End-to-End Speech Recognition using Deep LSTMs, CTC Training and WFST Decoding

@ MIT  Dec 07 2015

Yajie Miao
Carnegie Mellon University
Outline

• Motivation

• End-to-End Speech Recognition
  — Deep LSTM Models
  — CTC Training
  — WFST-based Decoding

• Experiments & Analysis

• Conclusions
Outline

- Motivation
- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding
- Experiments & Analysis
- Conclusions
Why End-to-End?

ImageNet Classification with Deep Convolutional Neural Networks. Krizhevsky et al.

End-to-End Image Classification
Why End-to-End?

Sequence to Sequence Learning with Neural Networks. Sutskever et al. 2014.

End-to-End Machine Translation

“Mary is in love with John”

“玛丽爱上了约翰”
The HMM/GMM or HMM/DNN pipelines are highly complex

- Multiple training stages: CI phone, CD senones, ...
- Various resources: dictionaries, decision trees, ...
- Many super-parameters: number of senones, number of Gaussians, ...

Complexity of ASR
End-to-End ASR!

- ASR is a sequence-to-sequence learning problem
- A simpler paradigm with a single model (and training stage)

“I am in Boston today”
Outline

- Motivation

- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding

- Experiments & Analysis

- Conclusions
Outline

- Motivation

- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding

- Experiments & Analysis

- Conclusions
LSTM Models

- RNNs model temporal dependency across speech frames.
- Long short-term memory (LSTM) units.
  - Memory cells store the history information.
  - Various gates control the information flow inside the LSTM.
  - Advantageous in learning long-term temporal dependency.
LSTM Models

- LSTMs outperform DNNs in the hybrid approach [Sainath et al., Miao et al.]
- This is uni-directional LSTM, i.e., forward LSTM.

\[
\begin{align*}
\mathbf{i}_t &= \sigma(\mathbf{W}_{ix}\mathbf{x}_t + \mathbf{W}_{ih}\mathbf{h}_{t-1} + \mathbf{W}_{ic}\mathbf{c}_{t-1} + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{W}_{fx}\mathbf{x}_t + \mathbf{W}_{fh}\mathbf{h}_{t-1} + \mathbf{W}_{fc}\mathbf{c}_{t-1} + \mathbf{b}_f) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \phi(\mathbf{W}_{cx}\mathbf{x}_t + \mathbf{W}_{ch}\mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_{ox}\mathbf{x}_t + \mathbf{W}_{oh}\mathbf{h}_{t-1} + \mathbf{W}_{oc}\mathbf{c}_t + \mathbf{b}_o) \\
\mathbf{h}_t &= \mathbf{o}_t \odot \phi(\mathbf{c}_t)
\end{align*}
\]

input gate \hspace{1cm} forget gate \hspace{1cm} memory cell \hspace{1cm} output gate
Bi-directional LSTMs

\[ h(t) = [h_f(t), h_b(t)] \]
Deep BiLSTM Model

BiLSTM

BiLSTM

BiLSTM

softmax

affine transform

t-1  t  t+1
Outline

• Motivation

• End-to-End Speech Recognition
  — Deep LSTM Models
  — CTC Training
  — WFST-based Decoding

• Experiments & Analysis

• Conclusions
Connectionist Temporal Classification

- CTC is a sequence-to-sequence learning technique [Graves et al.]

\[ L_{CTC} = \ln Pr(\mathbf{z} | \mathbf{X}) \]

- Label sequence: “A B C”
- Observations: \( \mathbf{X} = (x_1, \ldots, x_T) \)
Connectionist Temporal Classification

- CTC is a sequence-to-sequence learning technique [Graves et al.]

\[ L_{CTC} = \ln Pr(z \mid X) \]

- Label sequence: “A B C”
- Observations:
  \[ X = (x_1, \ldots, x_T) \]
  \[ Y = (y_1, \ldots, y_T) \]
CTC Paths

- CTC is a sequence-to-sequence learning technique [Graves et al.]

\[ L_{CTC} = \ln Pr(z \mid X) \]

CTC is a sequence-to-sequence learning technique [Graves et al.]

- CTC paths bridge frame-level labels with the label sequence
  - A CTC path is a sequence of labels on the frame level \( p = [p_1, \ldots, p_T] \)
  - The likelihood of a CTC path is decomposed onto the frames:

\[ Pr(p \mid X) = \prod_{t=1}^{T} y_t^{p_t} \]
CTC Paths

- CTC paths differ from labels sequences in that:
  - Add the blank as an additional label, meaning no (actual) labels are emitted
  - Allow repetitions of non-blank labels

\[
\begin{array}{cccccc}
A & A & \emptyset & \emptyset & B & C & \emptyset \\
\emptyset & A & A & B & \emptyset & C & C \\
\emptyset & \emptyset & \emptyset & A & B & C & \emptyset \\
\end{array}
\]

collapse

\[\begin{array}{ccc}
A & B & C \\
\end{array}\]

expand

- Many-to-one mapping from CTC paths \(\Phi(z)\) to the label sequence \(z\)

\[
Pr(z \mid X) = \sum_{p \in \Phi(z)} Pr(p \mid X)
\]

Computationally Intractable !!
### Forward-Backward Algorithm

<table>
<thead>
<tr>
<th></th>
<th>∅</th>
<th>A</th>
<th>∅</th>
<th>B</th>
<th>∅</th>
<th>C</th>
<th>∅</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Expanded labels: \( l \)

\[ \alpha_t (s) \]

Summed probability of all the CTC paths ending at \( t \) with \( s \)

\[
\alpha_1 (\emptyset) = y_{\emptyset}^1 \quad \alpha_1 (A) = y_A^1 \quad \alpha_1 (s) = 0, \quad \forall s > 2
\]
Forward Computation

\[ \alpha_t(s) = \begin{cases} 
 y_{l_s}^t \left( \alpha_{t-1}(s) + \alpha_{t-1}(s-1) \right) & \text{if } l_s = \emptyset \text{ or } l_s = l_{s-2} 
\end{cases} \]

\[ \alpha_t(s) \]

summed probability of all the CTC paths ending at \( t \) with \( s \)

expanded labels \( l \)
**Forward Computation**

\[
\alpha_t(s) = \begin{cases} 
  y^t_{l_s} \left( \alpha_{t-1}(s) + \alpha_{t-1}(s-1) \right) & \text{if } l_s = \emptyset \text{ or } l_s = l_{s-2} \\
  y^t_{l_s} \left( \alpha_{t-1}(s) + \alpha_{t-1}(s-1) + \alpha_{t-1}(s-2) \right) & \text{otherwise}
\end{cases}
\]

expanded labels $l$

\[
\begin{array}{c|c|c|c}
\emptyset & \emptyset & \emptyset \\
A & \emptyset & \emptyset \\
\emptyset & \emptyset & \emptyset \\
B & \emptyset & \emptyset \\
\emptyset & \emptyset & \emptyset \\
C & \emptyset & \emptyset \\
\emptyset & \emptyset & \emptyset \\
\end{array}
\]

\[
\begin{array}{c|c|c|c}
1 & 2 & 3 \\
\hline
T-2 & T-1 & T \\
\end{array}
\]

\(\alpha_t(s)\) summed probability of all the CTC paths ending at \(t\) with \(s\)
### Backward Computation

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\emptyset$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\emptyset$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\emptyset$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### $\beta_T (s)$

- Summed probability of all the CTC paths starting at $t$ with $s$

#### Formulas

$$\beta_T (\emptyset) = y^T_{\emptyset} \quad \beta_T (C) = y^T_C \quad \beta_T (s) = 0, \quad \forall s < |l| - 1$$
Backward Computation

\begin{align*}
\beta_t(s) &= \begin{cases} 
  y_{l_s}^t \left( \beta_{t+1}(s) + \beta_{t+1}(s+1) \right) & \text{if } l_s = \emptyset \text{ or } l_s = l_{s+2} \\
  y_{l_s}^t \left( \beta_{t+1}(s) + \beta_{t+1}(s+1) + \beta_{t+1}(s+2) \right) & \text{otherwise}
\end{cases}
\end{align*}

expanded labels \( l \)
CTC Training

- Evaluation of the objective $Pr(z \mid X)$

$$Pr(z \mid X) = \sum_{s=1}^{||x||} \alpha_t(s)\beta_t(s) \quad L_{CTC} = \ln Pr(z \mid X)$$

- Gradients w.r.t. the pre-softmax network outputs:

CTC

$$\frac{\partial L_{CTC}}{\partial a_k^t} = y_k^t - \frac{1}{Pr(z \mid X) \sum_{s \in B(1,k)} \alpha_t(s)\beta_t(s)}$$

soft labels

CE

$$\frac{\partial L_{CE}}{\partial a_k^t} = y_k^t - g_k^t$$

hard labels
What Happens during CTC Training?
Outline

- Motivation

- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding

- Experiments & Analysis

- Conclusions
CTC Decoding

• Difficulty of CTC decoding
  — Previous work proposed beam search for CTC [Graves et al, Hannun et al.]
  — Incorporating word language models efficiently was difficult
  — It was challenging to deal with the behaviors of blanks

• WFST-based Decoding
  — 3 WFSTs encode 3 components required in decoding

• The language model WFST $G$

![](image.png)
Lexicon -- L

- Maps sequences of lexicon units to words.
- **Phonemes** as CTC labels: the standard dictionary WFST
  
  \[ \begin{array}{c}
  0 \xrightarrow{\text{IH : is}} 1 \xrightarrow{Z : <\text{eps}>} 2 \\
  \end{array} \]

  - **Characters as CTC labels**
    - Contains word spellings, easy to include OOV words
    - The space `<space>` between each pair of words is taken as a label
    - Allows `<space>` to appear optionally at the beginning and end of the word
Token -- $T$

- Maps a (segment of) CTC path to a lexicon unit (phonemes or characters)

- Allows occurrences of blanks and repetitions of non-blank labels
Search Graph & Posterior Scaling

- The 3 WFSTs are composed into a search graph

\[ S = T \circ \min(det(L \circ G)) \]

\( \circ \) – composition \hspace{1em} \( det \) – determinization \hspace{1em} \( \min \) – minimization

- The search graph encodes the mapping from a sequence of frame-level CTC labels to a sequence of words.

- During decoding, scale the label posteriors with their priors

\[ p(x_t \mid k) \propto p(k \mid x_t)/p(k) \]

where the prior \( p(k) \) is estimated from the expanded label sequences (\( \emptyset A \emptyset B \emptyset C \emptyset \)) from the training set by simple counting.
# Eesen Recipe - Switchboard

https://github.com/yajiemiao/eesen

# Use the same datap preparation script from Kaldi
local/swbd1_data_prep.sh $swbd  || exit 1;

# Construct the phoneme-based lexicon
local/swbd1_prepare_phn_dict.sh  || exit 1;

# Compile the lexicon and token FSTs
utils/ctc_compile_dict_token.sh data/local/dict_phn data/local/lang_phn_tmp data/lang_phn  || exit 1;

# Train and compile LMs.
local/swbd1_train_lms.sh data/local/train/text data/local/dict_phn/lexicon.txt data/local/lm $fisher_dirs

# Compile the language-model FST and the final decoding graph TLG.fst
local/swbd1_decode_graph.sh data/lang_phn data/local/dict_phn/lexicon.txt  || exit 1;

# Data preparation for the eval2000 set
local/eval2000_data_prep.sh $eval2000_dirs

# Generate the fbank features; by default 40-dimensional fbanks on each frame
steps/make_fbank.sh --cmd "$train_cmd" --nj 32 data/train exp/make_fbank/train $fbankdir  || exit 1;
utils/fix_data_dir.sh data/train  || exit;
steps/compute_cmvn_stats.sh data/train exp/make_fbank/train $fbankdir  || exit 1;

steps/make_fbank.sh --cmd "$train_cmd" --nj 10 data/eval2000 exp/make_fbank/eval2000 $fbankdir  || exit 1;
utils/fix_data_dir.sh data/eval2000  || exit;
steps/compute_cmvn_stats.sh data/eval2000 exp/make_fbank/eval2000 $fbankdir  || exit 1;
# Specify network structure and generate the network topology
input_feat_dim=120  # dimension of the input feature
lstm_layer_num=4   # number of LSTM layers
lstm_cell_dim=320  # number of memory cells in every LSTM layer

dir=exp_110h/train_phn_l${lstm_layer_num}_c${lstm_cell_dim}
mkdir -p $dir

# Output the network topology
utils/model_topo.py --input-feat-dim $input_feat_dim --lstm-layer-num $lstm_layer_num \ 
   --lstm-cell-dim $lstm_cell_dim --target-num $target_num \ 
   --fgate-bias-init 1.0 > $dir/nnet.proto || exit 1;

# Label sequences; simply convert words into their label indices
utils/prep_ctc_trans.py data/lang_phn/lexicon_numbers.txt data/train_100k_nodup/text "<unk>" | gzip -c -
utils/prep_ctc_trans.py data/lang_phn/lexicon_numbers.txt data/train_dev/text "<unk>" | gzip -c - > $dir/

# Train the network with CTC. Refer to the script for details about the arguments
steps/train_ctc_parallel.sh --add-deltas true --num-sequence 10 --frame-num-limit 25000 \ 
   --learn-rate 0.00004 --report-step 1000 --halving-after-epoch 12 \ 
data/train_100k_nodup data/train_dev $dir || exit 1;

Decoding

# decoding
for lm_suffix in swl_tg swl_fsh_tgpr; do
   steps/decode_ctc_lat.sh --cmd "$decode_cmd" --nj 20 --beam 17.0 --lattice_beam 8.0 --max-active 5000 --acwt data/lang_phn_${lm_suffix} data/eval2000 $dir/decode_eval2000_${lm_suffix} || exit 1;
done
Eesen Recipe - Switchboard

[Link to GitHub repository]

```
# Specify network structure and generate the network topology
input_feat_dim=120    # dimension of the input features
lstm_layer_num=4      # number of layers
lstm_cell_dim=320     # size of hidden states

dir=exp_110h/train_phn
mkdir -p $dir

# Output the network topology
utils/model_topo.py --input $lstm_layer_num
                        --lstm-cell-dim $lstm_cell_dim
                        --fgate-bias-init 1.0
                        --
                        --
                        --

# Label sequences; simple
utils/prepare_ctc_trans.py

# Train the network with CTC
steps/train_ctc_parallel $dir
                        --learn-rate 0.00004
                        --data-train $dir
                        --data-dev $dir
                        --max-epochs 25000

# Decoding

# decoding
for lm_suffix in swl_tg swl_fsh tgpr; do
    steps/decode_ctc_lat.sh --cmd "$decode_cmd" --nj 20
                       --beam 17.0
                       --lattice-beam 8.0
                       --max-active 5000
                       --acwt 0.0
    data/lang_phn_${lm_suffix} data/eval2000 $dir/decode_eval2000_${lm_suffix} || exit 1;
done

```

- HMM
- GMM
- Phonetic decision trees
- Multiple training stages
- Dictionaries, if characters are CTC labels
- ... ...

Model Training

- NO

...
Outline

- Motivation
- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding
- Experiments & Analysis
- Conclusions
Results on WSJ

• Experimental Setup
  - 4 bi-directional LSTM layers, each with 640 memory cells
  - 40-dimensional log-mel filterbank features, plus $\Delta$ and $\Delta\Delta$
  - Training utterances are sorted by their lengths, and 10 utterances are processed in a batch each time
  - WERs on the eval92 testing set

• Phoneme-based Systems
  - The CMU dictionary as the lexicon
  - 72 labels including phonemes, noise marks and the blank

• Character-based Systems
  - 59 labels including letters, digits, punctuation marks and the blank.
## Results on WSJ

<table>
<thead>
<tr>
<th>Models</th>
<th>Vocabulary</th>
<th>LM</th>
<th>WER%</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTC</td>
<td>Original</td>
<td>NIST</td>
<td>7.87</td>
</tr>
<tr>
<td>Hybrid DNN</td>
<td>Original</td>
<td>NIST</td>
<td>7.14</td>
</tr>
</tbody>
</table>
# Results on WSJ

<table>
<thead>
<tr>
<th>Models</th>
<th>Vocabulary</th>
<th>LM</th>
<th>WER%</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTC</td>
<td>Original</td>
<td>NIST</td>
<td>7.87</td>
</tr>
<tr>
<td>Hybrid DNN</td>
<td>Original</td>
<td>NIST</td>
<td>7.14</td>
</tr>
<tr>
<td>CTC</td>
<td>Original</td>
<td>NIST</td>
<td>9.07</td>
</tr>
<tr>
<td>CTC</td>
<td>Expanded</td>
<td>Retrained</td>
<td>7.34</td>
</tr>
<tr>
<td>Graves et al.</td>
<td>Expanded</td>
<td>Retrained</td>
<td>8.7</td>
</tr>
<tr>
<td>Hannun et al.</td>
<td>Original</td>
<td>Unknown</td>
<td>14.1</td>
</tr>
</tbody>
</table>
Results on Switchboard

- **Experimental Setup**
  - 300 hours of training speech; tested on the SWBD part of Hub5’00
  - 5 bi-directional LSTM layers, each with 640 memory cells
  - CTC labels: 46 phonemes (including the blank)

- **Initialization of Forget-gate Bias**
  - Initializing the forget gate bias to a larger value helps LSTM learn long-term dependency [Jozefowicz et al.]
  - The bias of the forget gates is initialized to **1.0**

\[
\begin{align*}
    i_t &= \sigma(W_{ix}x_t + W_{ih}h_{t-1} + W_{ic}c_{t-1} + b_i) \\
    f_t &= \sigma(W_{fx}x_t + W_{fh}h_{t-1} + W_{fc}c_{t-1} + b_f) = 1.0 \\
    c_t &= f_t \odot c_{t-1} + i_t \odot \phi(W_{cx}x_t + W_{ch}h_{t-1} + b_c) \\
    o_t &= \sigma(W_{ox}x_t + W_{oh}h_{t-1} + W_{oc}c_t + b_o)
\end{align*}
\]
## Results on Switchboard

<table>
<thead>
<tr>
<th>Models</th>
<th>FG Bias</th>
<th>WER%</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTC</td>
<td>0</td>
<td>15.7</td>
</tr>
<tr>
<td>CTC</td>
<td>1.0</td>
<td>15.0</td>
</tr>
</tbody>
</table>
## Results on Switchboard

<table>
<thead>
<tr>
<th>Models</th>
<th>#Model Param</th>
<th>WER%</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTC</td>
<td>11M</td>
<td>15.0</td>
</tr>
<tr>
<td>Hybrid DNN</td>
<td>40M</td>
<td>16.9</td>
</tr>
<tr>
<td>Hybrid LSTM</td>
<td>12M</td>
<td>15.8</td>
</tr>
</tbody>
</table>

For fairness, all the systems use the filterbank features.
## Results on Switchboard

<table>
<thead>
<tr>
<th>Models</th>
<th>#Model Param</th>
<th>WER%</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTC</td>
<td>11M</td>
<td><strong>15.0</strong></td>
</tr>
<tr>
<td>Hybrid DNN</td>
<td>40M</td>
<td>16.9</td>
</tr>
<tr>
<td>Hybrid LSTM</td>
<td>12M</td>
<td>15.8</td>
</tr>
</tbody>
</table>

### 300-Hour Training

<table>
<thead>
<tr>
<th>Models</th>
<th>#Model Param</th>
<th>WER%</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTC</td>
<td>8M</td>
<td>19.9</td>
</tr>
<tr>
<td>Hybrid DNN</td>
<td>12M</td>
<td>20.2</td>
</tr>
<tr>
<td>Hybrid LSTM</td>
<td>8M</td>
<td><strong>19.2</strong></td>
</tr>
</tbody>
</table>
## Decoding Efficiency

<table>
<thead>
<tr>
<th>Models</th>
<th>Decoding Graph</th>
<th>Graph Size</th>
<th>RTF*</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTC</td>
<td>TLG</td>
<td>123M</td>
<td>0.71</td>
</tr>
<tr>
<td>Hybrid DNN</td>
<td>HCLG</td>
<td>216M</td>
<td>1.43</td>
</tr>
<tr>
<td>Hybrid LSTM</td>
<td>HCLG</td>
<td>216M</td>
<td>1.12</td>
</tr>
</tbody>
</table>

* RTF – Real time factor
Frame Skipping

blanks
Frame Skipping

<table>
<thead>
<tr>
<th>#Frames Skipped</th>
<th>Decoding RTF</th>
<th>WER%</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.71</td>
<td>15.0</td>
</tr>
<tr>
<td>1</td>
<td>0.38</td>
<td>15.8</td>
</tr>
<tr>
<td>2</td>
<td>0.25</td>
<td>16.5</td>
</tr>
</tbody>
</table>
Frame Skipping

<table>
<thead>
<tr>
<th>#Frames Skipped</th>
<th>Decoding RTF</th>
<th>WER%</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.71</td>
<td>15.0</td>
</tr>
<tr>
<td>1</td>
<td>0.38</td>
<td>15.8</td>
</tr>
<tr>
<td>2</td>
<td>0.25</td>
<td>16.5</td>
</tr>
<tr>
<td>&lt; 100Hrs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>110Hrs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>----</td>
<td>19.9</td>
</tr>
</tbody>
</table>
Results on HKUST Mandarin

- Experimental Setup
  - Chinese Mandarin conversational telephone speech [Liu et al.]
  - 175 hours of training
  - CTC labels: 3600+ Chinese characters

<table>
<thead>
<tr>
<th>Models</th>
<th>Features</th>
<th>CER%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid DNN</td>
<td>FBank</td>
<td>39.42</td>
</tr>
<tr>
<td>CTC</td>
<td>FBank</td>
<td>39.70</td>
</tr>
<tr>
<td>CTC</td>
<td>FBank+Pitch</td>
<td>38.67</td>
</tr>
</tbody>
</table>
Results on Multilingual CTC

- **Experimental Setup**
  - BABEL multilingual corpora
  - Around 80 hours of training speech per language
  - Features: filterbank + pitch

<table>
<thead>
<tr>
<th>Models</th>
<th>Training Hrs</th>
<th>#Labels</th>
<th>CTC WER%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tagalog</td>
<td>84.5</td>
<td>48</td>
<td>51.5</td>
</tr>
<tr>
<td>Turkish</td>
<td>77.2</td>
<td>51</td>
<td>54.5</td>
</tr>
<tr>
<td>Pashto</td>
<td>78.4</td>
<td>54</td>
<td>56.6</td>
</tr>
</tbody>
</table>
Outline

- Motivation
- End-to-End Speech Recognition
  - Deep LSTM Models
  - CTC Training
  - WFST-based Decoding
- Experiments & Analysis
- Conclusions
Conclusions

- We have presented a complete end-to-end ASR framework
- CTC models achieve comparable performance to the hybrid approach
- There are still a lot of unknowns about CTC

Open Questions

- How can we perform Keyword Search (KWS) over CTC models?
- How can we add context-dependency to CTC labels?
- How can we estimate i-vectors [Dehak et al.]?
- How can we perform pre-training of the deep BiLSTM models?
- ... ...
References

Questions?

@ MIT Dec 07 2015

Yajie Miao
Carnegie Mellon University