Mixed TRF LMs

Integrating Discrete and Neural Features via Mixed-Feature Trans-Dimensional Random Field Language Models

Silin Gao¹, Zhijian Ou¹, Wei Yang², Huifang Xu³

¹Speech Processing and Machine Intelligence (SPMI) Lab, Tsinghua University
²State Grid Customer Service Center
³China Electric Power Research Institute

http://oa.ee.tsinghua.edu.cn/ouzhijian/

Presented at International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2020
1. Introduction
   - Related Work
   - Motivation

2. Mixed TRF LMs
   - Definition
   - Training

3. Experiments
   - PTB
   - Google one-billion word

4. Conclusions
Introduction

• Language Modeling
  ■ For the word sequence $\mathbf{x} \triangleq x_1 x_2 \cdots x_l$, determine the joint probability $p(\mathbf{x})$

• Directed Graphical Language Models
  ■ Self-normalized, modeling conditional probabilities
  ■ e.g. N-gram language models, Neural network (NN) based language models (e.g. RNN/LSTM LMs)
    $$P(x_1, x_2, x_3, x_4) = P(x_1)P(x_2|x_1)P(x_3|x_2)P(x_4|x_1, x_3)$$

• Undirected Graphical Language Models
  ■ Involves the normalizing constant $Z$, potential function $\Phi$
  ■ e.g. Trans-dimensional random field language models (TRF LMs)
    $$P(x_1, x_2, x_3, x_4) = \frac{1}{Z} \Phi(x_1, x_2)\Phi(x_2, x_3)\Phi(x_3, x_4)\Phi(x_1, x_4)$$
Related Work: N-gram LMs

• N-gram Language Models

\[ p(x_1, x_2, \ldots, x_l) = \prod_{i=1}^{l} p(x_i | x_1, \ldots, x_{i-1}) \]

\[ \approx \prod_{i=1}^{l} p(x_i | x_{i-n+1}, \ldots, x_{i-1}) \]

• Back-off N-gram LMs with Kneser-Ney Smoothing\(^1\) (KNn LMs)

\[ p_{KN}(x_i|h) = (1 - \alpha_{KN}(h))\hat{p}(x_i|h) + \alpha_{KN}(h)p_{KN}(x_i|h') \]

\[ h = x_{i-n+1} \cdots x_{i-1} = x_{i-n+1}h' \]

Related Work: RNNs/LSTM LMs

- Recurrent Neural Nets (RNNs)/Long-Short Time Memory (LSTM) Language Models

\[
p(x_i | x_1, \cdots, x_{i-1}) \approx p(x_i | h_{i-1}(x_1, \cdots, x_{i-1})) \approx \frac{h_{i-1}^T w_k}{\sum_{k=1}^{V} h_{i-1}^T w_k}
\]

1. High computational cost of the Softmax output layer
   e.g. \( V = 10^4 \sim 10^6 \), \( w_k \in \mathbb{R}^{250 \sim 1024} \)

2. “Label bias” caused by the teacher-forcing training of the local conditional probabilities

---

"Label bias" caused by the teacher-forcing training of the local conditional probabilities

\[ p(x_i | x_1, \cdots, x_{i-1}) \approx p(x_i | h_{i-1}(x_1, \cdots, x_{i-1})) \approx \frac{h_{i-1}^T w_k}{\sum_{k=1}^{V} h_{i-1}^T w_k} \]


- Sundermeyer, 2012
Related Work: TRF LMs

• Trans-Dimensional Random Field (TRF) Language Models

Assume the sentences of length $l$ are distributed as follows:

$$p_l(x^l; \eta) = \frac{1}{Z_l(\eta)} e^{V(x^l; \eta)}, \quad x^l \triangleq x_1 x_2 \cdots x_l$$

$x^l \triangleq x_1, x_2, \ldots, x_l$ is a word sequence with length $l$;
$V(x^l; \eta)$ is the potential function extracting the features of $x^l$;
$\eta$ is the parameter of the potential function;
$Z_l(\eta) = \sum_{x^l} e^{V(x^l; \eta)}$ is the normalization constant.

Assume length $l$ is associated with prior probability $\pi_l$.

Therefore the pair $(l, x^l)$ is jointly distributed as:

$$p(l, x^l; \eta) = \pi_l \cdot p_l(x^l; \eta)$$
Related Work: TRF LMs

\[ p(l, x^l; \eta) = \frac{\pi_l}{Z_l(\eta)} e^{V(x^l; \eta)}, \ x^l \triangleq x_1 x_2 \cdots x_l \]

1. **Flexible**: no acyclic and local normalization constraint

Discrete TRF:

<table>
<thead>
<tr>
<th>Type</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td>((w_{-3}w_{-2}w_{-1}w_0)(w_{-2}w_{-1}w_0)(w_{-1}w_0)(w_0))</td>
</tr>
<tr>
<td>c</td>
<td>((c_{-3}c_{-2}c_{-1}c_0)(c_{-2}c_{-1}c_0)(c_{-1}c_0)(c_0))</td>
</tr>
<tr>
<td>ws</td>
<td>((w_{-3}w_0)(w_{-3}w_{-2}w_0)(w_{-3}w_{-1}w_0)(w_{-2}w_0))</td>
</tr>
<tr>
<td>cs</td>
<td>((c_{-3}c_0)(c_{-3}c_{-2}c_0)(c_{-3}c_{-1}c_0)(c_{-2}c_0))</td>
</tr>
<tr>
<td>wsh</td>
<td>((w_{-4}w_0)(w_{-5}w_0))</td>
</tr>
<tr>
<td>csh</td>
<td>((c_{-4}c_0)(c_{-5}c_0))</td>
</tr>
<tr>
<td>cpw</td>
<td>((c_{-3}c_{-2}c_{-1}w_0)(c_{-2}c_{-1}w_0)(c_{-1}w_0))</td>
</tr>
<tr>
<td>tied</td>
<td>((c_{-9}; -6, c_0)(w_{-9}; -6, w_0))</td>
</tr>
</tbody>
</table>

2. **Avoid high computational cost of the Softmax and “label bias”**

- The state-of-the-art Neural TRF LMs perform as good as LSTM LMs, and are computationally more efficient in inference (computing sentence probabilities)
Related Work: TRF LMs

- The development of TRF LMs

<table>
<thead>
<tr>
<th>Year</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACL-2015</td>
<td>• Discrete features&lt;br&gt;• Augmented stochastic approximation (AugSA) for model training</td>
</tr>
<tr>
<td>TPAMI-2018</td>
<td>• Discrete features&lt;br&gt;• Augmented stochastic approximation (AugSA) for model training</td>
</tr>
<tr>
<td>ASRU-2017</td>
<td>• Potential function as a deep CNN.&lt;br&gt;• Model training by AugSA plus JSA (joint stochastic approximation)</td>
</tr>
<tr>
<td>ICASSP-2018</td>
<td>• Use LSTM on top of CNN&lt;br&gt;• Noise Contrastive Estimation (NCE) is introduced to train TRF LMs</td>
</tr>
<tr>
<td>SLT-2018</td>
<td>• Simplify the potential definition by using only Bidirectional LSTM&lt;br&gt;• Propose Dynamic NCE for improved model training</td>
</tr>
</tbody>
</table>
Motivation

- **Language models using discrete features (N-gram LMs, Discrete TRF LMs)**
  - Mainly capture local lower-order interactions between words
  - Better suited to handling symbolic knowledges

- **Language models using neural features (LSTM LMs, Neural TRF LMs)**
  - Able to learn higher-order interactions between words
  - Good at learning smoothed regularities due to word embeddings

- **Interpolation of LMs**\(^1,2\): usually achieves further improvement
  - Discrete and neural features have complementary strength. 😊
  - Two-step model training is sub-optimal. 😞

---


Motivation

TRF LMs: \[ p(l,x^l; \eta) = \frac{\pi_l}{Z_l(\eta)} e^{V(x^l, \eta)}, \ x^l \triangleq x_1 x_2 \cdots x_l \]

1. TRF LMs are flexible to support both discrete and neural features

<table>
<thead>
<tr>
<th>Type</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td>(w_{-3} w_{-2} w_{-1} w_0) (w_{-2} w_{-1} w_0) (w_{-1} w_0) (w_0)</td>
</tr>
<tr>
<td>c</td>
<td>(c_{-3} c_{-2} c_{-1} c_0) (c_{-2} c_{-1} c_0) (c_{-1} c_0) (c_0)</td>
</tr>
<tr>
<td>ws</td>
<td>(w_{-3} w_0) (w_{-2} w_{-1} w_0) (w_{-1} w_0) (w_{-2} w_0)</td>
</tr>
<tr>
<td>cs</td>
<td>(c_{-3} c_0) (c_{-3} c_{-2} c_0) (c_{-3} c_{-1} c_0) (c_{-2} c_0)</td>
</tr>
<tr>
<td>wsh</td>
<td>(w_{-4} w_0) (w_{-5} w_0)</td>
</tr>
<tr>
<td>csh</td>
<td>(c_{-4} c_0) (c_{-5} c_0)</td>
</tr>
<tr>
<td>cpw</td>
<td>(c_{-3} c_{-2} c_{-1} w_0) (c_{-2} c_{-1} w_0) (c_{-1} w_0)</td>
</tr>
<tr>
<td>tied</td>
<td>(c_{-9} c_{-6} c_0) (w_{-9} c_{-6} w_0)</td>
</tr>
</tbody>
</table>

Discrete features

Neural features

Achieve feature integration in an optimal single-step model construction! (Mixed-feature TRF)

2. Lower the non-convexity

- Speed up convergence and reduce training time

3. Complementary strength in language modeling

- Further improve the performance of TRF LMs by using diversified features
Content

1. Introduction
   - Related Work
   - Motivation

2. Mixed TRF LMs
   - Definition
   - Training

3. Experiments
   - PTB
   - Google one-billion word

4. Conclusions
Mixed TRF LMs: Definition

- Mixed TRF LMs:
  \[ p(l, x^l; \eta) = \frac{\prod_{l} e^{V(x^l, \eta)}}{Z_l(\eta)}, \quad V(x^l, \eta) = \lambda^T f(x^l) + \phi(x^l; \theta), \quad \eta = (\lambda, \theta) \]

Discrete n-gram features, with parameter \( \lambda \):

\[ f(x^l) = (f_1(x^l), f_2(x^l), \ldots, f_N(x^l)) \]

\( N \): the total number of types of n-grams

\[ f_k(x^l) = c \]

where \( c \) is the count of the \( k \)th n-gram type in \( x^l \)

\[ x^l = \text{he is a teacher and he is also a good father} \]

\[ f_{he\ is}(x^l) = \text{count of “he is” in } x^l = 2 \]

\[ f_{a\ teacher}(x^l) = \text{count of “a teacher” in } x^l = 1 \]

Neural network features, with parameter \( \theta \)

\[ \phi(x^l; \theta) = \sum_{i=1}^{l-1} h_{f,i}^T e_{i+1} + \sum_{i=2}^{l} h_{b,i}^T e_{i-1} \]
Mixed TRF LMs: Training, Noise Contrastive Estimation

- Treat $\log Z_l(\eta)$ as a parameter $\zeta_l$ and rewrite

$$p(l, x^l; \eta) = \frac{\pi_l}{Z_l(\eta)} e^{V(x^l, \eta)}$$

$$p(x; \xi) = \pi_l e^{V(x^l, \eta) - \zeta_l}, x = (l, x^l), \xi = (\eta, \zeta_l)$$

- Introduce a **noise distribution** $q_n(x)$, and consider a binary classification

$$P(C = 0|x) = \frac{p(x; \xi)}{p(x; \xi) + \nu q_n(x)}$$, where $\nu = \frac{P(C = 1)}{P(C = 0)}$

$$P(C = 1|x) = 1 - P(C = 0|x)$$

- Noise Contrastive Estimation (NCE):

$$\max_{\xi} E_{x \sim p_0(x)}[\log P(C = 0|x)] + E_{x \sim q_n(x)}[\log P(C = 1|x)]$$

😊 Reliable NCE needs a large $\nu \approx 20$; Overfitting.

Dynamic-NCE\(^1\) in Wang & Ou, SLT 2018.

---

\(^1\)Bin Wang and Zhijian Ou, “Improved training of neural trans-dimensional random field language models with dynamic noise-contrastive estimation,” in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 70–76.
Content

1. Introduction
   - Related Work
   - Motivation

2. Mixed TRF LMs
   - Definition
   - Training

3. Experiments
   - PTB
   - Google one-billion word

4. Conclusions
Experiments: n-best list rescoring

• Two sets of experiments over two training datasets of different scales
  - Penn Treebank (PTB) dataset:
    16K sentences, 10K vocabulary (after preprocessing)
  - Google one-billion-word dataset:
    31M sentences, 568K vocabulary (after cutting off words counting less than 4)

• Test set for LM n-best list rescoring
  - Wall Street Journal (WSJ) ’92 dataset:
    330 sentences, each corresponds to a 1000-best list

• Implemented with Tensorflow

Open-source: https://github.com/thu-spmi/SPMILM
Experiments: PTB dataset

<table>
<thead>
<tr>
<th>Model</th>
<th>PPL</th>
<th>WER (%)</th>
<th>#param (M)</th>
<th>Training time</th>
<th>Inference time</th>
</tr>
</thead>
<tbody>
<tr>
<td>KN5</td>
<td>141.2</td>
<td>8.78</td>
<td>2.3</td>
<td>22 seconds</td>
<td>0.06 seconds</td>
</tr>
<tr>
<td>LSTM-2×1500</td>
<td>78.7</td>
<td>7.36</td>
<td>66.0</td>
<td>23.6 hours</td>
<td>9.09 seconds</td>
</tr>
<tr>
<td>Discrete TRF</td>
<td>~128</td>
<td>8.37</td>
<td>2.3</td>
<td>7.28 hours</td>
<td>0.11 seconds</td>
</tr>
<tr>
<td>Neural TRF</td>
<td>~75</td>
<td>7.34</td>
<td>2.6</td>
<td>22.1 hours</td>
<td>0.08 seconds</td>
</tr>
<tr>
<td>Mixed TRF</td>
<td>~69</td>
<td>7.17</td>
<td>4.9</td>
<td>18.2 hours</td>
<td>0.12 seconds</td>
</tr>
</tbody>
</table>

- Compared to the LSTM-2×1500, Mixed TRF achieves a 2.6% relative reduction on word error rate (WER), with 77.1% training time and only 7.4% parameters.

- Mixed TRF is 76x faster in inference (rescoring sentences) than the LSTM-2×1500.

- Compared to the state-of-the-art Neural TRF, Mixed TRF achieves a 2.3% relative reduction on word error rate (WER), with 82.4% training time, and comparable parameter size and inference speed.
Experiments: PTB dataset

WER curves of the three TRF LMs during the first 100 training epochs:

- Mixed TRF converges faster than the state-of-the-art Neural TRF, using only 58% training epochs.

😊 The discrete features in Mixed TRF lower the non-convexity of the optimal problem, and reduce the amount of patterns for neural features to capture.
Experiments: PTB dataset

More rescoring results of various interpolated LMs:

<table>
<thead>
<tr>
<th>Model</th>
<th>WER (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mixed TRF</td>
<td>7.17</td>
</tr>
<tr>
<td>LSTM-2×1500 + KN5</td>
<td>7.47</td>
</tr>
<tr>
<td>Neural TRF + KN5</td>
<td>7.30</td>
</tr>
<tr>
<td>LSTM-2×1500 + Discrete TRF</td>
<td>7.15</td>
</tr>
<tr>
<td>Neural TRF + Discrete TRF</td>
<td>7.17</td>
</tr>
<tr>
<td>LSTM-2×1500 + Neural TRF</td>
<td>7.01</td>
</tr>
<tr>
<td>LSTM-2×1500 + Neural TRF + KN5</td>
<td>6.89</td>
</tr>
<tr>
<td>LSTM-2×1500 + Mixed TRF</td>
<td>6.83</td>
</tr>
<tr>
<td>LSTM-2×1500 + Mixed TRF + KN5</td>
<td>6.82</td>
</tr>
</tbody>
</table>

“+” denotes the log-linear interpolation with equal weights

- Mixed TRF matches the best interpolated model combining a discrete-feature LM and a neural-feature LM together.
- Updating Neural TRF to Mixed TRF is beneficial in language model interpolations.
Experiments: Google one-billion-word dataset

<table>
<thead>
<tr>
<th>Model</th>
<th>PPL</th>
<th>WER (%)</th>
<th>#param (M)</th>
<th>Training time</th>
<th>Inference time</th>
</tr>
</thead>
<tbody>
<tr>
<td>KN5</td>
<td>94.5</td>
<td>6.13</td>
<td>133</td>
<td>2.48 hours</td>
<td>0.491 seconds</td>
</tr>
<tr>
<td>LSTM-2×1024</td>
<td>72.7</td>
<td>5.55</td>
<td>191</td>
<td>144 hours</td>
<td>0.909 seconds</td>
</tr>
<tr>
<td>Discrete TRF</td>
<td>~86</td>
<td>6.04</td>
<td>102</td>
<td>131 hours</td>
<td>0.022 seconds</td>
</tr>
<tr>
<td>Neural TRF</td>
<td>~72</td>
<td>5.47</td>
<td>114</td>
<td>336 hours</td>
<td>0.017 seconds</td>
</tr>
<tr>
<td>Mixed TRF</td>
<td>~68</td>
<td>5.28</td>
<td>216</td>
<td>297 hours</td>
<td>0.024 seconds</td>
</tr>
</tbody>
</table>

Note: To reduce parameter size and speed up inference, we adopt a small-scale LSTM LM, and apply adaptive softmax strategy¹.

- Compared to the LSTM-2×1024 with adaptive softmax, Mixed TRF achieves a 4.9% relative reduction on word error rate (WER) and a 38x inference speed, though having a bit more parameters and longer training time.

- Compared to the state-of-the-art Neural TRF, Mixed TRF achieves a 3.5% relative reduction on word error rate (WER) with 88.4% training time.

- The LM interpolation results are similar to those on PTB.

Results of various interpolated LMs:

<table>
<thead>
<tr>
<th>Model</th>
<th>WER (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mixed TRF</td>
<td>5.28</td>
</tr>
<tr>
<td>LSTM-2×1024 + KN5</td>
<td>5.38</td>
</tr>
<tr>
<td>Neural TRF + KN5</td>
<td>5.51</td>
</tr>
<tr>
<td>LSTM-2×1024 + Discrete TRF</td>
<td>5.31</td>
</tr>
<tr>
<td>Neural TRF + Discrete TRF</td>
<td>5.27</td>
</tr>
<tr>
<td>LSTM-2×1024 + Neural TRF</td>
<td>5.25</td>
</tr>
<tr>
<td>LSTM-2×1024 + Neural TRF + KN5</td>
<td>5.06</td>
</tr>
<tr>
<td>LSTM-2×1024 + Mixed TRF</td>
<td>5.02</td>
</tr>
<tr>
<td>LSTM-2×1024 + Mixed TRF + KN5</td>
<td>4.99</td>
</tr>
</tbody>
</table>

1. Introduction
   ■ Related Work
   ■ Motivation

2. Mixed TRF LMs
   ■ Definition
   ■ Training

3. Experiments
   ■ PTB
   ■ Google one-billion word

4. Conclusions
Conclusions

• We propose a mixed-feature TRF LM and demonstrate its advantage in integrating discrete and neural features.

• The Mixed TRF LMs trained on PTB and Google one-billion datasets achieve strong results in n-best list rescoring experiments for speech recognition.
  ■ Mixed TRF LMs outperform all the other single LMs, including N-gram LMs, LSTM LMs, Discrete TRF LMs and Neural TRF LMs;
  ■ The performance of Mixed TRF LMs matches the best interpolated model, and with simplified one-step training process and reduced training time;
  ■ Interpolating Mixed TRF LMs with LSTM LMs and N-gram LMs can further improve rescoring performance and achieve the lowest word error rate (WER).

• Next: Apply Mixed TRF LMs to one-pass ASR.
Thanks for your attention!

Silin Gao\textsuperscript{1}, Zhijian Ou\textsuperscript{1}, Wei Yang\textsuperscript{2}, Huifang Xu\textsuperscript{3}

\textsuperscript{1}Speech Processing and Machine Intelligence (SPMI) Lab, Tsinghua University
\textsuperscript{2}State Grid Customer Service Center
\textsuperscript{3}China Electric Power Research Institute

http://oa.ee.tsinghua.edu.cn/ouzhijian/