Towards End-to-End Speech Recognition with Recurrent Neural Networks

Alex Graves, Navdeep Jaitly
In previous speech recognition, the neural networks are at present only a single component in a complex pipeline.

**Shortage of previous speech recognition system:**
The frame-level training targets must be inferred from the alignments determined by the HMM.
A pronunciation dictionary is necessary to map from words to phoneme sequences.
Create pronunciation dictionary costs a lot of labor.
The quality of the dictionary affects the result of speech recognition system dramatically.

**Alignment**

**Pronunciation dictionary**

**Improvement of this paper**
Using a deep bidirectional LSTM network with a Connectionist Temporal Classification output layer.
No pronunciation dictionary is necessary.
CTC integrates out over all possible input-output alignments, no forced alignment is required to provide training targets.
Bidirectional Recurrent Neural Networks

In the speech recognition, we need consider the information from both past and future
Add backward Layer to Standard RNN to construct Bidirectional RNN

\[ h_t = \mathcal{H} (W_{ih} x_t + W_{hh} h_{t-1} + b_h) \]  \hspace{1cm} (1)

\[ y_t = W_{ho} h_t + b_o \]  \hspace{1cm} (2)

Input vector \( x=(x_1, \ldots, x_T) \)
Hidden vector \( h=(h_1, \ldots, h_T) \)
Output vector \( y=(y_1, \ldots, y_T) \)
Optimize \( W_{ih} \), \( W_{hh} \) and \( W_{ho} \)

Graves Towards End-to-End Speech Recognition with Recurrent Neural Networks
http://www.shareditor.com/blogshow/?blogId=116
Long Short Term Memory

Standards RNN performs bad for long-term dependencies because gradients prorogation over many stages tend to either vanish or explode. In practice, Long Short Term Memory is widely used to preserve long-term dependencies.

The key to LSTMs is the cell state, the horizontal line running through the top of the diagram. It’s very easy for information to just flow along it unchanged.

http://colah.github.io/posts/2015-08-Understanding-LSTMs/
Long Short Term Memory

The first step in our LSTM is to decide what information we’re going to throw away from the cell state.

\[ f_t = \sigma (W_f[h_{t-1}, x_t] + b_f) \]

The next step is to decide what new information we’re going to store in the cell state.

\[ i_t = \sigma (W_i[h_{t-1}, x_t] + b_i) \]
\[ \tilde{C}_t = \tanh(W_C[h_{t-1}, x_t] + b_C) \]

Forget Gate

Input gate: which values we’ll update

New candidate values, added to the state

http://colah.github.io/posts/2015-08-Understanding-LSTMs/
Long Short Term Memory

Update the old cell state $C_{t-1}$, into the new cell state $C_t$

Finally, we need to decide what we’re going to output. This output will be based on our cell state, but will be a filtered version.
Deep Recurrent Neural Network

Bidirectional RNN

Output
Hidden
Input

Deep Bidirectional RNN

Output
Multi Hidden Layers
Input

More ability to learning

http://blog.csdn.net/heyongluoyao8/article/details/48636251
Connectionist Temporal Classification (CTC)

**Typical frame-level classifier:**
Requires a training target for every frame in the audio
(1) circular dependency:
  good alignment \(\langle\rightarrow\rangle\) good classifier
(2) no need for character-level alignment

**Using CTC as objective function:**
Allows an RNN to be trained without any prior alignment between the input and the transcription
Connectionist Temporal Classification (CTC)

\( x \): Input sequence with length \( T \)
\( a \): output sequence of blank and label indices (CTC alignment)
\( y \): output transcription
\( B \): matching CTC alignment to transcription

Example:
\[ a_1 = (a, -, b, c, -, -) \]  \[ a_2 = (-, -, a, -, b, c) \]
\[ a_3 = (a, -, b, -, c, c) \]  \[ a_4 = (a, a, b, b, c, c) \]

\[ y = (a, b, c) \]

\[ B(a_1) = B(a_2) = B(a_3) = B(a_4) = y \]

\[
Pr(y|x) = \sum_{a \in B^{-1}(y)} Pr(a|x)
\]

\[
CTC(x) = - \log Pr(y^*|x)
\]
Expected Transcription Loss

$L(x, y)$: transcription loss function (e.g. WER)
$L(x)$: expected transcription loss

$$L(x) = \sum_y \Pr(y|x) L(x, y)$$

Approximate with Monte-Carlo sampling

$$L(x) \approx \frac{1}{N} \sum_{i=1}^{N} L(x, B(a^i)), \quad a^i \sim \Pr(a|x)$$

Differentiate $L(x)$ with respect to RNN outputs

$$\frac{\partial L(x)}{\partial \Pr(k, t|x)} \approx \frac{1}{N} \sum_{i=1}^{N} L(x, B(a^i, t, k))$$
Expected Transcription Loss

\[
\frac{\partial \mathcal{L}(x)}{\partial y^k_t} \approx \frac{\Pr(k, t|x)}{N} \sum_{i=1}^{N} \mathcal{L}(x, B(a^i)) - \mathcal{Z}(a^i, t)
\]

where

\[
\mathcal{Z}(a^i, t) = \sum \Pr(k', t|x) \mathcal{L}(x, B(a^{i,t,k'}))
\]

The derivative added to \(y^k_t\) equals to the difference between the loss with \(a^i_t = k\) and the expected loss with \(a^i_t\) sampled from \(\Pr(k', t|x)\)

Encourage outputs changing that alter the loss functions
Expected Transcription Loss

Target: HIS_FRIENDS_
Output: HIS_FRIEND’S_
Changing of the apostrophe is encouraged
Decoding

Decode the output of CTC to a sequence of result: beam search algorithms

Doing pruning doing the search and cut the nodes with less probability and only keep top n nodes

Experiments

Table 1. Wall Street Journal Results. All scores are word error rate/character error rate (where known) on the evaluation set. ‘LM’ is the Language model used for decoding. ‘14 Hr’ and ‘81 Hr’ refer to the amount of data used for training.

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>LM</th>
<th>14 Hr</th>
<th>81 Hr</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN-CTC</td>
<td>NONE</td>
<td>74.2/30.9</td>
<td>30.1/9.2</td>
</tr>
<tr>
<td>RNN-CTC</td>
<td>DICTIONARY</td>
<td>69.2/30.0</td>
<td>24.0/8.0</td>
</tr>
<tr>
<td>RNN-CTC</td>
<td>MONOGRAM</td>
<td>25.8</td>
<td>15.8</td>
</tr>
<tr>
<td>RNN-CTC</td>
<td>BIGRAM</td>
<td>15.5</td>
<td>10.4</td>
</tr>
<tr>
<td>RNN-CTC</td>
<td>TRIGRAM</td>
<td>13.5</td>
<td>8.7</td>
</tr>
<tr>
<td>RNN-WER</td>
<td>NONE</td>
<td>74.5/31.3</td>
<td>27.3/8.4</td>
</tr>
<tr>
<td>RNN-WER</td>
<td>DICTIONARY</td>
<td>69.7/31.0</td>
<td>21.9/7.3</td>
</tr>
<tr>
<td>RNN-WER</td>
<td>MONOGRAM</td>
<td>26.0</td>
<td>15.2</td>
</tr>
<tr>
<td>RNN-WER</td>
<td>BIGRAM</td>
<td>15.3</td>
<td>9.8</td>
</tr>
<tr>
<td>RNN-WER</td>
<td>TRIGRAM</td>
<td>13.5</td>
<td>8.2</td>
</tr>
<tr>
<td>BASELINE</td>
<td>NONE</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>BASELINE</td>
<td>DICTIONARY</td>
<td>56.1</td>
<td>51.1</td>
</tr>
<tr>
<td>BASELINE</td>
<td>MONOGRAM</td>
<td>23.4</td>
<td>19.9</td>
</tr>
<tr>
<td>BASELINE</td>
<td>BIGRAM</td>
<td>11.6</td>
<td>9.4</td>
</tr>
<tr>
<td>BASELINE</td>
<td>TRIGRAM</td>
<td>9.4</td>
<td>7.8</td>
</tr>
<tr>
<td>COMBINATION</td>
<td>TRIGRAM</td>
<td>—</td>
<td>6.7</td>
</tr>
</tbody>
</table>

Performance is close to the baseline, while baseline has a lot of priority knowledge. Improve 1% when combine the model.
Examples: (NO dictionary or language model used for decoding)

target: TO ILLUSTRATE THE POINT A PROMINENT MIDDLE EAST ANALYST IN WASHINGTON RECOUNTS A CALL FROM ONE CAMPAIGN
output: TWO ALSTRAIT THE POINT A PROMINENT MIDDLE EAST ANALYST IM WASHINGTON RECONNCACALL FROM ONE CAMPAIGN

target: T. W. A. ALSO PLANS TO HANG ITS BOUTIQUE SHINGLE IN AIRPORTS AT LAMBERT SAINT
output: T. W. A. ALSO PLANS TOHING ITS BOOTIK SINGLE IN AIRPORTS AT LAMBERT SAINT

target: ALL THE EQUITY RAISING IN MILAN GAVE THAT STOCK MARKET INDIGESTION LAST YEAR
output: ALL THE EQUITY RAISING IN MULONG GAVE THAT STACRK MARKET IN TO JUSTIAN LAST YEAR

target: THERE’S UNREST BUT WE’RE NOT GOING TO LOSE THEM TO DUKAKIS
output: THERE’S UNREST BUT WERE NOT GOING TO LOSE THEM TO DEKAKIS
Thanks!