Bridging the Gap between Training and Inference for Neural Machine Translation

Wen Zhang, Yang Feng, Fandong Meng, Di You, Qun Liu

Björn Bebensee

bebensee@bi.snu.ac.kr

Biointelligence Laboratory

November 14, 2019

Overview

A brief introduction to NMT

A brief introduction to NMT

A brief introduction to NMT

A brief introduction to NMT

Notation:

Source sequence $x = \{x_1, \ldots, x_{|x|}\}$

Word embeddings e_{x_i} for each $x_i \in x$

Observed translation $y^* = \{y^*_1, \ldots, y^*_{|y^*|}\}$

A brief introduction to NMT

Encoder: bidirectional Gated Recurrent Unit (GRU), obtain hidden states $h_i = [\ \overrightarrow{h_i}; \overleftarrow{h_i}\ ]$ where

$$\overrightarrow{h_i} = \mathbf{GRU}(e_{x_i}, \overrightarrow{h_{i-1}}) \tag{1}$$
$$\overleftarrow{h_i} = \mathbf{GRU}(e_{x_i}, \overleftarrow{h_{i+1}}) \tag{2}$$

A brief introduction to NMT

Attention: attention over source/target words:

$$r_{ij} = \mathbf{v}_a^T \tanh\left(\mathbf{W}_a s_{j-1} + \mathbf{U}_a h_i\right) \tag{3}$$

$$\alpha_{ij} = \frac{\exp\left(r_{ij}\right)}{\sum_{i'=1}^{|\mathbf{x}|} \exp\left(r_{i'j}\right)} \tag{4}$$

Yields "source context vector" c_j at the j-th time step as a weighted sum of all source annotations:

$$c_j = \sum_{i=1}^{|\mathbf{x}|} \alpha_{ij} h_i \tag{5}$$

A brief introduction to NMT

Decoder: another GRU, given the source context vector c_j "unrolls" the target hidden state s_j at time step j:

$$s_j = \mathbf{GRU}(e_{y^*_{j-1}}, s_{j-1}, c_j) \tag{6}$$

Gives probability distribution P_j over all words in the target vocabulary as follows:

$$t_j = g\left(e_{y^*_{j-1}}, c_j, s_j\right) \tag{7}$$

$$o_j = \mathbf{W}_o t_j \tag{8}$$

$$P_j = \mathrm{softmax}\left(o_j\right) \tag{9}$$

Motivation

NMT models are trained to predict the next word, given the previous context words.

Learn a distribution!

Motivation

However:

At training time: model uses ground truth words as context to predict the next word (context from data distribution)

At inference time: model uses own previous predictions as context (context from model distribution)

This gap is called *exposure bias*.

Motivation

Training typically uses cross-entropy loss, optimizes target sequence to fit ground truth sequence as closely as possible.

Problem: one sentence can have multiple possible translations.

reference:	We should comply with the rule.
cand1:	We should abide with the rule.
cand2:	We should abide by the law.
cand3:	We should abide by the rule.

Loss forces prediction back to ground truth (*overcorrection*).

Contributions

Introduce a method that “bridges the gap” between training and inference.

Improves the model’s ability to recover from overcorrection and overall performance.

Proposed method

Simple idea: Instead of just ground truth, feed the model either previous predicted words or ground truth with a certain probability.

Brings training conditions closer to inference time.

Proposed method

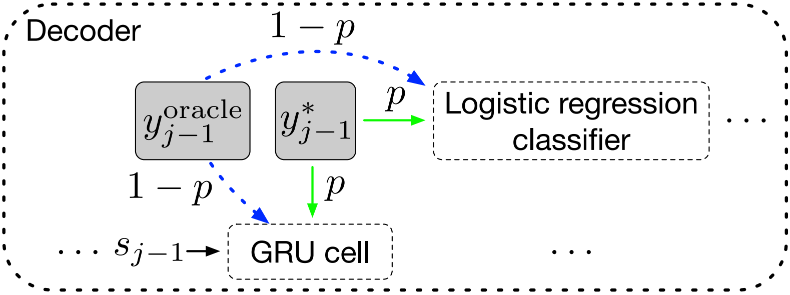


Figure: Sample between ground truth word and oracle word

Proposed method

To predict j-th target word y_j:

Recall:

$$s_j = \mathbf{GRU}(e_{y^*_{j-1}}, s_{j-1}, c_j)$$
$$P_j = \mathrm{softmax}\left(\mathbf{W}_o\ g\left(e_{y^*_{j-1}}, c_j, s_j\right)\right)$$

where s_j next hidden state and P_j probability distribution over target vocabulary

Proposed method

To predict j-th target word y_j:

1. Select an oracle word y_{j-1}^{oracle} at the $\{j-1\}$-th step.
2. Sample from the ground truth word y_{j-1}^{*} with a probability of p or from the oracle word y_{j-1}^{oracle} with a probability of $1-p$.
3. Use the sampled word as context $e_{y_{j-1}^{*}}$.

Oracle word selection

Two strategies to select the oracle words:

1. word-level oracle (greedy search)
2. sentence-level oracle (select an oracle sequence)

Word-level Oracle

Easiest way to pick an oracle word: pick the word with highest probability from the distribution P_{j-1}

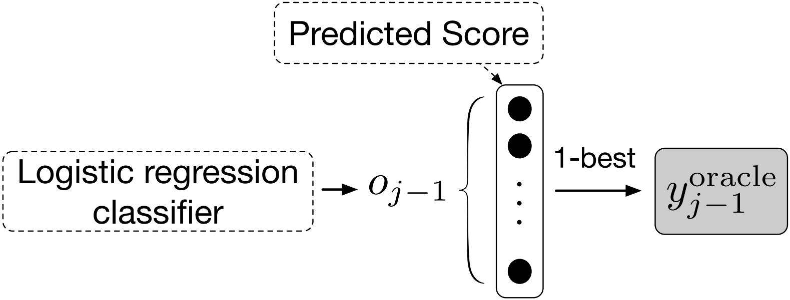


Figure: Word-level oracle without noise

Word-level Oracle

Better: Add noise to the the scores for a better sample

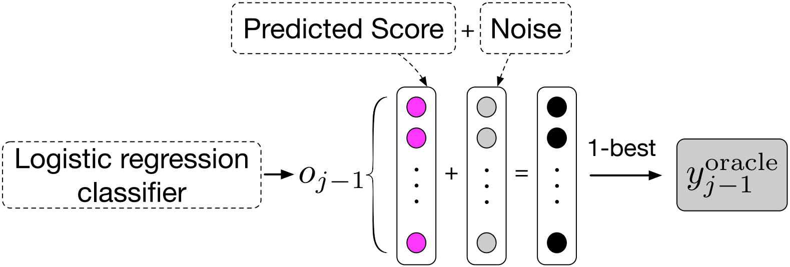


Figure: Word-level oracle with Gumbel noise

Word-level Oracle

Gumbel-Max technique:

Simple, efficient way to sample from categorical distribution

Given Gumbel noise η and a temperature τ, obtain

$$\tilde{o}_{j-1} = \left(o_{j-1} + \eta\right) / \tau \tag{10}$$

$$\tilde{P}_{j-1} = \text{softmax}(\tilde{o}_{j-1}) \tag{11}$$

and select the 1-best word from $\tilde{P}_{j-1}$.

Optimal temperature: $\tau = 0.5$ (found in experiments)

Sentence-level Oracle

Enlarge the search space: perform beam search, apply Gumbel noise at every word generation and get k-best candidate translations

Rank candidates according to some sentence-level metric (here: BLEU), best sentence is used as *oracle sentence*

Force Decoding

Problem

What if oracle sentence and ground truth do not have the same length?

Use **force decoding** to force oracle sentence length to be $|y^*|$ (length of ground truth). Modify beam search as follows:

If $j \leq |y^*|$ and top first word is `<EOS>`: select second word in $\tilde{P}_j$ for this candidate sentence

If `<EOS>` not top first word in $\tilde{P}_{|y^*|+1}$: select `<EOS>` as $\{|y^*| + 1\}$-th word for this candidate sentence

Sampling with decay

Convergence of the model depends on the choice of sampling probability p.

p too low: Sample from the ground-truth too often

p too high: Slow or no convergence

Sampling with decay

Let p decay as training progresses so we progressively sample more often from the model distribution and select oracle words.

Start with $p = 1$. Define it dependent on training epoch e:

$$p = \frac{\mu}{\mu + \exp(e/\mu)} \tag{12}$$

with hyperparameter μ.

Evaluation

Systems	Architecture	MT03	MT04	MT05	MT06	Average
Existing end-to-end NMT systems						
Tu et al. (2016)	Coverage	33.69	38.05	35.01	34.83	35.40
Shen et al. (2016)	MRT	37.41	39.87	37.45	36.80	37.88
Zhang et al. (2017)	Distortion	37.93	40.40	36.81	35.77	37.73
Our end-to-end NMT systems						
this work	RNNsearch	37.93	40.53	36.65	35.80	37.73
	+ SS-NMT	38.82	41.68	37.28	37.98	38.94
	+ MIXER	38.70	40.81	37.59	38.38	38.87
	+ OR-NMT	**40.40**‡†*	**42.63**‡†*	**38.87**‡†*	**38.44**‡	**40.09**
	Transformer	46.89	47.88	47.40	46.66	47.21
	+ word oracle	47.42	48.34	47.89	47.34	47.75
	+ sentence oracle	**48.31***	**49.40***	**48.72***	**48.45***	**48.72**

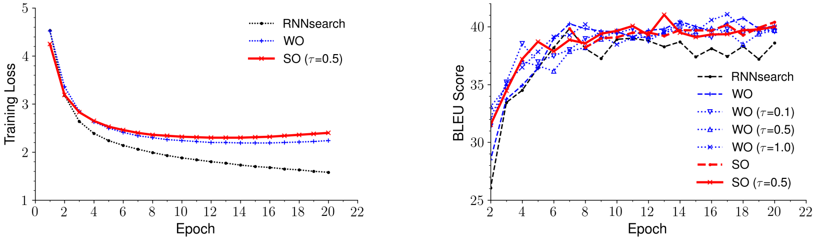
Figure: Case-insensitive BLEU scores (%) on Zh$\rightarrow$En translation task.

Factor analysis

Systems	Average
RNNsearch	37.73
+ word oracle	38.94
+ noise	39.50
+ sentence oracle	39.56
+ noise	**40.09**

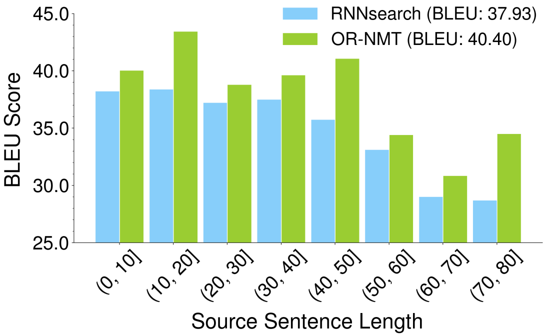
Figure: Factor analysis on Zh$\rightarrow$En translation, the results are average BLEU scores on MT03$\sim$06 datasets.

Convergence



Slower convergence but loss does not keep decreasing (intuition: WO and SO models help avoid overfitting)

Sentence length analysis



Cross-entropy loss requires predicted sequence to be same as ground truth, difficult for long sentences! Oracle helps recover from overcorrection

Conclusion

To conclude: Zhang et al. proposed a new training technique that helps to recover from overcorrection.

Outperforms previous methods as well as the transformer baseline.

Technique can be used with a variety of models.

Thank you for your attention.