Speaker Identification

Problem and Application

Speaker recognition can be used to provide secured personalized interactions to systems controlled by voice.

Global-password text dependent speaker recognition aims to distinguish among speakers using fixed phrases like “OK Google” or “Hey Google”.

Since 2014 end2end neural network architectures for speaker recognition have shown to outperform traditional approaches. As a result, most of the recent work is focused on end2end architectures for text-dependent speaker recognition.

This work is a study over advanced attention mechanisms to further improve the standard end2end architectures for text-dependent speaker recognition.

End to End Architecture

Baseline LSTM Model

We use 3-Layer LSTM in our baseline LSTM model.

Dimension of each layer - 128

Projection layer in each layer with dimension - 64

On top of the LSTM layers, a linear layer of dimension - 64

The acoustic parameters consist of 48-dimensional log-likelihood filterbank coefficients computed over a window of 25ms with 10ms frame shift.

Problems

Silence and background noise are NOT being well captured in this system.

The phonemes are usually surrounded by frames of silence and background noise.

Speaker embedding should be built only using the frames corresponding to phonemes.

Thus, we propose to use an attention layer as a soft mechanism to emphasize the most relevant elements of the input sequence.

Attention-based models on LSTM

Basic Idea

Motivation - A weighted combination of all hidden layer outputs to learn the more important parts of the input.

In our baseline LSTM model, computation of the weights is done on the final hidden layer outputs.

We use a scoring function $v_i = f(x_i)$ to compute weights based on the hidden status.

Then we get the normalized weights using $w_i = \frac{e^{v_i}}{\sum_{j=1}^{L} e^{v_j}}$.

Finally, we compute a weighted combination of the weights as:

$$\sum_{i=1}^{L} w_i x_i$$

Different Scoring Functions

We experimented using different types of scoring functions for computation of weights in the attention layer.

1. Only attention - It uses L2 norm on the LSTM output and weight.

2. Linear and non-linear attention - We call attention linear and non-linear based on the function used to calculate the attention.

3. Shared-parameters attention - We experimented using shared parameters through all time steps for linear and non-linear attentions.

Attention Layer Variants

We introduce two variants of the attention layer -

1. Bias only attention
2. Linear and non-linear attention
3. Shared-parameters attention

Motivation - To make the weights sparse. Sparse weights can bring more on the important parts with temporal variation in speech.

We used two variants for the weights pooling idea in our design.

1. Sliding window maxpooling - We ran sliding window on the weights. For each window, only keep the largest value and get others to 0.

2. Global log-k maxpooling - We only keep the largest k values in the weights, and set all others to 0.

Weights Pooling Idea and Variants

Attention Visualization

Results

Experiments

Datasets and Evaluation

Motivation - We visualize the attention weights of a training batch for different pooling methods.

- Intersecting observation - when there’s no pooling, we see a clear 4-strand or 3-strand pattern in the bank. This pattern corresponds to the “OK Google”/“Hey Google”/“Verify” structure of the keywords.

- When we apply sliding window maxpooling or global log-k maxpooling, the attention weights are much localized to the new end of the utterance. The LSTM has accumulated more information at the near-end than at the beginning, thus is more confident to produce the d-vector.

Baseline LSTM Model with divided-layer attention

Motivation - Using same layer for weight computation and d-vector computation is very inefficient.

We calculate weights from an intermediate LSTM layer.

We change the scoring function to $v_i(x_i, y)$ where, $y$ is an intermediate LSTM layer (e.g. second-to-last layer).

We calculate the weights from the weighted average of the last $L$ as before.

Cross-layer attention

Motivation - Using independent layers for weight computation and d-vector computation, we can use the size of the final LSTM layer and then divide it into two equal parts $k_1$ and parts $k_2$.

We compute the d-vector from parts by the scoring function $v_i(x_i, y_i)$.

We calculate the d-vector from the weighted average of the parts $k_1$.

Divided-layer attention

Training set

Test set

Table Evaluation EER(%): Basic attention layer vs. variants - all training.

<table>
<thead>
<tr>
<th>Method</th>
<th>EER(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Attention</td>
<td>0.94</td>
</tr>
<tr>
<td>Cross-layer attention</td>
<td>0.77</td>
</tr>
<tr>
<td>Cross-layer attention</td>
<td>0.72</td>
</tr>
<tr>
<td>Divided-layer attention</td>
<td>0.75</td>
</tr>
<tr>
<td>Divided-layer attention</td>
<td>0.71</td>
</tr>
<tr>
<td>Shared-parameters attention</td>
<td>0.72</td>
</tr>
</tbody>
</table>

For each experiment, we calculate the EER and Evaluation of the baseline model.

We compare the basic attention layer with cross-layer and divided-layer attentions using the best scoring function from previous experiments.

We compare different pooling strategies for feature selection from previous experiment.

We visualize sliding window maxpooling better than other two variants.

Performance wise non-linear with shared parameter is better than others. We compare our basic cross-layer and divided-layer attentions using the best scoring function from previous experiments.

Table Evaluation EER(%): Basic attention layer vs. variants - all training.

<table>
<thead>
<tr>
<th>Method</th>
<th>EER(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Attention</td>
<td>0.94</td>
</tr>
<tr>
<td>Cross-layer attention</td>
<td>0.77</td>
</tr>
<tr>
<td>Cross-layer attention</td>
<td>0.72</td>
</tr>
<tr>
<td>Divided-layer attention</td>
<td>0.75</td>
</tr>
<tr>
<td>Divided-layer attention</td>
<td>0.72</td>
</tr>
<tr>
<td>Shared-parameters attention</td>
<td>0.72</td>
</tr>
</tbody>
</table>

For each experiment, we calculate the EER and Evaluation of the baseline model.