Complex-valued Linear Layers for Deep Neural Network-based Acoustic Models for Speech Recognition

Izhak Shafran

Joint work with: Ehsan Variani, Arun Narayanan, and Tara Sainath

Google Inc
Outline

Background
Automatic Speech Recognition
Deep Neural Networks

Problem Definition
Motivation: Replace Hand-Crafted Feature Extraction
Problem Definition
One Approach: Learnable Time Domain Convolutional Filters

Learnable Complex-valued Frequency Domain Filters
Properties of the Model
Empirical Evaluations

Summary & Discussion
Automatic Speech Recognition

Feature Extraction
\[ X = T(S) \]

Acoustic Model
\[ P(X|W) \]

Decoder
\[ \text{argmax } P(W|X) \]

Language Model
\[ P(W) \]

“Call home”
Automatic Speech Recognition: Language Models
Scores likelihood of utterances in a given language

\[ P(\text{"call home"}) > P(\text{"call roam"}) \]

\[
P(w_1 w_2) = P(w_1 | \text{begin}) P(w_2 | w_1) P(\text{end} | w_1 w_2)
\]

\[
P(\text{"call home"}) = P(\text{"call"} | \text{begin}) \ldots
\]
Automatic Speech Recognition: Feature Extraction
Remove redundancy and irrelevant information
Automatic Speech Recognition: Feature Extraction
Remove redundancy and irrelevant information
Automatic Speech Recognition: Acoustic Models
Scores the conditional likelihood that the hypothesized words were spoken
Acoustic Models Illustration: Vowel classification task

Consider the (mag) FFT of a frame of (vowel) speech

Notice energy peaks (formants)
Turns out, their positions are different for different vowels
Acoustic Models Illustration: Vowel classification task
Values of \((F_1, F_2)\) for vowels in a training set
Acoustic Models Illustration: Vowel classification task
Score the conditional likelihood that a hypothesized vowel was observed

\[ P(X|\text{æ}) > P(X|\epsilon) \]
return "call home" if

\[ P(X|"call home")P("call home") > P(X|"call roam")P("call roam") \]

Mathematical justification,

\[ \hat{W} = \arg\max_W P(W|X) \]

\[ = \arg\max_W \frac{P(X|W)P(W)}{P(X)} \]

\[ = \arg\max_W P(X|W)P(W) \]
(Deep) Neural Networks

Illustration with a simple feed forward network
A Node in a Neural Network
Neural Networks
Illustration of Forward Propagation

\[ y_1 = f_1(w_{x11}x_1 + w_{x21}x_2) \]
Multistyle Training (MTR) improves robustness

\[ \delta = z - y \]
Deep Neural Networks: Why are they so hot now?

- Universal approximator
  - Superposition Theorem of Kolmogorov (1957)
  - A Computable Kolmogorov S. T. (Brattka, 2000)
  - Neural Network with Unbounded Activation Functions is a Universal Approximator (Sonoda & Murata, 2015)

- Back propagation algorithm
  - (Linnainmaa, 1970; Rummelhart et al, 1986)

- Efficient BackProp (LeCun 1998),
Outline

Background
  Automatic Speech Recognition
  Deep Neural Networks

Problem Definition
  Motivation: Replace Hand-Crafted Feature Extraction
  Problem Definition
  One Approach: Learnable Time Domain Convolutional Filters

Learnable Complex-valued Frequency Domain Filters
  Properties of the Model
  Empirical Evaluations

Summary & Discussion
In most ASR systems, features extraction:
- uses 3 decades old hand-crafted rules
- mimics human perception
- may not be optimal for speech recognition

Since neural networks are universal approximators, why not let the network learn the optimal features too!?
Recall: Feature Extraction

- Most problematic stage is Mel-filterbank
- Loss of information
- Log-mel features: useless for dereverberation or beamforming, precludes integration into neural networks
Problem

Can we feed the speech frames directly as input to the neural networks and learn the features jointly with acoustic models?
One Approach: Learnable Time Domain Filters (Hoshen et al, 2015)

- Skip transforming the signal into frequency domain
- Replace mel-filters with a convolutional neural network layer
- Learn the parameters of the layer (filter) from the data
- Implementation inspired by gammatone features
Gammatone Features

All the above operations can be approximated in the neural network.
Learnable Time Domain Filters (Hoshen et al, 2015)
Learned Time Domain Filters

- mel ($f_{break}=700$ Hz)
- gammatone untrained
- random init, MTR train
- gammatone init, MTR train
- gammatone init, clean train

Frequency (kHz)

Filter index

0 5 10 15 20 25 30 35 40
Learnable Time Domain Filters (Hoshen et al, 2015)

But:

- convolution layer has several parameters to tune
- computationally expensive
Convolution
Convolutional Layer

- Convolution layers use only the "valid" outputs

- Trade-off: Longer filters $\implies$ fewer valid outputs
For 64ms frame, there are still more settings to sweep!
Typically, max pooling works better than average pooling
Info loss, cannot cascade filters in the signal processing sense
Outline

Background
 Automatic Speech Recognition
 Deep Neural Networks

Problem Definition
 Motivation: Replace Hand-Crafted Feature Extraction
 Problem Definition
 One Approach: Learnable Time Domain Convolutional Filters

Learnable Complex-valued Frequency Domain Filters
 Properties of the Model
 Empirical Evaluations

Summary & Discussion
Convolution in Time $\equiv$ Multiplication in Frequency Domain

Time Domain:

$\begin{align*}
& x[n] \\
\ast & \quad h[k] \\
\end{align*}$

$= y[n]$

Frequency Domain:

$\begin{align*}
& X[n] \\
\cdot & \quad H[k] \\
\end{align*}$

$= Y[n]$
Multiplication in Frequency Domain: Complex-Value

- $X[k]$ is complex-valued
- $H[k]$ is complex-valued
- Need a linear layer with complex-valued neurons!
Let \( X := X_r + jX_i, \; W := W_r + jW_i, \; B := B_r + jB_i \)

\[
Z &= g(WX - B) \\
Z_r &= g(W_rX_r - W_iX_i - B_r) \\
Z_i &= g(W_iX_r + W_rX_i - B_i)
\]

- Easily implemented in a neural net w/ shared weights
- How is this different from using \( g([W_rW_i][X_rX_i]^T - B) \)?
Decision Boundaries are Orthogonal!

Easy to see from the norm to the boundary (Tohru Nitta, 2004)

\[ Z_r = g(W_r X_r - W_i X_i - B_r) \]
\[ Z_i = g(W_i X_r + W_r X_i - B_i) \]
\[ Q_r = \begin{bmatrix} \frac{\partial Z_r}{\partial X_r} & \frac{\partial Z_r}{\partial X_i} \end{bmatrix} \]
\[ = \begin{bmatrix} g'(\ldots) & g'(\ldots) \end{bmatrix} \begin{bmatrix} W_r \\ -W_i \end{bmatrix} \]
\[ Q_i = \begin{bmatrix} g'(\ldots) & g'(\ldots) \end{bmatrix} \begin{bmatrix} W_i \\ W_r \end{bmatrix} \]
\[ Q_r^T Q_i = 0 \implies Q_r \perp Q_i \]
Complex FFT Input & Complex-valued Linear Layers: The Appeal

- Orthogonal boundaries $\implies$ higher modeling capacity
e.g., XOR with one neuron!
- Gradient descent is faster
- Does not have trade-offs of convolutional layers
- L1 regularization is meaningful, unlike in time domain
- No max-pooling, allows cascading of filters
e.g., Dereverberation + Beamforming + Feature extraction
- Possibility to learn hybrid filters that optimize both signal processing cost function and WER
Complex-valued Linear Layers: Single Filter

\[ X \xrightarrow{\text{FFT}} X \xrightarrow{W} Y \xrightarrow{\log ||WX||} P \to \text{LSTM} \]

Dimensions: 257 x 40
Complex-valued Linear Layers: Set of Filters
A model is a lie that helps you see the truth.

Howard Skipper
Summation in the Frequency Domain

Lemma

*Summation in the frequency domain* \( \equiv \) *weighted average pooling in the time domain.*

If \( X \) is the Fourier transform of the \( 2N \) point signal \( x \), then

\[
\sum_{k=0}^{N} X_k = \sum_{n=0}^{2N-1} \alpha_n x_n
\]

where,

\[
\alpha_n = \begin{cases} 
N + 1 & n = 0 \\
\coth \left( j \frac{\pi i}{2N} \right) & \text{mod} \ (n, 2) = 1 \\
1 & \text{mod} \ (n, 2) = 0, \ n \neq 0
\end{cases}
\]
Proposition

The projection in the frequency domain is equivalent to convolution followed by a weighted average pooling:

\[
\sum_{j=0}^{N} W_{ij} X_{j} \longleftrightarrow \sum_{j=0}^{2N-1} \alpha_{j} (w_{i} \ast x) [j]
\]
Complex Linear Projection (CLP)
ASR Experimental Setup

- **Training set**: 2000 hrs, 3M anonymized voice search queries
  - SNR between 0-20dB ($\approx 12$ dB)
  - Reverb T60 between 400ms-900ms ($\approx 600$ms)
  - 8 channel linear mic with spacing of 2cm
  - Noise and target speaker location constant per utterance

- **Acoustic model**
  - A cascade of 3 LSTMs followed by a DNN layer
  - 13K HMM states
  - Learning with asynchronized stochastic gradient descent

- **Test set**: Voice search w/ matching conditions
- **All results reported here are after CE training**
Experimental Settings & WER Results

- 25 ms, 32 ms, 64 ms
- 128 filters

<table>
<thead>
<tr>
<th>System</th>
<th>25 ms</th>
<th>32 ms</th>
<th>64 ms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log-mel</td>
<td>23.4</td>
<td>22.8</td>
<td>21.8</td>
</tr>
<tr>
<td>Time-domain conv.</td>
<td>23.7</td>
<td>23.4</td>
<td>22.5</td>
</tr>
<tr>
<td>CLP</td>
<td>23.2</td>
<td>22.8</td>
<td>22.0</td>
</tr>
</tbody>
</table>

- CLP is at least as good as the alternatives!
Learned CLP Filters: Before and After Applying L1 Regularization
Learned Filters: Histogram of Learned Filter Coefficients

Time-domain filter coefficients (in time & in freq)

CLF filter coefficients (w/o & w/ L1)

CLP filters are sparse!
Complex Linear Projection: Two Channels

- Input channels: $x_1$, $x_2$
- FFT applied to $x_1$ and $x_2$
- Matrix multiplication $WX$ with $W 	imes 514 	imes 40$
- Logarithmic function $\log ||WX||$
- Output to LSTM
Experimental Settings & WER Results

- Two channels from microphones, 14cm apart
- 25 ms, 32 ms, 64 ms
- 256 filters

<table>
<thead>
<tr>
<th>System</th>
<th>25 ms</th>
<th>32 ms</th>
<th>64 ms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log-mel</td>
<td>21.8</td>
<td>21.3</td>
<td>20.7</td>
</tr>
<tr>
<td>Time-domain conv.</td>
<td>21.5</td>
<td>21.2</td>
<td>21.2</td>
</tr>
<tr>
<td>CLP</td>
<td>21.5</td>
<td>20.9</td>
<td>20.5</td>
</tr>
</tbody>
</table>

- Again, CLP is at least as good as the other alternatives!
Multichannel CLP: Frame Size vs Projection Size

- Frame size: 32ms, 64ms
- Projection size: 128, 256, 512
- Without L1 regularization and stop bands

<table>
<thead>
<tr>
<th></th>
<th>128</th>
<th>256</th>
<th>512</th>
</tr>
</thead>
<tbody>
<tr>
<td>32ms</td>
<td>21.7</td>
<td>21.4</td>
<td>21.7</td>
</tr>
<tr>
<td>64ms</td>
<td>21.1</td>
<td>21.1</td>
<td>21.5</td>
</tr>
</tbody>
</table>

- Higher projections size $\implies$ larger input to LSTM, slower
- Step time is about $1.5 \times$ slower going from 256 to 512
- Frame size has no impact for 32ms and 64ms
- Step time is about $2 \times$ slower going from 1 to 8 channels
Learned CLP Filters: Two Channel
Learned CLP Filters: Time-delay between Real and Imaginary

Filter number vs time delay
Histogram of Time-delay between Real and Imaginary

Time delay vs frequency of occurrence
Language and Task Dependency

- English high-pitch voice search (en-us-hp)
- Taiwanese voice search (cmn-hant-tw)

<table>
<thead>
<tr>
<th>System</th>
<th>en-us-hp</th>
<th>cmn-hant-tw</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log-mel</td>
<td>16.6</td>
<td>17.2</td>
</tr>
<tr>
<td>Time-domain conv.</td>
<td>16.3</td>
<td>16.8</td>
</tr>
<tr>
<td>CLP</td>
<td>16.4</td>
<td>16.6</td>
</tr>
</tbody>
</table>

- CLP as good or better than the alternatives!
Learned Filters: Sorted by Center Frequency
Computational Efficiency

- Time-domain filters $\approx O(Nkp + k^2 p))$
  - $k$ strides, $2N$ window size, $p$ filters
- CLP $\approx O(pN)$

<table>
<thead>
<tr>
<th>Model</th>
<th>Time Filters</th>
<th>CLP</th>
</tr>
</thead>
<tbody>
<tr>
<td>num-params</td>
<td>45K</td>
<td>66K</td>
</tr>
<tr>
<td>num-add-mult</td>
<td>14.51 M</td>
<td>263.17 K</td>
</tr>
</tbody>
</table>

- Computation cost is a magnitude lower!
Summary & Future Directions

- Introduced complex-valued linear layers for acoustic modeling
- Complex-valued linear projection (CLP) as one instance of it
- Showed a simple multi-channel extension for CLP
- Reported experimental comparisons with log-mel and time-domain filters
- Demonstrated task-dependent advantage
- Discussed properties of the CLP model
- Future directions:
  - More efficient multi-channel projections, hybrid systems with WER and beamforming-type cost functions, adding multiple complex layers with appropriate complex-valued activation functions, integrating de-reverberation, ...