On Using Monolingual Corpora in Neural Machine Translation &
Towards better decoding and integration of language models in sequence to sequence models

Rebekka Hubert

Department of Computational Linguistics (ICL)
University of Heidelberg
1 Monolingual Corpora in NMT

2 On language models in end-to-end systems

3 Conclusions
Motivation

- NMT systems suffer from low resources and domain restriction
- monolingual corpora exhibit linguistic structure
- integrate language model trained on monolingual corpora into a NMT model suffering from low resources or domain restriction
Encoder-decoder framework

Figure 1: overview of the model [Bahdanau et al., 2014]
Encoder

- bidirectional RNN
- sequence of annotation vectors is concatenation of pairs of hidden states

\[
h_j^T = [\overleftarrow{h}_j^T; \overrightarrow{h}_j^T]
\]

- \( h_j \) encodes information about \( j \)-word w.r.t. all of the surrounding words in the sentence
emulates searching through a source sentence during decoding at each timestep $t$

- compute $s_t = f_r(s_{t-1}, y_{t-1}, c_t)$
- computes context vector $c_t$ (expected annotation)

$$c_t = \sum_{j=1}^{T_x} \alpha_{tj} h_j$$

with

$$\alpha_{tj} = \frac{\exp(e_{tj})}{\sum_{k=1}^{T_x} \exp(e_{tk})}; e_{tj} = a(s_{t-1}, h_j, y_{t-1})$$

- $a$: soft-alignment model (feedforward neural network)
- $\alpha$: attention
Decoder

- compute probability of target word $y_t$ with deep output layer

$$p(y_t \mid y_{<t}) \propto \exp(y_t^T (W_o f_o(s_t^{TM}, y_{t-1}, c_t) + b_o))$$

- $y_t$: k-dimensional, one-hot encoded vector indicating one of the words in target vocabulary
  - for large target vocabularies: importance sampling technique [Jean et al., 2014]
- $W_o \in \mathbb{R}^{K_y \times I}$: weight matrix
- $f_o$: neural network with two-way maxout non-linearity
- $b_o$: bias
Sampling technique

- partition training corpus and define small subset $V'$ of unique target words for each partition
- at each update only the vectors associated with the sampled words in $V'$ and the correct word are updated
- equivalent to

\[
p(y_t \mid y_{<t}, x) = \frac{\exp\{y_t^T \phi(y_{t-1}, s_t, c_t) + b_t\}}{\sum_{y_k \in V'} \exp\{y_k^T \phi(y_{t-1}, s_t, c_t) + b_k\}}
\]

- *approximates* exact output probability
Maxout Networks

- characterized by the Maxout activation function
- Generalization of ReLu: \( h_i(x) = \max_{j \in [1,k]} z_{ij} \)
- \( z_{ij} = x^T W_{ij} + b_{ij}; \ W \in \mathbb{R}^{d \times m \times k} \)
- piecewise linear approximation to an arbitrary convex function
- locally linear almost anywhere
- no sparse representation can be produced
- many learned parameters - best used in combination with dropout

![Figure 2: maxout activation function implementing/approximating different functions [Goodfellow et al., 2013]]
Training

- Train NMT model with bilingual corpus of pairs \((x^{(n)}, y^{(n)})\)

\[
\max_{\theta} \frac{1}{N} = \sum_{n=1}^{N} \log p_{\theta}(y^{(n)} | x^{(n)})
\]

- \(\theta\): set of trainable parameters

Model setup
- RNNLM as language model: set \(c_t = 0\)
- NMT as described above
- pre-train RNNLM and NMT separately
Integrating the language model - Fusions

**Figure 3:** Shallow and Deep Fusion [Gülçehre et al., 2015]
Shallow Fusion

- at each timestep $t$
  - NMT: compute score of possible next words for all hypotheses $\{y_{<t-1}^{(i)}\}$
  - sort new hypotheses (old hypotheses with new word added)
  - select top $K$ hypotheses as candidates $\{\hat{y}_{<t}^{(i)}\}_{i=1,...,K}$
  - rescore hypotheses with weighted sum of words (add score of new word)
    \[
    \log p(y_t = k) = \log p_{TM}(y_t = k) + \beta \log p_{LM}(y_t = k)
    \]
  - $\beta$: hyper-parameter
Deep Fusion

- integrate RNNLM and decoder of NMT by concatenating their hidden states
- fine-tune model to use both hidden states by tuning only output parameters
- new input of deep output layer

\[ p(y_t \mid y_{<t}, x) \propto \exp(y_t(W_o f_o(s_t^{TM}, s_t^{LM}, y_{t-1}, c_t) + b_o)) \]

- keeps learned monolingual structure of LM intact
Deep Fusion - Balancing LM and TM

- use controller mechanism to dynamically weight LM and TM models

\[ g_t = \sigma(v_g^T s_t^{LM} + b_g) \]

- multiply controller output with hidden state of LM to control magnitude of LM signal
- decoder only regards LM when deemed necessary
## Corpora

- **parallel datasets**

<table>
<thead>
<tr>
<th></th>
<th>Chinese</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Sentences</td>
<td>436K</td>
<td></td>
</tr>
<tr>
<td># of Unique Words</td>
<td>21K</td>
<td>150K</td>
</tr>
<tr>
<td># of Total Words</td>
<td>8.4M</td>
<td>5.9M</td>
</tr>
<tr>
<td>Avg. Length</td>
<td>19.3</td>
<td>13.5</td>
</tr>
</tbody>
</table>

(a) Zh-En

<table>
<thead>
<tr>
<th></th>
<th>Czech</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Sentences</td>
<td>12.1M</td>
<td></td>
</tr>
<tr>
<td># of Unique Words</td>
<td>1.5M</td>
<td>911K</td>
</tr>
<tr>
<td># of Total Words</td>
<td>151M</td>
<td>172M</td>
</tr>
<tr>
<td>Avg. Length</td>
<td>12.5</td>
<td>14.2</td>
</tr>
</tbody>
</table>

(c) Cs-En

<table>
<thead>
<tr>
<th></th>
<th>Turkish</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Sentences</td>
<td>160K</td>
<td></td>
</tr>
<tr>
<td># of Unique Words</td>
<td>96K*</td>
<td>95K</td>
</tr>
<tr>
<td># of Total Words</td>
<td>11.4M*</td>
<td>8.1M</td>
</tr>
<tr>
<td>Avg. Length</td>
<td>31.6</td>
<td>22.6</td>
</tr>
</tbody>
</table>

(b) Tr-En

<table>
<thead>
<tr>
<th></th>
<th>German</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Sentences</td>
<td>4.1M</td>
<td></td>
</tr>
<tr>
<td># of Unique Words</td>
<td>1.16M†</td>
<td>742K</td>
</tr>
<tr>
<td># of Total Words</td>
<td>11.4M†</td>
<td>8.1M</td>
</tr>
<tr>
<td>Avg. Length</td>
<td>24.2</td>
<td>25.1</td>
</tr>
</tbody>
</table>

(d) De-En

---

**Table 1:** datasets used [Gülçehre et al., 2015]

- **Zh-En:** training on SMS/CHAT, test on conversational speech
- **all others:** close domains

- **monolingual dataset:** English gigaword corpus
### Results - Translation Tasks

<table>
<thead>
<tr>
<th></th>
<th>Development Set</th>
<th></th>
<th>Test Set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>dev2010</td>
<td>t2010</td>
<td>tst2011</td>
</tr>
<tr>
<td>Previous Best (Single)</td>
<td>15.33</td>
<td>17.14</td>
<td>18.77</td>
</tr>
<tr>
<td>Previous Best (Combination)</td>
<td>-</td>
<td>17.34</td>
<td>18.83</td>
</tr>
<tr>
<td>NMT</td>
<td>14.50</td>
<td>18.01</td>
<td>18.40</td>
</tr>
<tr>
<td>NMT+LM (Shallow)</td>
<td>14.44</td>
<td>17.99</td>
<td>18.48</td>
</tr>
<tr>
<td>NMT+LM (Deep)</td>
<td>15.69</td>
<td>19.34</td>
<td><strong>20.17</strong></td>
</tr>
</tbody>
</table>

(a) Tr-En for each set

<table>
<thead>
<tr>
<th></th>
<th>SMS/CHAT</th>
<th>CTS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Dev</td>
<td>Test</td>
</tr>
<tr>
<td>PB + CSLM</td>
<td>15.5</td>
<td>14.73</td>
</tr>
<tr>
<td>HPB + CSLM</td>
<td>15.93</td>
<td>15.8</td>
</tr>
<tr>
<td>NMT</td>
<td>17.32</td>
<td>17.36</td>
</tr>
<tr>
<td>Shallow</td>
<td>16.59</td>
<td>16.42</td>
</tr>
<tr>
<td>Deep</td>
<td><strong>17.58</strong></td>
<td><strong>17.64</strong></td>
</tr>
</tbody>
</table>

(b) Zh-En, PB: phrase-based, HPB: hierarchical phrase based

(c) De-En and Cs-En translation tasks

Table 2: translation results on parallel corpora [Gülçehre et al., 2015]
## Results - RNNLM perplexity on development set

<table>
<thead>
<tr>
<th></th>
<th>Zh-En</th>
<th>Tr-En</th>
<th>De-En</th>
<th>Cs-En</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perplexity</td>
<td>223.68</td>
<td>163.73</td>
<td>78.20</td>
<td>78.20</td>
</tr>
<tr>
<td>Average $g$</td>
<td>0.23</td>
<td>0.12</td>
<td>0.28</td>
<td>0.31</td>
</tr>
<tr>
<td>Std Dev $g$</td>
<td>0.0009</td>
<td>0.02</td>
<td>0.003</td>
<td>0.008</td>
</tr>
</tbody>
</table>

Table 3: Perplexity of RNNLM on dev. set and $g$ [Gülçehre et al., 2015]
Conclusions

- **Fusion**
  - generally improves performance, regardless of the size of the parallel corpora
  - Deep Fusion achieves bigger improvements than Shallow Fusion
  - Deep Fusion makes a model incorporate the LM information as needed

- **domain similarity**
  - divergence in domains generally hurts model performance
  - LM is most useful when domains of parallel and monolingual corpora are close

- **controller mechanism**
  - makes model more robust to domain mismatch
  - most active on De-En/Cs-En as LM was able to model the target sentences best here
Motivation - [Chorowski and Jaitly, 2016]

- DNN Automatic Speech Recognition (ASR) systems suffer from overconfidence
  - peaked probability distribution prevents model from finding sensible alternatives
  - harms learning deep layers as gradient approximates 0 on correct characters: $[y_i = c] - p(y_i | y_{<i}, x)$
  - integrating LM to combat this results in incomplete transcripts
- idea: better integrate language models
Figure 4: Listen, Attend and Spell model [Chan et al., 2015]
Model

- encodes into a space-delimited sequence of characters directly using the encoder - decoder structure
- Listener: pyramid BLSTM (pBLSTM)
  
  \[ h^j_t = pBLSTM(h^j_{t-1}, [h^{j-1}_{2t}, h^{j-1}_{2i+1}]) \]

- Speller
  - attention-based LSTM transducer

  \[ c_i = AttentionContext(s_t, h) \]
  
  \[ s_t = RNN(s_{t-1}, y_{t-1}, c_{t-1}) \]

  \[ P(y_t \mid x, y_{<t}) = CharacterDistribution(s_t, c_t) \]

- RNN: 2-layer LSTM
- CharacterDistribution: MLP with softmax
- beam search to find character sequence \( y^* \)
Model - Attention Context for decoder timestep $t$

\[ c_t = \sum_u \alpha_{t,u} h_u \]
\[ \alpha_{t,u} = \frac{\exp(e_{t,u})}{\sum_u \exp(e_{t,u})} \]
\[ e_{t,u} = w^T \tanh(Ws_{t-1} + Vh_j + Uf_{t,j} + b) \]
\[ f_t = F \ast \alpha_{t-1} \]

- $\alpha$ very sharp probability distribution (attention)
- $e_{t,u}$: location aware scoring mechanism [Chorowski et al., 2015]
  - employs convolutional filters over the previous attention weights
  - extract $k$ $f_{t,j} \in \mathbb{R}^k$ vectors with $F \in \mathbb{R}^{k \times r}$ for every position $j$ of $\alpha_{t-1}$
Training

- minimize cross-entropy between ground-truth characters and model predictions
- criterion

\[
\text{loss}(y, x) = - \sum_i \sum_c T(y_i, c) \log(p)(y_t | y_{<t}, x)
\]

- \(T(y_i, c)\): target-label function, implemented differently for each model
Language Model Integration - based on [Gülçehre et al., 2015] Shallow Fusion

- extends beam search with a language modelling term

\[ y^* = \arg \min_y -\log(p(y \mid x)) - \lambda \log(p_{LM}(g)) - \gamma \text{coverage} \]

- \( \lambda, \gamma \): tunable parameters
- coverage: term promoting longer transcript to avoid incomplete transcripts
- overconfidence drastically influences \(-\log(p(y \mid x))\) if deviation from network’s best guess
- using a non-heuristic measurements does not change results, yet is far more difficult to compute
Experimental Setup

- **dataset**
  - Wall Street Journal dataset
  - extracted and augmented 80-dimensional mel-scale filterbanks
  - vocabulary consists of lower case letters, space, apostrophe, noise marker, SOS, EOS

- **LM**
  - extended-vocabulary trigram LM constructed with Kaldi [Povey et al., 2011] and spelling lexicon [Miao et al., 2015]

- **baseline**
  - Listener: 4 BLSTM, 3 time-pooling layers
  - Speller: LSTM, character embeddings, attention MLP
  - beam size: 10

- **final model**: baseline + LM
Improvement tests - Overconfidence

- problem
  - good prediction of speller results in little training signal through the attention mechanism to the listener
  - next predicted character may have low accuracy
  - beam search’s usefulness is greatly reduced

- Solution 1: Temperature Parameter $T$

$$p(y_i) = \frac{\exp(l_i/T)}{\sum_j \exp(l_j/T)}$$

- increase $T$: more uniform distribution; more deletion errors; better beam search results
- constrain emission of EOS
Improvement tests - Overconfidence: Label Smoothing

- variants of label smoothing
  - unigram label smoothing
  - neighborhood smoothing
- model is regularized
- higher entropy of network predictions
Improvement tests - Overconfidence

**Figure 5:** influence of temperature parameter on softmax
[Chorowski and Jaitly, 2016]
Improvement tests - Partial Transcripts Problem

- problem: using LM + beam search results in incomplete transcripts
- solution: extend beam search criterion with coverage term to promote long transcripts

\[
\text{coverage} = \sum_j \left[ \sum_i \alpha_{ij} > \tau \right]
\]

- prevents looping over the utterance

**Figure 6:** impact of techniques to promote long sequences [Chorowski and Jaitly, 2016]
Results

(a) baseline configuration and results

(b) trigram language model

Table 4: results on WSJ [Chorowski and Jaitly, 2016]

- competitive with other non-HMM techniques at the time
integrating LM into sequence-to-sequence models is competitive with other approaches, if done successfully
integrating a LM into a NN has proven to be difficult
LM is especially useful in cases the NN itself is uncertain
ablation study to see the actual effect of the LM on the results is advisable
Critique and Discussion
Neural machine translation by jointly learning to align and translate.
cite arxiv:1409.0473Comment: Accepted at ICLR 2015 as oral presentation.

Listen, attend and spell.
CoRR, abs/1508.01211.

Attention-based models for speech recognition.
CoRR, abs/1506.07503.
Towards better decoding and language model integration in sequence to sequence models.
*CoRR*, abs/1612.02695.

Maxout networks.

On using monolingual corpora in neural machine translation.
*CoRR*, abs/1503.03535.

On using very large target vocabulary for neural machine translation.
*CoRR*, abs/1412.2007.

EESEN: end-to-end speech recognition using deep RNN models and wfst-based decoding.

*CoRR*, abs/1507.08240.


The kaldi speech recognition toolkit.

*In In IEEE 2011 workshop.*