Sequence to Sequence Learning: CNNs, Training and Uncertainty

Michael Auli

with Sergey Edunov, Myle Ott, Jonas Gehring, Angela Fan, Denis Yarats, Yann Dauphin, David Grangier, Marc’Aurelio Ranzato

http://github.com/facebookresearch/fairseq-py
Overview

- Sequence to Sequence Learning & NLP
- Architecture: Sequence to Sequence Learning with CNNs.
- Exposure bias/Loss Mismatch: Training at the Sequence Level.
- Analyzing Uncertainty: model fitting and effects on search.
Sequence to Sequence Learning & Natural Language Processing
Sequence to Sequence Learning

The idea is to use one LSTM to read the input sequence, one time step at a time, to obtain large fixed-dimensional vector representation, and then to use another LSTM to extract the output sequence from that vector (fig. 1). The second LSTM is essentially a recurrent neural network language model except that it is conditioned on the input sequence. The LSTM's ability to successfully learn on data with long range temporal dependencies makes it an a natural choice for this application due to the considerable time lag between the inputs and their corresponding outputs (fig. 1).

There have been a number of related attempts to address the general sequence to sequence learning problem with neural networks. Our approach is closely related to Kalchbrenner and Blunsom [18] who were the first to map the entire input sentence to vector, and is very similar to Cho et al. [5]. Graves [10] introduced a novel differentiable attention mechanism that allows neural networks to focus on different parts of their input, and an elegant variant of this idea was successfully applied to machine translation by Bahdanau et al. [2]. The Connectionist Sequence Classification is another popular technique for mapping sequences to sequences with neural networks, although it assumes a monotonic alignment between the inputs and the outputs [11].

The main result of this work is the following. On the WMT'14 English to French translation task, we obtained a BLEU score of 34.81 by directly extracting translations from an ensemble of 5 de LSTMs (with 380M parameters each) using a simple left-to-right beam-search decoder. This is by far the best result achieved by direct translation with large neural networks. For comparison, the BLEU score of a SMT baseline on this dataset is 33.30 [29].

The 34.81 BLEU score was achieved by an LSTM with a vocabulary of 80k words, so the scores penalize whenever the reference translation contained a word not covered by these 80k. This result shows that a relatively unoptimized neural network architecture which has much room for improvement outperforms a mature phrase-based SMT system.

Finally, we used the LSTM to rescore the publicly available 1000-best lists of the SMT baseline on the same task [29]. By doing so, we obtained a BLEU score of 36.5, which improves the baseline by 3.2 BLEU points and is close to the previous state-of-the-art (which is 37.0 [9]).

Surprisingly, the LSTM did not suffer on very long sentences, despite recent experience of other researchers with related architectures [26]. We were able to do so because we reversed the order of words in the source sentence but not the target sentences in the training and test set. By doing so, we introduced many short term dependencies that made the optimization problem much simpler (see sec. 2 and 3.3). As a result, SGD could learn LSTMs that had no trouble with long sentences. The simple trick of reversing the words in these source sentences is one of the key technical contributions of this work.

- **Encode** source sequence, and **decode** target sequence with **RNNs** (Sutskever et al., 2014)
- **Attention:** choose relevant encoder states (Bahdanau et al., 2014)

---

**Figure 1:** Our model reads an input sentence “ABC” and produces “WXYZ” as the output sentence. The model stops making predictions after outputting the end-of-sentence token. Note that the LSTM reads the input sentence in reverse, because doing so introduces many short term dependencies in the data that make the optimization problem much easier.

---

Figure from: Sutskever et al., 2014, ”Sequence to Sequence Learning with Neural Networks”
Sequence to Sequence Learning

- Applications: translation, summarization, parsing, dialogue, ...

- "... basis for 25% of papers at ACL."
  Mirella Lapata at ACL’17 keynote
Sequence to Sequence Learning

Recurrent Continuous Translation Models

Nal Kalchbrenner  Phil Blunsom
Department of Computer Science
University of Oxford
{nal.kalchbrenner,phil.blunsom}@cs.ox.ac.uk

Joint Language and Translation Modeling with Recurrent Neural Networks

Michael Auli, Michel Galley, Chris Quirk, Geoffrey Zweig
Microsoft Research
Redmond, WA, USA
{michael.auli,mgalley,chrisq,gzweig}@microsoft.com

Neural Machine Translation by Jointly Learning to Align and Translate

Dmitry Bahdanau
 Jacobs University Bremen, Germany
KyoungHyun Cho Yoshua Bengio* 
Université de Montréal

Sequence to Sequence Learning with Neural Networks

Ilya Sutskever
Google
ilyas@google.com

Oriol Vinyals
Google
vinyals@google.com

Quoc V. Le
Google
quoc@google.com
Sequence to Sequence Learning

Deep Recurrent Models with Fast-Forward Connections for Neural Machine Translation

Jie Zhou, Ying Cao, Xuguang Wang, Peng Li, Wei Xu
Baidu Research - Institute of Deep Learning
Baidu Inc., Beijing, China
{zhoujie01, caoying03, wangxuguang, lipeng17, wei.xu}@baidu.com

Convolutional Sequence to Sequence Learning

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, Yann N. Dauphin

Attention Is All You Need

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit
Google Brain
avaswani@google.com, nshazeer@google.com, niparmar@google.com, jaksz@google.com

Noam Shazeer, Ashish Vaswani, Noam Shazeer, Jakob Uszkoreit
Google Brain
nshazeer@google.com, avaswani@google.com, niparmar@google.com, jaksz@google.com

Llion Jones, Aidan N. Gomez
Google Research
llion@google.com, aidan@gcs.utoronto.edu

Aidan N. Gomez
University of Toronto
aidan@gcs.utoronto.edu

Lukasz Kaiser
Google Brain
lukaszkaiser@google.com

Press Information – DeepL Translator Launch

Iliia Polonskhi
iliia.polonskhi@gmail.com
Architecture: Sequence to Sequence Learning with CNNs

Convolutional Sequence to Sequence Learning.
Jonas Gehring, Michael Auli, Yann Dauphin, David Grangier.
ICML 2017.
https://arxiv.org/abs/1711.04956
Convolutions vs Recurrent Networks

**CNN**
- 1d, 2d, 3d...
- vision
- convolutional filter

**RNN**
- 1d
- language, speech
- autoregressive filter
Convolutions vs Recurrent Networks

**CNN**
- 1d, 2d, 3d...
- vision
- convolutional filter
- bounded dependencies
- highly parallel

**RNN**
- 1d
- language, speech
- autoregressive filter
- unbounded dep. (theory)
- sequential
Recurrent Neural Network

The cat jumps far
Recurrent Neural Network

The cat jumps far.
Recurrent Neural Network

The cat jumps far

The cat jumps far
Recurrent Neural Network

The cat jumps far
Recurrent Neural Network

The cat jumps far.
Recurrent Neural Network

- O(T) sequential steps
- Recurrent connection causes vanishing gradient
- Are the recurrent connections necessary?
Convolutional Neural Network

- Time Delay Neural Network (Waibel et al., 1989)
- O(1) sequential steps
- Incrementally build context of context windows
- Builds **hierarchical** structure
Convolutional Neural Network

- Time Delay Neural Network (Waibel et al., 1989)
- \(O(1)\) sequential steps
- Incrementally build context of context windows
- Builds \textit{hierarchical} structure
Convolutional Neural Network

- Time Delay Neural Network (Waibel et al., 1989)
- \(O(1)\) sequential steps
- Incrementally build context of context windows
- Builds **hierarchical** structure
Gated Convolutional Neural Network

- Processes a sentence with a set of convolutions
- Each convolution learns higher level features
- Gates filter information to propagate up the hierarchy
Gated Convolutional Neural Network

- Processes a sentence with a set of convolutions
- Each convolution learns higher level features
- Gates filter information to propagate up the hierarchy
Gated Convolutional Neural Network

- Processes a sentence with a set of convolutions
- Each convolution learns higher level features
- Gates filter information to propagate up the hierarchy
Gated Convolutional Neural Network

- Processes a sentence with a set of convolutions
- Each convolution learns higher level features
- Gates filter information to propagate up the hierarchy
Gated Convolutional Neural Network

- Processes a sentence with a set of convolutions
- Each convolution learns higher level features
- Gates filter information to propagate up the hierarchy

\[ y = x \otimes \sigma(x') \]  \hspace{1cm} \text{(gated linear)}
Gated Linear Unit

- The gated linear unit can be seen as a multiplicative skip connection.
- We find this approach to gating improves performance.

Similar to ‘Swish’ (Ramachandran et al., 2017)
Gated Linear Unit

- The gated linear unit can be seen as a multiplicative skip connection
- We find this approach to gating improves performance

Similar to ‘Swish’ (Ramachandran et al., 2017)
Gated Linear Unit

- The gated linear unit can be seen as a multiplicative skip connection
- We find this approach to gating improves performance

Similar to ‘Swish’ (Ramachandran et al., 2017)
Gated Linear Unit

- The gated linear unit can be seen as a multiplicative skip connection
- We find this approach to gating improves performance

Similar to ‘Swish’ (Ramachandran et al., 2017)
Convolutional S2S: Encoder

- Input: word + position embeddings: 1, 2, 3, ...
- Weight Normalization (Salimans & Kingma, 2016)
- No batch or layer norm: initialization (He at al. ’15) and scale by sqrt(1/2)
- Repeat N times
Convolutional S2S: Decoder

- Input: word embeddings + position embeddings: 1, 2, 3, ...
- **Causal** convolution over generated sequence so far
- **Dot-product attention** at every layer
Convolutional S2S: Attention

The cat sat.
Convolutional S2S: Multi-hop Attention

- Attention in every decoder layer
- Queries contain information about previous source contexts
Convolutional S2S: Multi-hop Attention

- Attention in every decoder layer
- Queries contain information about previous source contexts
la maison de Léa
. la maison de Léa <end> .

Encoder
Encoder

. la maison de Léa <end> .
. la maison de Léa <end> .

Encoder

Decoder

. <start>
. la maison de Léa <end> .

Encoder

Decoder

. <start>
. la maison de Léa <end> .
. la maison de Léa <end> .

Encoder

Attention

Decoder

. <start>
la maison de Léa
. la maison de Léa <end> .

Encoder

Attention

Decoder
. la maison de Léa <end> .
. la maison de Léa <end> .

Encoder

Attention

Decoder

. <start> Léa
. la maison de Léa <end> .

Encoder

Attention

Decoder

. <start> Léa
. la maison de Léa <end> .

Encoder

Attention

Decoder

. <start> Léa
Léa la maison de Léa <end>
. la maison de Léa <end> .

Encoder

Attention

Decoder
Léa

. la maison de Léa <end>.

Encoder

Attention

Decoder

. <start> Léa
la maison de Léa
. la maison de Léa <end> .
. la maison de Léa <end> .

Encoder

Attention

Decoder

. <start> Léa's
La maison de Léa.
Léa's maison de Léa <end>
Léa's maison de Léa <end>.
. la maison de Léa <end> .

Encoder

Attention

Decoder

. <start> Léa 's
Léa's maison de Léa <end>.
Léa's house
## WMT’14 English-German Translation

<table>
<thead>
<tr>
<th>Method</th>
<th>Vocabulary</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN ByteNet (Kalchbrenner et al., 2016)</td>
<td>Characters</td>
<td>23.75</td>
</tr>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>Word 80k</td>
<td>23.12</td>
</tr>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>Word pieces</td>
<td>24.61</td>
</tr>
<tr>
<td>ConvS2S</td>
<td>BPE 40k</td>
<td>25.16</td>
</tr>
</tbody>
</table>

ConvS2S: 15 layers in encoder/decoder (10x512 units, 3x768 units, 2x2048)
Maximum context size: 27 words
### WMT’14 English-German Translation

<table>
<thead>
<tr>
<th>Model</th>
<th>Vocabulary</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN ByteNet (Kalchbrenner et al., 2016)</td>
<td>Characters</td>
<td>23.75</td>
</tr>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>Word 80k</td>
<td>23.12</td>
</tr>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>Word pieces</td>
<td>24.61</td>
</tr>
<tr>
<td>ConvS2S</td>
<td>BPE 40k</td>
<td>25.16</td>
</tr>
<tr>
<td>Transformer (Vaswani et al., 2017)</td>
<td>Word pieces</td>
<td>28.4</td>
</tr>
</tbody>
</table>

ConvS2S: 15 layers in encoder/decoder (10x512 units, 3x768 units, 2x2048)  
Maximum context size: 27 words

More work on non-RNN models!
<table>
<thead>
<tr>
<th>Model</th>
<th>Vocabulary</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>Word 80k</td>
<td>37.90</td>
</tr>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>Word pieces</td>
<td>38.95</td>
</tr>
<tr>
<td>RNN GNMT + RL (Wu et al., 2016)</td>
<td>Word pieces</td>
<td>39.92</td>
</tr>
</tbody>
</table>
## WMT’14 English-French Translation

<table>
<thead>
<tr>
<th>Model</th>
<th>Vocabulary</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>Word 80k</td>
<td>37.90</td>
</tr>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>Word pieces</td>
<td>38.95</td>
</tr>
<tr>
<td>RNN GNMT + RL (Wu et al., 2016)</td>
<td>Word pieces</td>
<td>39.92</td>
</tr>
<tr>
<td>ConvS2S</td>
<td>BPE 40k</td>
<td>40.51</td>
</tr>
<tr>
<td>Transformer (Vaswani et al., 2017)</td>
<td>Word pieces</td>
<td>41.0</td>
</tr>
</tbody>
</table>

ConvS2S: 15 layers in encoder/decoder (5x512 units, 4x768 units, 3x2048, 2x4096)
## Inference Speed on WMT’14 En-Fr

<table>
<thead>
<tr>
<th>Hardware</th>
<th>BLEU</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN GNMT (Wu et al., 2016) GPU (K80)</td>
<td>31.20</td>
<td>3028</td>
</tr>
<tr>
<td>RNN GNMT (Wu et al., 2016) CPU (88 cores)</td>
<td>31.20</td>
<td>1322</td>
</tr>
<tr>
<td>RNN GNMT (Wu et al., 2016) TPU</td>
<td>31.21</td>
<td>384</td>
</tr>
</tbody>
</table>

ntst1213 (6003 sentences)
## Inference Speed on WMT’14 En-Fr

<table>
<thead>
<tr>
<th>Model</th>
<th>Hardware</th>
<th>BLEU</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>GPU (K80)</td>
<td>31.20</td>
<td>3028</td>
</tr>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>CPU (88 cores)</td>
<td>31.20</td>
<td>1322</td>
</tr>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>TPU</td>
<td>31.21</td>
<td>384</td>
</tr>
<tr>
<td>ConvS2S, beam=5</td>
<td>GPU (K40)</td>
<td>34.10</td>
<td>587</td>
</tr>
<tr>
<td>ConvS2S, beam=1</td>
<td>GPU (K40)</td>
<td>33.45</td>
<td>327</td>
</tr>
</tbody>
</table>

ntst1213 (6003 sentences)
## Inference Speed on WMT’14 En-Fr

<table>
<thead>
<tr>
<th>Model</th>
<th>Hardware</th>
<th>BLEU</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>GPU (K80)</td>
<td>31.20</td>
<td>3028</td>
</tr>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>CPU (88 cores)</td>
<td>31.20</td>
<td>1322</td>
</tr>
<tr>
<td>RNN GNMT (Wu et al., 2016)</td>
<td>TPU</td>
<td>31.21</td>
<td>384</td>
</tr>
<tr>
<td>ConvS2S, beam=5</td>
<td>GPU (K40)</td>
<td>34.10</td>
<td>587</td>
</tr>
<tr>
<td>ConvS2S, beam=1</td>
<td>GPU (K40)</td>
<td>33.45</td>
<td>327</td>
</tr>
<tr>
<td>ConvS2S, beam=1</td>
<td>GPU (GTX-1080ti)</td>
<td>33.45</td>
<td>142</td>
</tr>
<tr>
<td>ConvS2S, beam=1</td>
<td>CPU (48 cores)</td>
<td>33.45</td>
<td>142</td>
</tr>
</tbody>
</table>

ntst1213 (6003 sentences)
Summary

- Alternative architecture for sequence to sequence learning
- Higher accuracy than models of similar size, despite fixed size context
- Faster generation (9x faster on lesser hardware)

Code & pre-trained models:
+ lua/torch:  http://github.com/facebookresearch/fairseq
+ PyTorch:  http://github.com/facebookresearch/fairseq-py
Exposure bias/Loss Mismatch: Training at the Sequence Level

*Classical Structured Prediction Losses for Sequence to Sequence Learning*
Sergey Edunov*, Myle Ott*, Michael Auli, David Grangier, Marc'Aurelio Ranzato
NAACL 2018
https://arxiv.org/abs/1711.04956

slide credit: Sergey Edunov, Marc’Aurelio Ranzato
Problems

- **Exposure bias**: training and testing are inconsistent. At training, the model has never observed its own predictions as input.
- Training criterion != Evaluation criterion
- Evaluation criterion is not differentiable
Selection of Recent Literature

● Reinforcement Learning-inspired methods
  ● MIXER (Ranzato et al., ICLR 2016)
  ● Actor-Critic (Bahdanau et al., ICLR 2017)
● Using beam search at training time:
  ● Beam search optimization (Wiseman et al. ACL 2016)
  ● Distillation based (Kim et al., EMNLP 2016)
Questions

1) How do classical structure prediction losses compare to recent methods?
2) Classical losses were often applied to log-linear models - how well do they work for neural nets?

Bottou et al. “Global training of document processing systems with graph transformer networks” CVPR 1997
Collins “Discriminative training methods for HMMs” EMNLP 2002
Taskar et al. “Max-margin Markov networks” NIPS 2003
Tsochantaridis et al. “Large margin methods for structured and interdependent output variables” JMLR 2005
Och “Minimum error rate training in statistical machine translation” ACL 2003
Smith and Eisner “Minimum risk annealing for training log-linear models” ACL 2006
Gimpel and Smith “Softmax-margin CRFs: training log-linear models with cost functions” ACL 2010
Notation

\[ \mathbf{x} = (x_1, \ldots, x_m) \]  
input sentence
Notation

\( x \)  \hspace{1cm} \text{input sentence}

\( t \)  \hspace{1cm} \text{target sentence}
Notation

\( x \)  input sentence

\( t \)  target sentence

\( u \)  hypothesis generated by the model
Notation

\( x \) \quad \text{input sentence}

\( t \) \quad \text{target sentence}

\( u \) \quad \text{hypothesis generated by the model}

\[ u^* = \arg \min_{u \in U(x)} \text{cost}(u, t) \quad \text{oracle hypothesis} \]
Notation

\( x \) input sentence

\( t \) target sentence

\( u \) hypothesis generated by the model

\( u^* \) oracle hypothesis

\( \hat{u} \) most likely hypothesis
Baseline: Token Level NLL

\[ \mathcal{L}_{\text{TokNLL}} = - \sum_{i=1}^{n} \log p(t_i | t_1, \ldots, t_{i-1}, x) \]

for one particular training example.

'Locally' normalized over vocabulary.
Sequence Level NLL

\[ \mathcal{L}_{\text{SeqNLL}} = - \log p(u^*|x) + \log \sum_{u \in \mathcal{U}(x)} p(u|x) \]

The sequence log-probability is simply the sum of the token-level log-probabilities.

'Globally' normalized over set of hypothesis \( \mathcal{U}(x) \)
Wir müssen unsere Einwanderungspolitik in Ordnung bringen.

Target:
We have to fix our immigration policy.

Beam:
BLEU  Model score  Target
75.0   -0.23   We need to fix our immigration policy.
100.0  -0.30   We have to fix our immigration policy.
36.9   -0.36   We need to fix our policy policy.
66.1   -0.42   We have to fix our policy policy.
66.1   -0.44   We've got to fix our immigration policy.
Wir müssen unsere Einwanderungspolitik in Ordnung bringen.

We have to fix our immigration policy.

Beam:

<table>
<thead>
<tr>
<th>BLEU</th>
<th>Model score</th>
<th>Translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>75.0</td>
<td>-0.23</td>
<td>We need to fix our immigration policy.</td>
</tr>
<tr>
<td>100.0</td>
<td>-0.30</td>
<td>We have to fix our immigration policy.</td>
</tr>
<tr>
<td>36.9</td>
<td>-0.36</td>
<td>We need to fix our policy policy.</td>
</tr>
<tr>
<td>66.1</td>
<td>-0.42</td>
<td>We have to fix our policy policy.</td>
</tr>
<tr>
<td>66.1</td>
<td>-0.44</td>
<td>We've got to fix our immigration policy.</td>
</tr>
</tbody>
</table>
Observations

• Important to use **oracle hypothesis** as surrogate target. Otherwise, the model learns to assign very **bad scores** to its hypotheses but is not trained to reach the target.

• Evaluation metric only used for **oracle selection** of target.

• Several ways to generate $U(x)$
Expected Risk

\[ \mathcal{L}_{\text{Risk}} = \sum_{u \in \mathcal{U}(x)} \text{cost}(t, u) \frac{p(u|x)}{\sum_{u' \in \mathcal{U}(x)} p(u'|x)} \]

• The cost is the evaluation metric; e.g.: 100-BLEU.

• REINFORCE is a special case of this (a single sample Monte Carlo estimate of the expectation over the whole hypothesis space).
Expected Risk

Source:
Wir müssen unsere Einwanderungspolitik in Ordnung bringen.

Target
We have to fix our immigration policy.

Beam:
BLEU Model score
75.0 -0.23 We need to fix our immigration policy.
100.0 -0.30 We have to fix our immigration policy.
36.9 -0.36 We need to fix our policy policy.
66.1 -0.42 We have to fix our policy policy.
66.1 -0.44 We've got to fix our immigration policy.

\( U(x) \) 

(expected BLEU=69)
Check out the paper for more examples of sequence level training losses!
Practical Tips

- Start from a model pre-trained at the token level. Training with search is excruciatingly slow...
- Even better if pre-trained model had label smoothing.
- Accuracy vs speed trade-off: offline/online generation of hypotheses.
- Mix token level NLL loss with sequence level loss to improve robustness.
## Results on IWSLT’14 De-En

<table>
<thead>
<tr>
<th>Method</th>
<th>TEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>TokNLL (Wiseman et al. 2016)</td>
<td>24.0</td>
</tr>
<tr>
<td>BSO (Wiseman et al. 2016)</td>
<td>26.4</td>
</tr>
<tr>
<td>Actor-Critic (Bahdanau et al. 2016)</td>
<td>28.5</td>
</tr>
<tr>
<td>Phrase-based NMT (Huang et al. 2017)</td>
<td>29.2</td>
</tr>
</tbody>
</table>
Results on IWSLT’14 De-En

<table>
<thead>
<tr>
<th>Method</th>
<th>TEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>TokNLL (Wiseman et al. 2016)</td>
<td>24.0</td>
</tr>
<tr>
<td>BSO (Wiseman et al. 2016)</td>
<td>26.4</td>
</tr>
<tr>
<td>Actor-Critic (Bahdanau et al. 2016)</td>
<td>28.5</td>
</tr>
<tr>
<td>Phrase-based NMT (Huang et al. 2017)</td>
<td>29.2</td>
</tr>
<tr>
<td>our TokNLL</td>
<td>31.8</td>
</tr>
<tr>
<td>SeqNLL</td>
<td>32.7</td>
</tr>
<tr>
<td>Risk</td>
<td>32.8</td>
</tr>
<tr>
<td>Max-Margin</td>
<td>32.6</td>
</tr>
</tbody>
</table>
## Fair Comparison to BSO

<table>
<thead>
<tr>
<th></th>
<th>TEST</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TokNLL</strong></td>
<td>24.0</td>
</tr>
<tr>
<td>(Wiseman et al. 2016)</td>
<td></td>
</tr>
<tr>
<td><strong>BSO</strong></td>
<td>26.4</td>
</tr>
<tr>
<td>(Wiseman et al. 2016)</td>
<td></td>
</tr>
<tr>
<td><strong>Our re-implementation of their TokNLL</strong></td>
<td>23.9</td>
</tr>
<tr>
<td><strong>Risk on top of the above TokNLL</strong></td>
<td>26.7</td>
</tr>
</tbody>
</table>
Fair Comparison to BSO

<table>
<thead>
<tr>
<th></th>
<th>TEST</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TokNLL</strong></td>
<td>24.0</td>
</tr>
<tr>
<td>(Wiseman et al. 2016)</td>
<td></td>
</tr>
<tr>
<td><strong>BSO</strong></td>
<td>26.4</td>
</tr>
<tr>
<td>(Wiseman et al. 2016)</td>
<td></td>
</tr>
<tr>
<td><strong>Our re-implementation of their TokNLL</strong></td>
<td>23.9</td>
</tr>
<tr>
<td><strong>Risk on top of the above TokNLL</strong></td>
<td>26.7</td>
</tr>
</tbody>
</table>

Methods fare comparably once the baseline is the same…
Diminishing Returns

On WMT’14 En-Fr, TokNLL gets 40.6 while Risk gets 41.0.

The stronger the baseline, the less to be gained.
Summary

- Sequence level training does improve, but with **diminishing returns**. It’s computationally very expensive.
- Particular **method** to train at the sequence level **does not matter**.
- Important to use **pseudo reference** as opposed to real reference.
Analyzing Uncertainty: model fitting and effects on search

Why do larger beams perform worse?
Why is the model under-estimating rare words?

Analyzing uncertainty in neural machine translation
Myle Ott, Michael Auli, David Grangier, Marc'Aurelio Ranzato
in submission
Goal

BETTER UNDERSTANDING

rare word under-estimation
- artifact of beam search (argmax)?
- due to exposure bias?
- due to poor estimation?

performance degradation with wide beams
- due to heuristic nature of beam search?
- is the model poorly trained?

model fitting
- are NMT models calibrated?
- what do NMT models over/under-estimate?
Outline

- Data uncertainty
- Search
- Analyzing the model distribution
Data Uncertainty

Intrinsic

- there are many semantically equivalent translations of the same sentence. E.g.: style, skipping prepositions, choice of words, structural choices (active/passive), etc.

EXAMPLE
Source: The night before would be practically sleepless.

Target #1: La nuit qui précède pourrait s’avérer quasiment blanche.
Target #2: Il ne dormait pratiquement pas la nuit précédente.
Target #3: La nuit précédente allait être pratiquement sans sommeil.
Target #4: La nuit précédente, on n’a presque pas dormi.
Target #5: La veille, presque personne ne connaîtra le sommeil.
Data Uncertainty

Intrinsic

• there are many semantically equivalent translations of the same sentence. E.g.: style, skipping prepositions, choice of words, structural choices (active/passive), etc.
• under-specification. E.g.: gender, tense, number, etc.

EXAMPLE
Source: nice .
Target #1: chouette .
Target #2: belle .
Target #3: beau .
Data Uncertainty

Intrinsic

• there are many semantically equivalent translations of the same sentence. E.g.: style, skipping prepositions, choice of words, structural choices (active/passive), etc.
• under-specification. E.g.: gender, tense, number, etc.

Extrinsic

• noise in the data. E.g.: partial translation, copies of the source, etc.
Data Uncertainty

Intrinsic

- there are many semantically equivalent translations of the same sentence. E.g.: style, skipping prepositions, choice of words, structural choices (active/passive), etc.
- under-specification. E.g.: gender, tense, number, etc.

Extrinsic

- noise in the data. E.g.: partial translation, copies of the source, etc.

Example: on WMT between 1 and 2% of the training target sentences are copies of the source.
Data Uncertainty

Intrinsic

- there are many semantically equivalent translations of the same sentence. E.g.: style, skipping prepositions, choice of words, structural choices (active/passive), etc.
- under-specification. E.g.: gender, tense, number, etc.

Extrinsic

- noise in the data. E.g.: partial translation, copies of the source, etc.

HOW DOES THIS AFFECT NMT?

Example: on WMT between 1 and 2% of the training target sentences are copies of the source.
Outline

- Data uncertainty
- Search
- Analyzing the model distribution
Search

Find the most likely sequence according to the model:

\[ \arg \max_y p(y|x; \theta) \]

Preliminary questions:

• is beam search effective?
• is beam search efficient?
• are there better search strategies?
Search

- Convolutional NMT with attention
- 15 layers
- 716D embeddings
- ~250M parameters

Beam search is very effective; only 20% of the tokens with probability < 0.7 (despite exposure bias)!
Increasing the beam width does not increase BLEU, while probability increases.
Increasing the beam width does not increase BLEU, while probability increases.

Sampling can find hypotheses with similar logprob but:
Increasing the beam width does not increase BLEU, while probability increases.

Sampling can find hypotheses with similar logprob but:
  - lower BLEU
  - it's more than 10x more inefficient
• Increasing the beam width does not increase BLEU, while probability increases.

• Sampling can find hypotheses with similar logprob but:
  • lower BLEU
  • it’s 20x less inefficient
Increasing the beam width does not increase BLEU, while probability increases.

Sampling can find hypotheses with similar logprob but...

Among the generated hypotheses, there exist at least one that is pretty close to the reference.
Beam search is very effective and efficient. However, large beams yield worse BLEU!
Beam 200/sampling 10K cover only about 22% of the total probability mass. Where is the rest?
Model distribution has a lot of uncertainty.
Puzzling Observations

- Increasing beam size hurts performance in terms of BLEU.
- Large beam accounts only for fraction of total probability mass.
Scatter Plot of Samples
The first nine episodes of Sheriff [unk]'s Wild West will be available from November 24 on the site [unk] or via its application for mobile phones and tablets.

Les neuf premiers épisodes de [unk] [unk] s Wild West seront disponibles à partir du 24 novembre sur le site [unk] ou via son application pour téléphones et tablettes.

The first nine episodes of Sheriff [unk] s Wild West will be available from November 24 on the site [unk] or via its application for mobile phones and tablets.
The first nine episodes of Sheriff [unk]'s Wild West will be available from November 24 on the site [unk] or via its application for mobile phones and tablets.

Les neuf premiers épisodes de [unk] [unk] s Wild West seront disponibles à partir du 24 novembre sur le site [unk] ou via son application pour téléphones et tablettes.

Model generates copies of source sentence! Why does beam find this?

High-logp low BLEU sample:
The first nine episodes of Sheriff [unk]'s Wild West will be available from November 24 on the site [unk] or via its application for mobile phones and tablets.
Uncertainty $\leftrightarrow$ Search

- Hard to characterize how uncertainty affects search in general.
- We can however simulate (extrinsic) uncertainty:
  - add fraction of “copy noise” and check effects on search.
Uncertainty $\leftrightarrow$ Search

Large beams are more prone to copy the source, hence the lower BLEU.
Uncertainty \(\Leftrightarrow\) Search

- **Source**: The first nine episodes of Sheriff <unk> 's Wild West will be available from November 24 on the site <unk> or via its application for mobile phones and tablets.

- **Target (reference)**: Les neuf premiers épisodes de <unk> <unk> s Wild West seront disponibles à partir du 24 novembre sur le site <unk> ou via son application pour téléphones et tablettes.

- **Sample**: The first nine episodes of Sheriff <unk> s Wild West will be available from November 24 on the site <unk> or via its application for mobile <unk> and tablets.
Uncertainty $\leftrightarrow$ Search

**Source:** The first nine episodes of Sheriff <unk>'s Wild West will be available from November 24 on the site <unk> or via its application for mobile phones and tablets.

**Target (reference):** Les neuf premiers épisodes de <unk> <unk> s Wild West seront disponibles à partir du 24 novembre sur le site

**Sample:** The first nine episodes of Sheriff <unk> s Wild West will be available from November 24 on the site <unk> or via its application for mobile <unk> and tablets.

**Inductive bias:**
NMT + attention has easy time to learn how to copy!
Uncertainty $\leftrightarrow$ Search

Initial tokens pay big penalty, but afterwards copying the source is cheap. Only large beams can discover this.
Uncertainty \textless{}\textless{} Search

- On WMT’14 En-Fr: \(~2\% of the training target sentences are copies of the corresponding source.

- Beam@1 yields copies 2.6\% of the time.
- Beam@20 yields copies 3.5\% of the time.
Fixing Search

- **Filtering the data** with model trained on “clean data” to remove copies from training set.
- **Constrain beam search** not to output too many words from the source sentence.
Fixing Search

![Graph showing BLEU scores for different beam sizes and conditions](image-url)

- **original**
- **filtered**
- **clean**
- **original (no copy)**
- **filtered (no copy)**
- **clean (no copy)**

The graph illustrates how BLEU scores change with varying beam sizes for different conditions.
Search & Uncertainty

- Search works very well, i.e. beam finds likely model hypotheses.
- However, it can find noisy sentences (model is wrong), that are merely due to noise in the data collection process.
- This explains why BLEU deteriorates for large beams.
- There are easy fixes.
Puzzling Observations

- Increasing beam size hurts performance in terms of BLEU.

- Large beam accounts only for fraction of total probability mass.
Outline

- Data uncertainty
- Search
- Analyzing the model distribution
Model Distribution

Check match between model and data distribution is challenging:

- For a given source sentence, we typically observe only one sample from the data distribution (the provided reference).
- Enumeration of all possible sequences using the data distribution is intractable anyway.

We would like to:

- check how closely model and data distribution match
- understand when they differ and why
Anecdotal Example

In the training set, some source sentences appear many times. Use corresponding targets to estimate the underlying data distribution!

**EXAMPLE**

Source: (The president cutoff the speaker).

Appears 798 times on the training set with 36 unique translations.

For this source sentence, model and data distribution match very well!
Analysis Tools

- Token level fitting
- Sentence level calibration
- Set level calibration
- Other necessary conditions
Model grossly under-estimate rare words.
Beam over-estimates frequent words, as expected.
Model grossly under-estimate rare words. Beam over-estimates frequent words, as expected.
Token Level: Matching Unigram Stats

WMT’17 En-De news-comm. portion

- ~300K parallel sentences
- 21 BLEU on test
- median freq. in 10% bin: 12

WMT’14 En-Fr

- ~35M parallel sentences
- 41 BLEU on test
- median freq. in 10% bin: 2500

More data & better model close the gap, but rare words are still under-estimated.
Token Level: Matching Unigram Stats

WMT’17 En-De news-comm. portion

- ~300K parallel sentences
- 21 BLEU on test
- Median freq. in 10% bin: 12

WMT’14 En-Fr

- ~35M parallel sentences
- 41 BLEU on test
- Median freq. in 10% bin: 2500

Match may look better than it is if model shifts probability mass within each of these buckets, let’s take a closer look then…
Sentence Level Calibration

Copy source sentences at a given rate during training, check whether probability assigned by the model to copies matches the copy production rate.

NMT model under-estimates copy probability at low rates, while it over-estimates it at high rates. Model spills probability mass on partial copies.
Set Level Calibration

$\mathbb{E}_{x \sim p_d} [\mathbb{I}\{x \in S\}] = p_m(S)$

where $S$ is the set of hypotheses produced by beam.

NMT model is very well calibrated at the set level.
Distance Matching

\[
\mathbb{E}_{y \sim p_d, y' \sim p_d} \left[ \text{BLEU}(y, y') \right] = \mathbb{E}_{y \sim p_m, y' \sim p_m} \left[ \text{BLEU}(y, y') \right]
\]

<table>
<thead>
<tr>
<th></th>
<th>En-Fr</th>
<th>En-De</th>
</tr>
</thead>
<tbody>
<tr>
<td>human</td>
<td>44.5</td>
<td>32.1</td>
</tr>
<tr>
<td>NMT</td>
<td>28.6</td>
<td>24.2</td>
</tr>
</tbody>
</table>

NMT model produces samples that have low BLEU and that are too diverse. Model spreads probability mass.
Distance Matching

$$\mathbb{E}_{y \sim p_d, y' \sim p_d} [BLEU(y, y')] \neq \mathbb{E}_{y \sim p_m, y' \sim p_m} [BLEU(y, y')]$$

<table>
<thead>
<tr>
<th></th>
<th>En-Fr</th>
<th>En-De</th>
</tr>
</thead>
<tbody>
<tr>
<td>human</td>
<td>44.5</td>
<td>32.1</td>
</tr>
<tr>
<td>NMT</td>
<td>28.6</td>
<td>24.2</td>
</tr>
</tbody>
</table>

NMT model produces samples that have low BLEU and that are too diverse. Model spreads probability mass.
## Multi-Reference Experiments

<table>
<thead>
<tr>
<th></th>
<th>Beam@5</th>
<th>Beam@200</th>
<th>200 Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>single reference</td>
<td>41.4</td>
<td>36.2</td>
<td>38.2</td>
</tr>
<tr>
<td>oracle reference</td>
<td>BLEU with reference yielding the largest BLEU score</td>
<td></td>
<td></td>
</tr>
<tr>
<td>average oracle</td>
<td>average BLEU over all hypothesis of beam/sampling - with closest ref</td>
<td></td>
<td></td>
</tr>
<tr>
<td>coverage</td>
<td>number of unique references using in matching</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We collected 10 additional references for 500 randomly selected source sentences from the test set.
# Multi-Reference Experiments

<table>
<thead>
<tr>
<th></th>
<th>Beam@5</th>
<th>Beam@200</th>
<th>200 Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>single reference</td>
<td>41.4</td>
<td>36.2</td>
<td>38.2</td>
</tr>
<tr>
<td>oracle reference</td>
<td>70.2</td>
<td>61.0</td>
<td>64.1</td>
</tr>
<tr>
<td>average oracle</td>
<td>65.7</td>
<td>56.4</td>
<td>39.1</td>
</tr>
<tr>
<td>coverage</td>
<td>1.9</td>
<td>5.0</td>
<td>7.4</td>
</tr>
</tbody>
</table>
Multi-Reference Experiments

<table>
<thead>
<tr>
<th></th>
<th>Beam@5</th>
<th>Beam@200</th>
<th>200 Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>single reference</td>
<td>41.4</td>
<td>36.2</td>
<td>38.2</td>
</tr>
<tr>
<td>oracle reference</td>
<td>70.2</td>
<td>61.0</td>
<td>64.1</td>
</tr>
<tr>
<td>average oracle</td>
<td>65.7</td>
<td>56.4</td>
<td>39.1</td>
</tr>
<tr>
<td>coverage</td>
<td>1.9</td>
<td>5.0</td>
<td>7.4</td>
</tr>
</tbody>
</table>

Beam produces outputs close to an actual reference. Lower scoring hypotheses are not far from a reference. However, they often map to the same reference.
Multi-Reference Experiments

<table>
<thead>
<tr>
<th></th>
<th>Beam@5</th>
<th>Beam@200</th>
<th>200 Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>single reference</td>
<td>41.4</td>
<td>36.2</td>
<td>38.2</td>
</tr>
<tr>
<td>oracle reference</td>
<td>70.2</td>
<td>61.0</td>
<td>64.1</td>
</tr>
<tr>
<td>average oracle</td>
<td>65.7</td>
<td>56.4</td>
<td>39.1</td>
</tr>
<tr>
<td>coverage</td>
<td>1.9</td>
<td>5.0</td>
<td>7.4</td>
</tr>
</tbody>
</table>

Sampling is more diverse but several samples poorly match any given reference. Mass is spread too much.
Conclusions

• Uncertainty in data: intrinsic/extrinsic
• Search: works really well. For large beams, we find spurious modes, but we know how to fix it!
• Model & Data distribution: model is surprisingly well calibrated. In general, it spreads probability mass too much compared to the data distribution.
Collaborators

Sergey Edunov    Myle Ott    Jonas Gehring    Yann Dauphin    David Grangier    Marc’Aurelio Ranzato
Questions?

Come work with us!
Openings for internships, postdocs, research scientists