Non-linear $n$-best list reranking with few features

Artem Sokolov, Guillaume Wisniewski, François Yvon

LIMSI-CNRS
Orsay, France

gposhta@gmail.com

Talk Plan

1 Motivation
2 Non-linear score functions
3 Features
4 Experiments
5 Conclusion & Future Work
Performance discrepancy in SMT

In SMT we are:

1. using linear scoring (max-entropy)
   \[ p(e|f) \propto e^{\lambda \cdot \bar{g}(e,f)} \]
2. MAP inference
   \[ e^* = \arg \max_e \lambda \cdot \bar{g}(e, f) \]
3. happy with 0.5-1 BLEU increase...

... until we search for oracle e (knowing reference):

<table>
<thead>
<tr>
<th>measure</th>
<th>found by decoder</th>
<th>lattice oracles</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLEU (fr2en)</td>
<td>( \sim 28 )</td>
<td>( \sim 50 )</td>
</tr>
<tr>
<td>BLEU (de2en)</td>
<td>( \sim 22 )</td>
<td>( \sim 38 )</td>
</tr>
<tr>
<td>BLEU (en2de)</td>
<td>( \sim 16 )</td>
<td>( \sim 30 )</td>
</tr>
</tbody>
</table>

potentially two-fold improvement
Performance discrepancy in SMT

1. search spaces contain excellent oracles:
   - \(~80\) BLEU, unrestricted [Wisniewski 10]
   - \(~50\) BLEU, restricted to lattices [Sokolov 12]

2. oracles not reachable even with advanced learning:
   - lattice MERT [Macherey 08, Kumar 09, Sokolov 11b]
   - exact MERT [Galley 11]
   - MIRA [Chiang 08]
   - tuning as ranking [Hopkins 11]

3. is scoring function main bottleneck?
   - poor and few features?
   - wrong models?

Motivation question
Can conventional SMT systems benefit from non-linear scoring?
Why aren’t linear functions sufficient?

1. convenient, but too simple
   - linear combination of 15-20 loosely related features
   - non-trivial measures’ approximation

2. theory: motivated by max-entropy principle
   - maximize entropy + known means of observables ⇔
   - ⇔ optimize likelihood + log-linear prob. distribution

3. practice: we optimize BLEU, not likelihood

4. features means not saved:

![Graph showing Moses n-best list reranking](image-url)
Related work

## Ranking in SMT
- tuning as ranking (PRO) [Hopkins 11]
- sampling (SampleRank) [Haddow 11]

## Boosting for better scoring functions
- attractive learning algorithm
- applied several times for SMT
- all attempts used boosting for classification (like AdaBoost)
  - whole MERT procedure as a weak learner [Duh 08, Xiao 10]
  - reweighting a separate feature in the log-linear model [Lagarda 08]
Non-linear scoring

$N$-best lists vs. Lattice

- dynamic programming doesn’t work in non-linear case
- cannot discard hypotheses early based on the partial score
  - partially fixable with monotone functions
- ⇒ use $n$-best lists

Which function family to use?

- $\bar{g}(e, f)$ – feature vector
- $\alpha_t$ – coefficients
- $h_t$ – ‘simple’ non-linear functions
- total score:

$$H(\bar{g}(e, f)) = \sum_{t=1}^{T} \alpha_t \cdot h_t(\bar{g}(e, f))$$
Non-linear score functions

**Tuning with Ranking**

### Why ranking?
- hypotheses naturally ordered under sentence-level BLEU
- deduce parameters comparing even mediocre or bad hypothesis
- earlier ranking approaches redefined losses, not scoring functions

### Pair-Wise Loss

\[
\mathcal{L}(H) = \sum_{f} \sum_{e_i, e_j \in \text{n-best}(f)} D(i, j) \left[ H(\bar{g}(e_i, f)) \geq H(\bar{g}(e_j, f)) \right]
\]

- \(b(e, r)\) – sentence level BLEU wrt. \(r\)
- \([A] = 1\) if the \(A = \text{true}\) and \(0\) otherwise
- the higher is \(D(i, j)\), the more important is pair \((e_i, e_j)\)
RankBoost

Given \((\bar{g}_1, y_1), \ldots, (\bar{g}_m, y_m) : g_i \in \mathcal{G}, y_i \in \mathcal{Y}\)

**Algorithm**

- Initialize \(D\) on \(\mathcal{G} \times \mathcal{G}\), to reflect “importance” of pairs.
- For \(t = 1, \ldots, T\):
  - Find weak learner \(h_t : \mathcal{G} \to \mathbb{R}\) using \(D_t\)
  - Choose \(\alpha_t \in \mathbb{R}\)
  - Update
    \[
    D_{t+1}(\bar{g}_0, \bar{g}_1) = \frac{D_t(\bar{g}_0, \bar{g}_1) \exp(\alpha_t (h_t(\bar{g}_0) - h_t(\bar{g}_1)))}{Z_t},
    \]
    where \(Z_t\) is chosen to make \(D_t\) a distribution.
- Output the final ranking: \(H(\bar{g}) = \sum_{t=1}^{T} \alpha_t h_t(\bar{g})\).

**Ranking loss**

\[
\mathcal{L}(H) \leq \prod_{t=1}^{T} Z_t.
\]
## Weak Functions

### One-dimensional learners

- **decision stumps**
  - simplest nonlinear
  - approximates complex curves

\[ h(\bar{g}(e, f); \theta, k) = [g_k > \theta] \]

- **linear weak learners to discriminate:**
  - learning with ranking
  - non-linearity

\[ h(\bar{g}(e, f); k) = g_k \]

- **piece-wise linear learners:**
  - non-linear, but captures linear dependencies easily
  - important, as beam-search pruning is linear

\[ h(\bar{g}(e, f); \theta, k) = g_k \cdot [g_k > \theta] \]
Decoder configurations

N-code decoder configurations:

- N-code decoder (based on biling. $n$-grams)
- **basic**: 11 features (found in any decoder), 100-best
  - language model
  - distortion + 2 reordering models,
  - translation model + 4 lexical translation weights
  - 2 penalties for words and phrases
- **extended**: 23 features (WMT’12 best system for fr↔en), 300-best
  1. +2 translation models, +2 lexicalized reordering models
  2. reoptimize with MERT (15 features)
  3. add neural-network models’ features:
     - +1 linear score (over 15 features)
     - +4 neural-network language models [Le 11]
     - +4 neural-network translation models [Le 12]
Feature Transformations

<table>
<thead>
<tr>
<th>qid:0</th>
<th>configuration</th>
<th>feature sets</th>
<th>#features</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: -108.63</td>
<td>linear</td>
<td>basic</td>
<td>11</td>
</tr>
<tr>
<td>2: -108.09</td>
<td>linear</td>
<td>extended</td>
<td>23</td>
</tr>
<tr>
<td>3: -116.04</td>
<td>non-linear</td>
<td>basic</td>
<td>32</td>
</tr>
<tr>
<td>4: -118.12</td>
<td>non-linear</td>
<td>scale &amp; rank</td>
<td>44</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>extended</td>
<td>68</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale &amp; rank</td>
<td>92</td>
</tr>
</tbody>
</table>

# at the end of trading, the Prague bascula award in the negative
Feature Transformations

0 qid:0
1: -108.63 1N:-8.59 1P:-8.59
2: -108.09 2N:-8.33 2P:-8.33
3: -116.04 3N:-8.57 3P:-8.57
4: -118.12 4N:-9.17 4P:-9.17
...

# at the end of trading, the Prague bascula award in the negative

<table>
<thead>
<tr>
<th>non-linear</th>
<th>configuration</th>
<th>feature sets</th>
<th>#features</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic</td>
<td>scale</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td></td>
<td>scale &amp; rank</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>extended</td>
<td>scale</td>
<td>68</td>
<td></td>
</tr>
<tr>
<td></td>
<td>scale &amp; rank</td>
<td>92</td>
<td></td>
</tr>
</tbody>
</table>

Normalization

- prob. features incomparable for different sentence lengths (e.g., LM score)
- divide by number of words (N) / phrases (P)
Feature Transformations

0 qid:0
1:  -108.63 1N:-8.59 1P:-8.59 1S:0.17 1NS:0.07 1PS:0.29
2:  -108.09 2N:-8.33 2P:-8.33 2S:0.60 2NS:0.08 2PS:0.33
3:  -116.04 3N:-8.57 3P:-8.57 3S:0.82 3NS:0.17 3PS:0.37
4:  -118.12 4N:-9.17 4P:-9.17 4S:0.29 4NS:0.15 4PS:0.31
...
# at the end of trading, the Prague bascula award in the negative

<table>
<thead>
<tr>
<th>configuration</th>
<th>feature sets</th>
<th>#features</th>
</tr>
</thead>
<tbody>
<tr>
<td>linear</td>
<td>basic</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>extended</td>
<td>23</td>
</tr>
<tr>
<td>non-linear</td>
<td>basic</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>scale</td>
<td></td>
</tr>
<tr>
<td></td>
<td>scale &amp; rank</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>extended</td>
<td>68</td>
</tr>
<tr>
<td></td>
<td>scale</td>
<td></td>
</tr>
<tr>
<td></td>
<td>scale &amp; rank</td>
<td>92</td>
</tr>
</tbody>
</table>

Normalization

- prob. features incomparable for different sentence lengths (e.g., LM score)
- divide by number of words (N) / phrases (P)

Scale-features

- regularize range among different phrases
- rescale to [0; 1] (S)
Feature Transformations

<table>
<thead>
<tr>
<th>qid:0</th>
<th>configuration</th>
<th>feature sets</th>
<th>#features</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>basic</td>
<td>linear</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>extended</td>
<td>extended</td>
<td>23</td>
</tr>
<tr>
<td>2</td>
<td>basic</td>
<td>non-linear</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>scale</td>
<td>basic</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>scale &amp; rank</td>
<td>extended</td>
<td>68</td>
</tr>
<tr>
<td></td>
<td>scale &amp; rank</td>
<td>extended</td>
<td>92</td>
</tr>
</tbody>
</table>

# at the end of trading, the Prague bascula award in the negative

Normalization

- prob. features incomparable for different sentence lengths (e.g., LM score)
- divide by number of words (N) / phrases (P)

Scale-features

- regularize range among different phrases
- rescale to \([0; 1]\) (S)

Rank-features

- sort according to feature
- take its rank (R)

used S, NS, PS & R features – the rest discarded
Datasets

- dev-set for MERT: WMT’09
- dev-set for RankBoost: WMT’09
  - training labels: sentence level BLEU
- test-sets:
  - WMT’10
  - WMT’11
  - WMT’12

MERT setup

- MERT is unstable $\Rightarrow$ 8 independent (re)runs, each with:
  - 20 init. points restarts
  - 30 random direction (additional to axes)
Learning reranking

Process
- run full MERT
- take last iteration’s \( n \)-best list
- transform features
- sample:
  - 2000 pairs
  - discard those less than 0.05 BLEU apart
- train RankBoost for some \( T \)

Tested variants
- 2 weighting schemes:
  - uniform \( D(i, j) = 1 \)
  - difference \( D(i, j) = \text{BLEU}(e_i, r) - \text{BLEU}(e_j, r) \)
- 3 weak learners:
  - stumps
  - linear
  - piece-wise linear
Weak learners, weighting schemes and features sets on WMT'10.

Weighting: uniform (uni), weighted with sentence-BLEU (wht).
Learners: stumps (stm/stm), linear (lin), piece-wise linear (pwl/pwl).
Features sets: scale- and scale-+rank-.

piece-wise linear + uniform weighting performs the best
Weak learner analysis

- piece-wise linear functions include linear as subclass
- possible that RankBoost still uses linear models

Selection phases of models/features

1. $T \lesssim 10$: score feature is selected (reusing MERT linear model)
2. $10 \lesssim T \lesssim 50$: more features, still linear model (but better BLEU)
3. $50 \lesssim T$: piece-wise linear models start to appear (improving BLEU)
4. $T \gtrsim M$: over-fitting

Per dataset performance peak analysis

- WMT’10 – “ranking” (2) phase
- WMT’11 – “linear” (1) or “non-linear” (3) phases
- WMT’12 – “non-linear” (3) phase
Experiments

Maximum relative gains

Figure: Maximum relative gains in BLEU on WMT’10.
WMT’10 Basic

(a) Basic conf., WMT’10, scale-features
(b) Basic conf., WMT’10, scale- & rank-features
WMT’10 Extended

(c) Ext. conf., WMT’10, scale-features

(d) Ext. conf., WMT’10, scale- & rank-features
**WMT’11 Extended**

(e) Ext. conf., WMT’11, scale-features

(f) Ext. conf., WMT’11, scale- & rank-features
WMT’12 Extended

(g) Ext. conf., WMT’12, scale-features

(h) Ext. conf., WMT’12, scale- & rank-features
Experiments

All WMT - curve comparison, extended feature set

WMT’10

WMT’11

WMT’12

Artem Sokolov (LIMSI)

Non-linear \( n \)-best list reranking

20 / 23
## Cross-results

<table>
<thead>
<tr>
<th>test \ valid</th>
<th>WMT'10</th>
<th>WMT'11</th>
<th>WMT'12</th>
<th>MERT mean</th>
<th>MERT interval</th>
<th>300-best oracle</th>
</tr>
</thead>
<tbody>
<tr>
<td>WMT'10</td>
<td>-</td>
<td>29.68±0.07</td>
<td>29.58±0.09</td>
<td>29.38±0.09</td>
<td>[29.26,29.54]</td>
<td>39.72</td>
</tr>
<tr>
<td>all scores</td>
<td>-</td>
<td>29.66</td>
<td>29.55</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>no scores</td>
<td>-</td>
<td>29.58</td>
<td>29.54</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WMT'11</td>
<td>30.42±0.07</td>
<td>-</td>
<td>30.41±0.05</td>
<td>30.16±0.11</td>
<td>[29.97,30.34]</td>
<td>41.11</td>
</tr>
<tr>
<td>all scores</td>
<td>30.55</td>
<td>-</td>
<td>30.46</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>no scores</td>
<td>30.26</td>
<td>-</td>
<td>30.35</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WMT'12</td>
<td>30.50±0.08</td>
<td>30.52±0.06</td>
<td>-</td>
<td>30.38±0.12</td>
<td>[30.19,30.62]</td>
<td>40.64</td>
</tr>
<tr>
<td>all scores</td>
<td>30.59</td>
<td>30.62</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>no scores</td>
<td>30.36</td>
<td>30.42</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Conclusion

- non-linear approach to reranking \( n \)-best lists
- approach potentially boosts performance by \(+0.4\) BLEU-points
- heterogenous validation/test corpora lessen gains
- stable gains on homogeneous corpora

Gains obtained attributable to:
- appropriateness of the ranking loss
- flexibility of non-linear modeling
Future Work

Learning non-linear functions on lattices

- MERT on $n$-best lists is $\sim$optimal [Duh 08, Galley 11]
- $n$-best lists rescoring was
  - proof-of-concept to avoid tight decoder integration
  - limited, as influence pruning and scoring partial hyps
  - restricted to forms like $h_t(\sum_i g_k) \Rightarrow$ triggers feature design problems
- lattice non-linear learning
  - boosting = functional gradient descend
  - large-margin framework
  - can derive non-linear functions in forms $\sum_i h_t(g_k)$
    (less normalizing/scaling problems)