EBMs for Structured Prediction
A Case Study: Machine Translation

Marc’Aurelio Ranzato
Facebook AI Research - New York City
ranzato@fb.com
https://ranzato.github.io/
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
Machine Translation

- Case-study for sequence to sequence transduction.
- It works in practice and has lots of applications.
- Some challenges:
  - input and output are discrete sequences of variable length
  - alignment
  - large vocabulary, large hypothesis space, need to search
  - one-to-many mapping / uncertainty, metric
  - domain shift
  - some language pairs may have little parallel data
Machine Translation

- Case-study for sequence to sequence transduction.
- It works in practice and has lots of applications.
- Some challenges:
  - input and output are discrete sequences of variable length
  - alignment
  - large vocabulary, large hypothesis space, need to search
  - one-to-many mapping / uncertainty, metric
  - domain shift
  - some language pairs may have little parallel data
Neural Machine Translation
(in 3 slides)

Example:

ITA (source) : Il gatto si e’ seduto sul tappetino.

EN (target) : The cat sat on the mat.

Approach:

Have one RNN/CNN to encode the source sentence, and another RNN/CNN/MemNN to predict the target sentence. The target RNN learns to (soft) align via attention.

Neural machine translation by jointly learning to align and translate, Bahdanau et al. ICLR 2015
Y. LeCun’s diagram
1) Represent source

Source Encoder (RNN/CNN)

il gatto si e’ seduto sul tappetino
2) score each source word (attention)

Source Encoder (RNN/CNN) → dot product → softmax

Source: il gatto si è seduto sul tappetino
Target: the cat sat on the carpet

Source Encoder (RNN/CNN)

Attention score: 0.95
3) combine target hidden with source vector
Alignment is learnt implicitly.
NMT Training & Inference

**Training**: predict one target token at the time and minimize cross-entropy loss.

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

**Evaluation**: BLEU at inference time.
NMT Training & Inference

**Training**: predict one target token at the time and minimize cross-entropy loss.

**Inference**: find the most likely target sentence (approximately) using beam search.

\[ \hat{u} = \arg \min - \log p(u|x) \]
Beam Search

the
-1
-2
-4
a
-0.5
Beam Search

-1
the
-1
a
-0.5
-0.5
-2
-3
-1
kitty
-2
-3
-0.5
cat
Beam Search

The diagram shows a beam search process with the following sequence of words and associated scores:

- **The** (-1)
- **A** (-0.5)
- **Kitty** (-1)
- **Cat** (-0.5)
- **Sat** (-0.5)
- **Ate** (-0.25)

The scores are indicated at each node of the tree, with the root node having a score of -1, and each subsequent node having a score of -0.5 or -0.25.
Beam Search

```
the -1
 a -0.5

kitty -1

sat -0.5

ate -0.25

at -0.5

on -0.25

-2

-1
```

M. Ranzato
Beam Search

a

the

-0.5

-1

-0.5

kitty

sat

-0.5

-0.25

on

ate

-0.25

-0.5

-0.5

-1

-0.5

-0.25

at

M. Ranzato
NMT Training & Inference

**Training**: predict one target token at the time and minimize cross-entropy loss.

**Inference**: find the most likely target sentence (approximately) using beam search.

**Evaluation**: compute BLEU on hypothesis returned by the inference procedure

\[
p_n = \frac{\sum_{\text{generated sentences}} \sum_{\text{ngrams}} \text{Clip}(\text{Count}(\text{ngram matches}))}{\sum_{\text{generated sentences}} \sum_{\text{ngrams}} \text{Count}(\text{ngram})}
\]

\[
\text{BLEU} = \text{BP} \ e^{\sum_{n=1}^{N} \frac{1}{N} \log p_n}
\]

**BLEU**: a method for automatic evaluation of machine translation, Papineni et al. ACL 2002
Problems

• Model is asked to predict a single token at training time, but the whole sequence at test time.

• Exposure bias: training and testing are inconsistent because model has never observed its own predictions at training time.

• At training time, we optimize for a different loss.

• Evaluation criterion is not differentiable.
Selection of Recent Literature

• RL-inspired methods
  • MIXER \textsuperscript{Ranzato et al. ICLR 2016}
  • Actor-Critic \textsuperscript{Bahdanau et al. ICLR 2017}

• Using beam search at training time:
  • BSO \textsuperscript{Wiseman et al. ACL 2016}
  • Distillation based \textsuperscript{Kim et al. EMNLP 2016}
Question

How do classical structure prediction losses compare against these recent methods?

Classical losses were often applied to log-linear models and/or other problems than MT.

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
Question

How do classical structure prediction losses compare against these recent methods?

Classical losses were often applied to log-linear models and/or other problems than MT.

Can the Energy-Based Model framework help unifying these different approaches?

During training

\[
\frac{\partial L}{\partial \theta} = \frac{\partial E}{\partial \theta}(x,y^+) - \frac{\partial E}{\partial \theta}(x,y^-)
\]

Some losses have explicit negative term, others replace it with constraints in the loss or in the architecture.

Energy-Based Learning

Energy-Based Learning

After training

Energy

target=prediction

space of possible predictions

Challenges

Key questions if we want to extend EBMs to MT:
• how to search for most likely output? Enumeration & exact search are intractable.
Key questions if we want to extend EBMs to MT:
• how to search for most likely output? Enumeration & exact search are intractable.
• how to deal with uncertainty? What if we only observe one minimum among many?

Challenges

EXAMPLE
Source: nice.

Target #1: chouette.
Target #2: belle.
Target #3: beau.

Key questions if we want to extend EBMs to MT:
• how to search for most likely output? Enumeration & exact search are intractable.
• how to deal with uncertainty? What if we only observe one minimum among many?

Challenges

Key questions if we want to extend EBMs to MT:
- how to search for most likely output? Enumeration & exact search are intractable.
- how to deal with uncertainty? What if we only observe one minimum among many?
- what if target is not reachable? E.g.: Not reachable = no hyp. in the beam is close to the reference.

Notation

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

\[ x \] input sentence

\[ t \] target sentence
Notation

\(x\) input sentence
\(t\) target sentence
\(u\) hypothesis generated by the model
Notation

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

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

\( u \) \hspace{1cm} \text{hypothesis generated by the model}

\[ u^* = \arg \min_{u \in \mathcal{U}(x)} \text{cost}(u, t) \] \hspace{1cm} \text{oracle hypothesis}
Notation

\( x \) input sentence

\( t \) target sentence

\( u \) hypothesis generated by the model

\( u^* \) oracle hypothesis

\[ \hat{u} = \arg \min_{u \in \mathcal{U}(x)} - \log p(u|x) \] 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 and omitting dependence on model parameters.
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.
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.**

**Two key differences:** choice of target and hypothesis set.

**Homework:** compute gradients of loss w.r.t. inputs to token level softmaxes.
Sequence Level NLL

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

Energy

gradients

\[ \mathcal{U}(x) \text{ set of hypotheses reachable by the model} \]
Example

Source:
Wir müssen unsere Einwanderungspolitik in Ordnung bringen.

Target
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>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>
Example

Source:
Wir müssen unsere Einwanderungspolitik in Ordnung bringen.

Target
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>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 as opposed to golden target. Otherwise, the model learns to assign very bad scores to its own hypotheses but is not trained to reach the target.

- Evaluation metric only used for oracle selection of target.

- Several ways to generate $U(x)$: beam, sampling, …

- Similar to token level NLL but normalizing over (subset of) hypotheses. Hypothesis score: average token level log-probability.
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 [1] is a special case of this (a single sample Monte Carlo estimate of the expectation over the whole hypothesis space).

Homework: compute gradients of loss w.r.t. inputs to token level softmaxes.

[1] Sequence level training with RNNs, Ranzato et al. ICLR 2016
Example

Source:
Wir müssen unsere Einwanderungspolitik in Ordnung bringen.

Target
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>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>

(expected BLEU=42)
Example

$$\{ \mathcal{U}(x) \}$$ set of hypotheses reachable by the model
Max-Margin

\[ \mathcal{L}_{\text{MaxMargin}} = \max[0, m - (E(\hat{u}) - E(u^*))] \]

- Energy: (negative) un-normalized score (or log-odds).
- Margin: \( m = \text{cost}(t, \hat{u}) - \text{cost}(t, u^*) \)
- The cost is our evaluation metric; e.g.: 100-BLEU.
- Increase score of oracle hypothesis, while decreasing score of most likely hypothesis.

Homework: compute gradients of loss w.r.t. inputs to token level softmaxes.
Max-Margin

Source:
Wir müssen unsere Einwanderungspolitik in Ordnung bringen.

Target
We have to fix our immigration policy.

Beam:
<table>
<thead>
<tr>
<th>BLEU</th>
<th>Model score</th>
<th>Sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>66.1</td>
<td>-0.20</td>
<td>We have to fix our policy policy.</td>
</tr>
<tr>
<td>75.0</td>
<td>-0.23</td>
<td>We need 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>-0.44</td>
<td>We've got to fix our immigration policy.</td>
</tr>
</tbody>
</table>
Max-Margin

$\mathcal{U}(x)$ set of hypotheses reachable by the model

$\hat{u}$ hypothesis space

gradients
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.

• Cost rescaling.

• Mix token level NLL loss with sequence level loss to improve robustness.

• Need to regularize more.
## 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>31.7</td>
</tr>
<tr>
<td>SeqNLL</td>
<td>32.7</td>
</tr>
<tr>
<td>Risk</td>
<td><strong>32.9</strong></td>
</tr>
<tr>
<td>Max-Margin</td>
<td>32.6</td>
</tr>
</tbody>
</table>
Observations

• Sequence level training does improve evaluation metric (both on training and) on test set.

• There is not so much difference between the different variants of losses. Risk is just slightly better.

• In our implementation and using the same computational resources, sequence level training is 26x slower per update using online beam generation of 5 hypotheses.
Observations

• Sequence level training does improve evaluation metric (both on training and) on test set.

• There is not so much difference between the different variants of losses. Risk is just slightly better.

• In our implementation and using the same computational resources, sequence level training is 26x slower per update using online beam generation of 5 hypotheses.

• Hard comparison since each paper has a different baseline!
## Fair Comparison to BSO

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

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

These 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.
Large Models in MT

Beam search is very effective; only 20% of the tokens with probability < 0.7 (despite exposure bias)!
Large Models in MT

Very large NMT models make almost deterministic transitions. No much to be gained by sequence level training.
Conclusion

• Sequence level training does improve, but with diminishing returns. It’s computationally very expensive.

• If model has little uncertainty (because of the task and because of the model being well (over)fitted), then sequence level training does not help much.

• The particular method to train at the sequence level does not really matter.

• Sequence level training is more prone to overfitting.
EBMs & MT

• Nice unifying framework.

• Different losses apply different weights to the “pull-up” and “pull-down” gradients.

• Two key differences two usual EBM learning:
  
  • restrict set of hypotheses to those that are reachable, and
  
  • replace actual target by oracle hypothesis.
Questions?
Вопросы?
¿Preguntas?
THANK YOU

ranzato@fb.com