**Problem**

Structured output prediction: learning a mapping from inputs to complex multivariate outputs \((x \rightarrow y)\)

Given a dataset of input-output pairs, \(D = \{(x^i, y^i)\}_{i=1}^n\).

learn a conditional distribution \(p(y | x)\) consistent with \(D\).

- Image captioning
- Semantic segmentation
- Machine translation

As data sets change, people get bigger but plane seating has not radically changed.

- Speech recognition

**Model**

We use autoregressive sequence to sequence models with attention, but our approach is more generic.

\[ p(y | x) = \prod_{i=1}^{\tau} p(y_i | x, y_{<i}) \]

- At inference, beam search finds \(\hat{y}(x) \approx \argmax p(y | x) \).
- As the reward signal, BLEU score or negative edit distance measure the quality of the predictions: \(\sum_{i=1}^\tau \delta(y_i \neq \hat{y}_i)\)

**Related Work**

- [Sutskever et al., CVPR 16] Rethinking the Inception
- Label smoothing can be thought as a special case of our method

Some alternative methods all of which require either sampling or inference from the model during training.

- [S. Bengio et al., NIPS 15] Schedule sampling
- [Ranzato et al., ICML’16] Sequence level training
- [Wiseman & Rush, EMNLP’16] Beam search optimization

**Model**

We use autoregressive sequence to sequence models with attention, but our approach is more generic.

\[ p(y | x) = \prod_{i=1}^{\tau} p(y_i | x, y_{<i}) \]

- At inference, beam search finds \(\hat{y}(x) \approx \argmax p(y | x) \).
- As the reward signal, BLEU score or negative edit distance measure the quality of the predictions: \(\sum_{i=1}^\tau \delta(y_i \neq \hat{y}_i)\)

**Related Work**

- [Sutskever et al., CVPR 16] Rethinking the Inception
- Label smoothing can be thought as a special case of our method

Some alternative methods all of which require either sampling or inference from the model during training.

- [S. Bengio et al., NIPS 15] Schedule sampling
- [Ranzato et al., ICML’16] Sequence level training
- [Wiseman & Rush, EMNLP’16] Beam search optimization

**Model**

We use autoregressive sequence to sequence models with attention, but our approach is more generic.

\[ p(y | x) = \prod_{i=1}^{\tau} p(y_i | x, y_{<i}) \]

- At inference, beam search finds \(\hat{y}(x) \approx \argmax p(y | x) \).
- As the reward signal, BLEU score or negative edit distance measure the quality of the predictions: \(\sum_{i=1}^\tau \delta(y_i \neq \hat{y}_i)\)

**Related Work**

- [Sutskever et al., CVPR 16] Rethinking the Inception
- Label smoothing can be thought as a special case of our method

Some alternative methods all of which require either sampling or inference from the model during training.

- [S. Bengio et al., NIPS 15] Schedule sampling
- [Ranzato et al., ICML’16] Sequence level training
- [Wiseman & Rush, EMNLP’16] Beam search optimization

---

**RL**

Entropy regularized expected reward (with a regularizer \(\tau\))

\[ \hat{Q}(x; \tau) = \sum_{y \in Y} \mathbb{E}[\text{exp}(\tau \log p(y | x)) / \tau] \]

To optimize \(\hat{Q}\), one uses REINFORCE, e.g. [Runciano et al.]

The gradients are high variance. The training is slow.

One needs to bootstrap training from an ML trained model.

REINFORCE ignores direct supervision after initialization.

**Sampling from Exponentiated Payoff**

Stratified sampling: first select a particular reward value, and then sample an output with that reward value.

- If reward is negative Hamming distance, \(r(y, \hat{y}) = -D_H(y, \hat{y})\) one can draw exact samples from \(q_i(y | x)\)

\[ q_i(y | x) = \frac{1}{Z} \exp(\tau \log p(y | x) / \tau) \]

Optimal \(p(y | x)\):

**KL as a Bregman divergence**

\[ D_p(p \parallel q) = \sum_{q \in Y} p(q) \log \frac{p(q)}{q(q)} \]

**TIMIT Speech Recognition**

Phone error rates (PER) for different methods on TIMIT dev & test sets. Average (min, max) PER for 4 training runs:

<table>
<thead>
<tr>
<th>Method</th>
<th>Dev set</th>
<th>Test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>ML baseline</td>
<td>20.9% (20.9, 22.8)</td>
<td></td>
</tr>
<tr>
<td>RAML, (r = 0.1)</td>
<td>19.74 (19.23, 20.16)</td>
<td></td>
</tr>
<tr>
<td>RAML, (r = 0.1)</td>
<td>19.64 (19.23, 20.16)</td>
<td></td>
</tr>
<tr>
<td>RAML, (r = 0.7)</td>
<td>21.28 (20.5, 21.97)</td>
<td></td>
</tr>
<tr>
<td>RAML, (r = 0.7)</td>
<td>21.28 (20.5, 21.97)</td>
<td></td>
</tr>
<tr>
<td>RAML, (r = 0.00)</td>
<td>20.15 (19.44, 20.84)</td>
<td></td>
</tr>
<tr>
<td>RAML, (r = 0.00)</td>
<td>20.15 (19.44, 20.84)</td>
<td></td>
</tr>
<tr>
<td>RAML, (r = 0.90)</td>
<td>18.46 (17.86, 19.07)</td>
<td></td>
</tr>
<tr>
<td>RAML, (r = 1.00)</td>
<td>18.46 (17.86, 19.07)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Dev set</th>
<th>Test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>ML baseline</td>
<td>36.95 (36.95, 37.95)</td>
<td></td>
</tr>
<tr>
<td>RAML, (r = 0.75)</td>
<td>36.62 (36.62, 37.91)</td>
<td></td>
</tr>
<tr>
<td>RAML, (r = 0.80)</td>
<td>36.90 (36.90, 37.91)</td>
<td></td>
</tr>
<tr>
<td>RAML, (r = 0.85)</td>
<td>36.91 (37.23, 37.23)</td>
<td></td>
</tr>
<tr>
<td>RAML, (r = 0.90)</td>
<td>36.91 (37.23, 37.23)</td>
<td></td>
</tr>
<tr>
<td>RAML, (r = 0.95)</td>
<td>36.94 (36.94, 37.94)</td>
<td></td>
</tr>
</tbody>
</table>

The RAML approach with different \(r\) considerably improves upon the maximum likelihood baseline.

---

**Follow-up work: UREX**

- Is RAML applicable to RL with unknown reward landscapes?

Improving Policy Gradient by Exploring User-Appreciated Rewards. (arXiv:1611.09921)

The key idea is to sample from \(p(y | x)\) and perform importance correction given \(\exp(\tau \log p(y | x) / \tau)\).

---

**Dalle Schuurman**

Reward Augmented Maximum Likelihood (RAML) for Neural Structured Prediction

Mohammad Norouzi, Samy Bengio, Zhifeng Chen, Navdeep Jaitly, Mike Schuster, Yonghui Wu, Dale Schuurmans