Minimum Error Training of Log-Linear Translation Models

Mauro Cettolo & Marcello Federico

ITC-irst - Centro per la Ricerca Scientifica e Tecnologica
I-38050 Povo (Trento), Italy

{cettolo, federico}@itc.it
Overview

- Log-Linear Models for MT
- Minimum Error Training
- Simplex Algorithm
- Experimental Results
- Conclusions
Log-linear Models in ASR and SMT

Log-linear models were introduced in ASR by Philips labs in the late ’90s:

- log-linear interpolation of language models [Klakow, 1998]
- scaling factor estimation to minimize recognition errors [Beyerlein, 1997]

More recently, log-linear models have been introduced in SMT:

- maximum entropy models and discriminative training for SMT [Och & Ney, 2002]
- minimum error rate training in SMT [Och, 2003].

Our work is related to [Och, 2003], but investigates a different training technique.
Log-Linear SMT Models

Maximum Entropy Framework for SMT

Maximum Entropy framework for word-alignment MT approach:

\[ e^* = \arg \max_e \sum_a \Pr(e, a | f) \approx \arg \max_e \max_a \Pr(e, a | f) \tag{1} \]

\( \Pr(e, a | f) \) is determined through real valued feature functions \( h_i(e, f, a), i = 1 \ldots M \), and takes the parametric form:

\[ p_\lambda(e, a | f) = \frac{\exp\{\sum_i \lambda_i h_i(e, f, a)\}}{\sum_{e, a} \exp\{\sum_i \lambda_i h_i(e, f, a)\}} \tag{2} \]

Example: feature functions of IBM Model 4:

\[ h_1(e, f, a) = \log \Pr(e) \quad \text{(target language model)} \]
\[ h_2(e, f, a) = \log \Pr(\phi | e) \quad \text{(fertility model)} \]
\[ h_3(e, f, a) = \log \Pr(\tau | e, \phi) \quad \text{(lexicon model)} \]
\[ h_4(e, f, a) = \log \Pr(\pi | e, \phi, \tau) \quad \text{(distortion model)} \]
Log-Linear SMT Models

Search Criterion and Properties

The search criterion of MT can be rewritten as:

$$e^* = \arg \max_{e} \max_{a} \sum_{i} \lambda_i h_i(e, f, a)$$  \hspace{1cm} (3)

The ME framework gives the following advantages:

- directly models the posterior probability (discriminative model)
- does not rely on probability factorizations with independence assumptions
- its mathematically sound framework permits to add any kind of feature
- includes any IBM-model as special case, e.g. see previous slide with $\lambda$ set to 1
- ML or minimum error training can be applied to estimate free parameters ($\lambda$)
Training of Log-Linear Models

Instead of applying MLE, training can directly address performance optimization:

\[ \lambda^* = \arg \min_{\lambda} E_D(\lambda) \quad \text{(4)} \]

where \( E_D(\lambda) \):

- measures translation errors over a development set \( D \), e.g. Bleu, Nist, WER, PER
- can be very irregular, i.e. has many local minima

We apply a multi-variate minimization algorithm, called simplex, which:

- empirically evaluates \( E_D(\lambda) \) several times until convergence
- requires running the SMT search algorithm for each evaluation

The same approach was independently applied by [Zens & Ney, 2004]
Experiments

Experimental Setting

- Baseline system: phrase-based extension of IBM Model-4 (6 feature functions)
  - Beam-search decoder: threshold pruning, histogram pruning

- **Task 1:** Chinese-English NIST 2003 Large data condition
  - domain: news agencies
  - statistics: vocab: CH 148K, EN 110K; #words: CH 13.1M, EN 13.5M
  - develop/test: 877sp with 4 references/919sp with 4 references

- **Task 2:** Chinese-English C-STAR Eval 2003
  - domain: basic traveling expressions
  - statistics: vocab: CH 12K, EN 11K; #words: CH 434K, EN 450K
  - develop/test: 1,000sp with 1 reference/506sp with 16 references

- Translation Error Measures: BLEU/NIST scores

- Performance Measures: BLEU/NIST scores versus search complexity (#hyp)
Experiments

Results

Additional information:

- **Baseline uses uniform parameters**
- **Beam-search settings:**
  - loose threshold-pruning
  - tight histogram pruning
- **Minimum Error Training:**
  - with 12 CPUs - Xeon 2.4GHz
  - single iteration takes 7min
  - convergence in about 100 steps

<table>
<thead>
<tr>
<th>NIST 2003 TASK</th>
<th>Criterion</th>
<th>BLEU</th>
<th>NIST</th>
<th># hyp</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BLEU</td>
<td>0.1854</td>
<td>7.2882</td>
<td>116M</td>
</tr>
<tr>
<td></td>
<td>NIST</td>
<td>0.1840</td>
<td>7.3362</td>
<td>115M</td>
</tr>
<tr>
<td></td>
<td>Baseline</td>
<td>0.1803</td>
<td>7.2115</td>
<td>116M</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C-STAR 2003 TASK</th>
<th>Criterion</th>
<th>BLEU</th>
<th>NIST</th>
<th># hyp</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BLEU</td>
<td>0.4614</td>
<td>8.4945</td>
<td>14.9M</td>
</tr>
<tr>
<td></td>
<td>NIST</td>
<td>0.4581</td>
<td>8.4675</td>
<td>14.9M</td>
</tr>
<tr>
<td></td>
<td>Baseline</td>
<td>0.4208</td>
<td>8.3169</td>
<td>14.8M</td>
</tr>
</tbody>
</table>
Experiments

Performance after Parameter Training

BLEU score after applying threshold pruning.

M. Cettolo & M. Federico

IWSLT 2004

Kyoto, October 1, 2004
Experiments

Performance after Parameter Training

NIST score after applying threshold pruning.

M. Cettolo & M. Federico
IWSLT 2004
Kyoto, October 1, 2004
Conclusions & Future Work

- Small but consistent and stable score improvements
  - however, no subjective assessment has been made

- BLEU score optimization is more effective than NIST score optimization

- Simplex can also be used to tune other parameters
  - e.g. pruning parameters of the search-algorithm [Zens & Ney, 2004]

- Future work will investigate:
  - the use of n-best lists and re-ranking methods [Shen et al., 2004]
  - the joint optimization of ASR and SMT model parameters [Zhang et al. 2004]