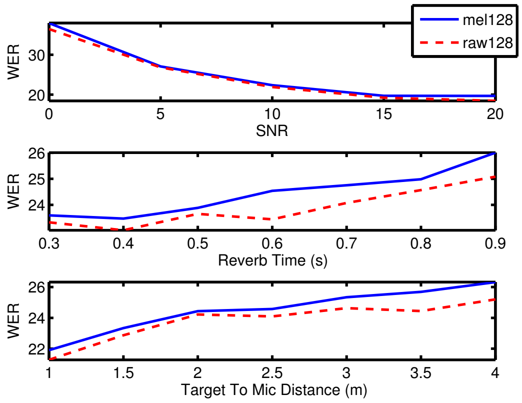

Fig. 2: WER breakdown.

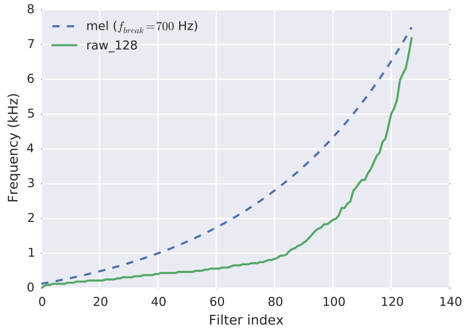

Fig. 3: Filterbank Center Frequencies

4.2. Multichannel

4.2.1. *Comparison to log-mel*

In this section, we analyze the behavior of raw waveform and log-mel CLDNNs as we increase the number of channels. The multichannel log-mel system is trained by computing log-mel for each channel, and treating these as separate feature maps into the CLDNN. Table 2 shows the results for 2, 4 and 8 channels. All of these experiments are run using $P = 128$ filters. Notice that log-mel shows improvements with more channels, which is consistent with [13], but does not improve as much as raw waveform. Since log-mel features are computed from the FFT magnitude, the fine time structure (stored in the phase), and therefore information about inter-microphone delays, is discarded. Log-mel models can therefore only make use of the weaker inter-microphone level difference cues. However, the multichannel time-domain filterbanks in the raw waveform models utilize the fine time structure and improve more as the number of channels increases.

Feature	1ch	2ch (14cm)	4ch (4-6-4cm)	8ch (2cm)
log-mel	24.4	22.0	21.7	22.0
raw	23.5	21.8	21.3	21.1

Table 2: WER for multichannel models, all use 128 filters.

Multi-microphone processing can help to enhance the signal and supress the noise. Therefore, we should expect to see improvements in WER, especially under more challenging conditions. A breakdown of WER in Figure 4 shows that this is indeed the case.

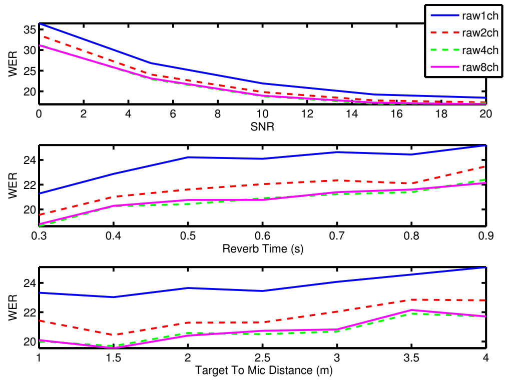

Fig. 4: WER breakdown for multichannel models.

4.2.2. *Increasing number of filters*

To help understand what the multichannel `tConv` filters are doing, Figure 5 plots the multichannel filter coefficients and the corresponding spatial responses, or beampatterns, after training. The beampatterns show the magnitude response in dB as a function of frequency and direction of arrival, i.e. each horizontal slice of the beampattern corresponds to the filter's magnitude response for a signal coming from a particular direction. In each frequency band (vertical slice), lighter shades indicate sounds from those angles are passed through, while darker shades indicate directions whose energy is filtered out, also known as a "null direction".

Similar to [9], the network tends to learn very similar filter coefficients in each channel, but shifted relative to each other, consistent with the concept of the steering delay τ_c described in Section 2. Most filters have bandpass response in frequency, often steered to have stronger response in a particular direction. Approximately 80% of the filters in the model shown in Figure 5 demonstrate a significant spatial response, i.e. show a difference of at least 6dB between the direction with the minimum and maximum response at the filter center frequency.

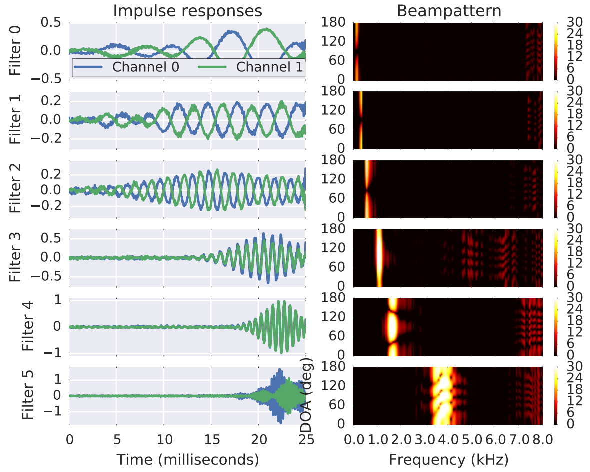

Fig. 5: Trained filters and spatial responses for a 2 channel network.

Because each filter has a fixed directional response, the number of filters limits the ability of the network to exploit directional cues. By increasing the number of filters, we can potentially improve the spatial diversity of the learned filters and therefore allow the network to better exploit directional cues. We can observe this in Figure 6, a scatter plot of filter maximum response frequency vs. null direction. Notice in the histogram at the bottom of the figure that as we increase the number of filters from 40 to 256, the diversity of null directions increases.

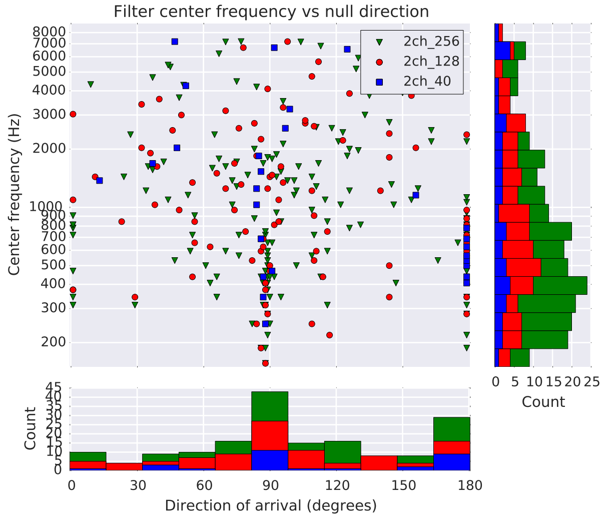

Fig. 6: Scatter plot of filter maximum response frequency vs. null direction for models with 40, 128, and 256 filters. "Null direction" is computed as the direction of minimum response for filters where the minimum response is at least 6dB below the maximum. (Filters for which minimum and maximum directional responses differ by less than 6dB are not included in the plot.)

Now that we have shown how increasing the number of filters helps to improve spatial diversity, Table 3 shows how increasing the filters affects WER. For 2 channels, improvements saturate at 128 filters, while for 4 and 8 channels we continue to get improvements at 256 filters. As we increase the number of input channels, more complex spatial responses can be learned, and therefore the network can benefit from more filters.

Filters	2ch (14cm)	4ch (4-6-4cm)	8ch (2cm)
128	21.8	21.3	21.1
256	21.7	20.8	20.6
512	-	20.8	20.6

Table 3: WER for raw and log-mel multichannel CLDNNs.

4.2.3. *Sequence training*

Finally, we evaluate the result of our multichannel raw waveform CLDNN after sequence training. Table 4 shows the WER, where the number of filters used for each channel is also indicated. The table shows a relative improvement of 6%, 11% and 11% over single channel raw model for 2, 4 and 8 channel models respectively.

Method	Filters (P)	WER - CE	WER - Seq
raw, 1ch	128	23.5	19.3
raw, 2ch	128	21.8	18.2
raw, 4ch	256	20.8	17.2
raw, 8ch	256	20.6	17.2

Table 4: Multichannel WER after sequence training.

4.3. Comparison to oracle knowledge of speech TDOA

Note that the models presented in the previous subsection do not explicitly estimate the time delay of arrival of the target source arriving at different microphones, which is commonly done in beamforming [2]. TDOA estimation is useful for two reasons: First, time aligning and combining signals steers the array such that the speech signal is enhanced relative to noise sources coming from other directions. Second, explicitly time aligning signals can make the overall system robust to differences in microphone geometry between the training and testing environments.

In this section, we analyze the behavior of raw waveform CLDNNs when the signals are time aligned using the true TDOA calculated using the room geometry. For the delay-and-sum approach, we shift each signal by the TDOA, average them together, and pass the result into a 1-channel raw waveform CLDNN. For the time-aligned multichannel (TAM) approach, we align the signals in time and pass them as separate channel inputs to a multichannel raw waveform CLDNN. Thus the difference between the multichannel raw waveform CLDNNs described in Section 2 and TAM is solely in how the data is presented to the network (whether or not they are first explicitly aligned to "steer" toward the target speaker direction); the network architectures are identical.

Table 5 compares the WER of D+S, TAM, and raw waveform models when we do not shift the signals by the TDOA. First, notice that as we increase the number of channels, D+S continues to improve, since finer spatial sampling of the signal makes it possible to steer a narrower beam, leading to increased suppression of noise and reverberation energy arriving from other directions. Second, notice that TAM always performs better than D+S, as TAM is more general than D+S because it allows individual channels to be filtered before being combined. But notice that the raw waveform CLDNN, without any explicit time alignment or localization (TDOA estimation), does as well as TAM with the time alignment. This shows us that the trained un-aligned network is implicitly robust to varying TDOA.

Feature	1ch	2ch (14cm)	4ch (4-6-4cm)	8ch (2cm)
D+S, tdoa	23.5	22.8	22.5	22.4
TAM, tdoa	23.5	21.7	21.3	21.3
raw, no tdoa	23.5	21.8	21.3	21.1

Table 5: WER with true TDOA.

However, one of the drawbacks of not estimating the TDOA is the network is not robust to mismatches in microphone geometry between training and test. To understand this better, Table 6 shows the WER for different networks trained on 2 channels (channels 1 and 8, with 14cm spacing), but evaluated both on matched channel spacing (channels 1 and 8) and mismatched spacing (e.g., channels 2 and 7 with 10cm spacing, channels 3 and 6 with 6cm spacing, and channels 4 and 5 with 2cm spacing). As a reference, we also show results for the single channel raw waveform case. Notice that when evaluating mismatched channels, D+S degrades slightly because the mismatched channels are more closely spaced, decreasing the spatial resolution of the D+S filter and therefore the degree of noise suppression. However, the performance with D+S for mismatched channels does not significantly degrade compared to a single channel. In contrast, both TAM and raw waveform degrade significantly.

Method	14cm	10cm	6cm	2cm
raw, 1ch	23.5	23.5	23.5	23.5
D+S, 2ch, tdoa	22.8	23.2	23.3	23.7
TAM, 2ch, tdoa	21.7	22.1	23.2	30.6
raw, 2ch	21.8	22.2	23.3	30.7

Table 6: WER as a function of microphone spacing for 2 channel systems trained on data with 14cm spacing.

The filters learned for TAM (not shown) appear qualitatively very similar to the raw waveform plot in Figure 5. Specifically, it steers nulls for matched channels but lets everything in for mismatched channels. While estimating TAM shifts the speech signal, this means both the target speech and the noise are shifted. The TAM network learns to filter out the noise according to its position relative to the target speaker signal. When we change the microphone spacing during test, and the position of target speaker and noise change, the network cannot adapt to this change of delay.

4.4. Adapting to mismatched array spacing

To make the network more robust to mismatched channels, we created a new 2 channel data set where we randomly sampled, with replacement, pairs of 2 channels from the original 8 channel array to create a "multi-geometry" data set. Performance of this model is shown in Table 7.

Here, we see that when we train a single network on multiple array geometries, the network learns to handle varying microphone spacings. In addition, Table 8 indicates if we repeat a single channel twice during decoding, the network recovers the performance of single channel. This shows that the same network can be used for single and multichannel tasks. In addition, this shows that an MTR-type approach to beamforming, where we give the network a variety of microphone configurations during training, allows the network to be robust to different configurations during test.

Method	14cm	10cm	6cm	2cm
raw, 2ch	21.8	22.2	22.3	30.7
raw, 2ch, multi-geo	21.9	21.7	21.9	21.8

Table 7: WER when trained on pairs of channels. All models use 128 filters.

Method	1ch, repeated twice
raw, 1ch	23.5
raw, 2ch	33.9
raw, 2ch, multi-geo	23.1

Table 8: WER when trained on pairs of channels and decoding single channel repeated twice.

Figure 7 shows example filters learned when trained on multi-geometry data. The filters appear qualitatively quite different to those shown in Figure 5. Specifically, the filters no longer exhibit strong spatial responses, i.e. they do not appear to steer nulls in different directions. Indeed, only 30% of the filters have a significant spatial response, less than half of those in the corresponding model trained on a fixed array geometry shown in Figure 5.

In other words, the network appears to handle candidate beamforming patterns for varying steering directions and varying mic configurations, and appears capable to infer the optimization with the upper layers of the network. This is in sharp contrast to the model-based estimation model and, since effective from empirical results, a more practical setup to combine enhancement and acoustic modeling in a single framework with a consistent optimization criterion.

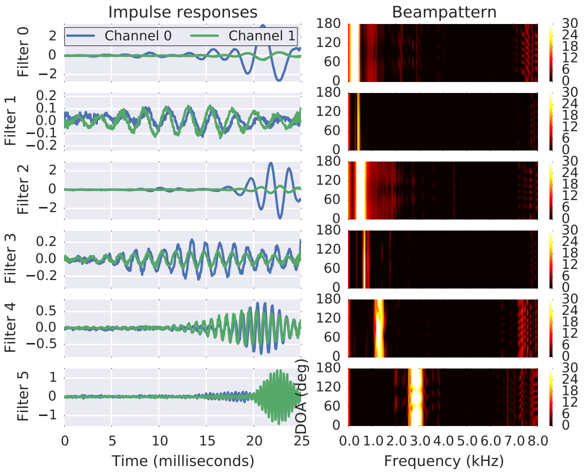

Fig. 7: Trained filters and their spatial responses for a 2 channel multi-geometry trained model. Note that the beampatterns are plotted assuming the widest microphone spacing of 14 cm.

Finally, we evaluate numbers after sequence training. We also show numbers on the "Rerecorded" data set. We see that the multi-geometry trained model is robust to microphone mismatches.

Method	WER - Simulated	WER - Rerecorded
raw, 1ch	19.3	23.8
raw, 2ch	18.2	23.7
D+S, 2ch, tdoa	19.2	23.3
raw, 2ch, multi-geo	17.8	21.1

Table 9: WER after sequence training, simulated and rerecorded.

5. CONCLUSIONS

In this paper, we presented a technique for doing beamforming jointly with acoustic modeling, using a raw waveform CLDNN. We showed that our raw waveform CLDNN offered improvements over log-mel CLDNNs, for both single and multiple channels. Analysis showed that the raw waveform model learns to do spatial filtering, and does as well as both D+S and TAM trained with the true TDOA. Finally, by training on multiple array geometries results in a model that is robust to a wide range of microphone spacings. The effective empirical results shown in this paper indicate that doing beamforming jointly with acoustic modeling within a single neural network framework appears to be a more practical setup compared to many model-based approaches.

6. ACKNOWLEDGEMENTS

Thank you to Yedid Hoshen for discussions related to raw waveform and multichannel processing, and to Bo Li for help with WER breakdown scripts.

7. REFERENCES

[1] M. Brandstein and D. Ward, *Microphone Arrays: Signal Processing Techniques and Applications*, Springer, 2001.

[2] J. Benesty, J. Chen, and Y. Huang, *Microphone Array Signal Processing*, Springer, 2009.

[3] T. Hain, L. Burget, J. Dines, P.N. Garner, F. Grezl, A.E. Hannani, M. Huijbregts, M. Karafiat, M. Lincoln, and V. Wan, "Transcribing Meetings with the AMIDA Systems," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 2, pp. 486–498, 2012.

[4] A. Stolcke, X. Anguera, K. Boakye, O. Çetin, A. Janin, M. Magimai-Doss, C. Wooters, and J. Zheng, "The SRI-ICSI Spring 2007 Meeting and Lecture Recognition System," *Multimodal Technologies for Perception of Humans*, vol. Lecture Notes in Computer Science, no. 2, pp. 450–463, 2008.

[5] B. D. Veen and K. M. Buckley, "Beamforming: A Versatile Approach to Spatial Filtering," *IEEE ASSP Magazine*, vol. 5, no. 2, pp. 4–24, 1988.

[6] M. Delcroix, T. Yoshioka, A. Ogawa, Y. Kubo, M. Fujimoto, N. Ito, K. Kinoshita, M. Espi, T. Hori, T. Nakatani, and A. Nakamura, "Linear Prediction-based Dereverberation with Advanced Speech Enhancement and Recognition Technologies for the REVERB Challenge," in *REVERB Workshop*, 2014.

[7] M. Seltzer, B. Raj, and R. M. Stern, "Likelihood-maximizing Beamforming for Robust Handsfree Speech Recognition," *IEEE Trascations on Audio, Speech and Language Processing*, vol. 12, no. 5, pp. 489–498, 2004.

[8] J. R. Hershey, J. Le Roux, and F. Weninger, "Deep Unfolding: Model-Based Inspiration of Novel Deep Architectures," *CoRR*, vol. abs/1409.2574, 2014.

[9] Y. Hoshen, R. J. Weiss, and K. W. Wilson, "Speech Acoustic Modeling from Raw Multichannel Waveforms," in *Proc. ICASSP*, 2015.

[10] A. Mohamed, G. Hinton, and G. Penn, "Understanding how Deep Belief Networks Perform Acoustic Modelling," in *ICASSP*, 2012.

[11] T. N. Sainath, R. J. Weiss, K. W. Wilson, A. Senior, and O. Vinyals, "Learning the Speech Front-end with Raw Waveform CLDNNs," in *Proc. Interspeech*, 2015.

[12] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, Long Short-Term Memory, Fully Connected Deep Neural Networks," in *Proc. ICASSP*, 2015.

[13] P. Swietojanski, A. Ghoshal, and S. Renals, "Hybrid Acoustic Models for Distant and Multichannel Large Vocabulary Speech Recognition," in *Proc. ASRU*, 2013.

[14] T. N. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep Convolutional Neural Networks for LVCSR," in *Proc. ICASSP*, 2013.

[15] T. N. Sainath, B. Kingsbury, A. Mohamed, G. Dahl, G. Saon, H. Soltau, T. Beran, A. Aravkin, and B. Ramabhadran, "Improvements to Deep Convolutional Neural Networks for LVCSR," in *Proc. ASRU*, 2013.

[16] H. Sak, A. Senior, and F. Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling," in *Proc. Interspeech*, 2014.

[17] J. B. Allen and D. A. Berkley, "Image Method for Efficiently Simulation Room-Small Acoustics," *Journal of the Acoustical Society of America*, vol. 65, no. 4, pp. 943 – 950, April 1979.

[18] J. Dean, G.S. Corrado, R. Monga, K. Chen, M. Devin, Q.V. Le, M.Z. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A.Y. Ng, "Large Scale Distributed Deep Networks," in *Proc. NIPS*, 2012.

[19] G. Heigold, E. McDermott, V. Vanhoucke, A. Senior, and M. Bacchiani, "Asynchronous Stochastic Optimization for Sequence Training of Deep Neural Networks," in *Proc. ICASSP*, 2014.

[20] X. Glorot and Y. Bengio, "Understanding the Difficulty of Training Deep Feedforward Neural Networks," in *Proc. AISTATS*, 2014.