Discriminative Training

Kevin Gimpel
Noisy Channel Model
Noisy Channel Model for Translating French \((x)\) to English \((y)\)

\[
P(y)
\]

\[
\begin{align*}
y^* &= \arg\max_y P(y \mid x) \\
&= \arg\max_y \frac{P(x \mid y)P(y)}{P(x)} \\
&= \arg\max_y P(x \mid y)P(y)
\end{align*}
\]
Noisy Channel

\[ y^* = \arg\max_y P(x \mid y) P(y) \]
Noisy Channel

\[ y^* = \arg\max_{y} P(x \mid y) \, P(y) \]

assumes we have the right model, and that we estimate it perfectly
Noisy Channel

\[ y^* = \arg \max_y P(x \mid y) P(y) \]

assumes we have the right model, and that we estimate it perfectly

\[ y^* = \arg \max_y P(x \mid y)^\alpha P(y)^\beta \]
Noisy Channel

\[ y^* = \arg\max_y P(x \mid y) P(y) \]

assumes we have the right model, and that we estimate it perfectly

\[ y^* = \arg\max_y P(x \mid y)^\alpha P(y)^\beta \]

\[ = \arg\max_y \alpha \log P(x \mid y) + \beta \log P(y) \]

extra parameters to tune, can tune to optimize BLEU (or whatever metric you want)

“tuning”
Noisy Channel $\rightarrow$ Linear Model?

\[
y^* = \arg\max_y \alpha \log P(x \mid y) + \beta \log P(y)
\]

since we’re not using idealized decoding rule anymore, why not add more feature functions?
Noisy Channel $\rightarrow$ Linear Model?

$$y^* = \arg\max_y \alpha \log P(x \mid y) + \beta \log P(y)$$

since we’re not using idealized decoding rule anymore, why not add more feature functions?

“word count feature”:

$$y^* = \arg\max_y \alpha \log P(x \mid y) + \beta \log P(y) + \gamma |y|$$
Noisy Channel $\rightarrow$ Linear Model?

$$y^* = \arg\max_y \alpha \log P(x \mid y) + \beta \log P(y)$$

since we’re not using idealized decoding rule anymore, why not add more feature functions?

“word count feature”:

$$y^* = \arg\max_y \alpha \log P(x \mid y) + \beta \log P(y) + \gamma |y|$$

“reverse translation model feature”:

$$y^* = \arg\max_y \alpha \log P(x \mid y) + \beta \log P(y) + \gamma |y| + \delta \log P(y \mid x)$$
Noisy Channel $\rightarrow$ Linear Model?

$\mathbf{y}^* = \arg\max_y \alpha \log P(\mathbf{x} \mid \mathbf{y}) + \beta \log P(\mathbf{y})$

since we’re not using idealized decoding rule anymore, why not add more feature functions?

“word count feature”:

$\mathbf{y}^* = \arg\max_y \alpha \log P(\mathbf{x} \mid \mathbf{y}) + \beta \log P(\mathbf{y}) + \gamma |\mathbf{y}|$

“reverse translation model feature”:

$\mathbf{y}^* = \arg\max_y \alpha \log P(\mathbf{x} \mid \mathbf{y}) + \beta \log P(\mathbf{y}) + \gamma |\mathbf{y}| + \delta \log P(\mathbf{y} \mid \mathbf{x})$

but if we keep adding features, tuning gets harder...
Noisy Channel $\rightarrow$ Linear Model

$$y^* = \arg\max_y \alpha \log p(x \mid y) + \beta \log p(y)$$

generalize to a linear model
Noisy Channel \(\rightarrow\) Linear Model

\[ y^* = \arg\max_y \alpha \log P(x \mid y) + \beta \log P(y) \]

generalize to a linear model

\[ y^* = \arg\max_y \sum_{i=1}^{K} \theta_i f_i(x, y) \]

"feature weights"  "feature functions"
Noisy Channel $\rightarrow$ Linear Model

$$y^* = \arg \max_y \alpha \log P(x \mid y) + \beta \log P(y)$$

generalize to a linear model

$$y^* = \arg \max_y \sum_{i=1}^{K} \theta_i f_i(x, y)$$

we will write it like this:

$$y^* = \arg \max_y \theta^\top f(x, y)$$

“feature weight vector”  “feature function vector”
Log-Linear Models

\[ y^* = \arg \max_y \theta^\top f(x, y) \]

if you want a probability distribution over translations of a given source sentence, exponentiate and normalize:
Log-Linear Models

\[ y^* = \arg\max_y \theta^\top f(x, y) \]

if you want a probability distribution over translations of a given source sentence, exponentiate and normalize:

\[ P(y \mid x) = \frac{\exp\{\theta^\top f(x, y)\}}{\sum_{y'} \exp\{\theta^\top f(x, y')\}} \]

this is a (conditional) “log-linear” model
More General Formulation: Derivations

\[ \langle y^*, h^* \rangle = \arg \max_{\langle y, h \rangle \in \mathcal{T}_x} \theta^T f(x, y, h) \]
More General Formulation: Derivations

\[
\langle y^*, h^* \rangle = \arg \max_{\langle y, h \rangle \in T_x} \theta^T f(x, y, h)
\]
More General Formulation: Derivations

\[ \langle y^*, h^* \rangle = \arg \max_{\langle y, h \rangle \in \mathcal{T}_x} \theta \top f(x, y, h) \]

- Feature weight vector
- Feature function vector
- Source sentence
- Translation

"derivation"

could be a phrase-based derivation, operation sequence model derivation, hierarchical or syntax-based derivation, word alignments, etc.
More General Formulation: Derivations

\[ \langle y^*, h^* \rangle = \arg\max_{\langle y, h \rangle \in \mathcal{T}_x} \theta^T f(x, y, h) \]

We won’t talk much more about derivations in this lecture, but they’re (almost) always there.
Overview

This lecture is about algorithms for choosing the feature weights $\theta$

Since we have a linear model, we can use supervised machine learning

But MT differs from typical supervised tasks (as we’ll see), so MT training procedures differ too
We’ll start with a way to visualize training for machine translation
African National Congress   opposition   sanction   Zimbabwe
非国大   反对   制裁   津巴布韦
Gold standard:
African National Congress opposes sanctions against Zimbabwe
Gold standard: African National Congress opposes sanctions against Zimbabwe

each point is a translation
Gold standard:
African National Congress opposes sanctions against Zimbabwe

model_score(x, y) = \theta^\top f(x, y)
Gold standard:
African National Congress opposes sanctions against Zimbabwe

which translation will be predicted?
Gold standard:
African National Congress opposes sanctions against Zimbabwe

predicted translation
opposition to sanctions against Zimbabwe African National Congress
African National Congress 

opposition 
sanction 
Zimbabwe 

Gold standard: 
African National Congress opposes sanctions against Zimbabwe 

predicted translation 
opposition to sanctions against Zimbabwe African National Congress 

BLEU score
Gold standard:
African National Congress opposes sanctions against Zimbabwe

African National Congress opposition sanctions against Zimbabwe

predicted translation
opposition to sanctions against Zimbabwe African National Congress

African sanctioning to Zimbabwe’s opposing

BLEU score

model score
Gold standard:
African National Congress opposes sanctions against Zimbabwe

training moves translations in this plot
Gold standard:
African National Congress opposes sanctions against Zimbabwe

training moves translations left or right in this plot
Gold standard:
African National Congress opposes sanctions against Zimbabwe

“ideal” model?
Gold standard:
African National Congress opposes sanctions against Zimbabwe

“ideal” model?
Where’s the reference (gold standard) translation?

Gold standard:
African National Congress opposes sanctions against Zimbabwe
Issue: gold standard translation is often *unreachable* by the model.

Why?
- limited translation rules,
- free translations,
- noisy data.
Each point is a translation for a single Arabic sentence.

10,000-best list from Arabic-English phrase-based Moses system, default Moses weights.
1-best: 46 BLEU
Why are there horizontal “bands"?
Why are there horizontal “bands”?

same translation with different derivations, different translations with same BLEU
Roadmap

We’ll cover some of the most widely-used discriminative training algorithms for MT:

- Minimum Error Rate Training*
- Minimum Bayes Risk
- Pairwise Ranking Optimization*
- Batch MIRA*

* implemented in Moses!
Minimum Error Rate Training in Statistical Machine Translation

Franz Josef Och
Information Sciences Institute
University of Southern California
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292
och@isi.edu
Minimum Error Rate Training (MERT)

\[
\min_{\theta} \text{error}(\text{predicted translations})
\]

\[
\min_{\theta} -\text{BLEU}(\text{references, predicted translations})
\]

\[
\max_{\theta} \text{BLEU}(\text{references, predicted translations})
\]

\[
\max_{\theta} \text{BLEU}(\text{references,} \{\text{translate}(x_i, \theta)\}_{i=1}^{N})
\]

\[
\text{translate}(x, \theta) = \arg\max_y \theta^\top f(x, y)
\]
Minimum Error Rate Training (MERT)

$$\max_{\theta} \text{BLEU}(\text{references}, \{\text{translate}(x_i, \theta)\}_{i=1}^N)$$

optimizing this objective is intractable in general – how can we do it?

- generate k-best lists of translations,
- approximately optimize on k-best lists,
- repeat with new parameters
- (pool k-best lists across iterates)
MERT

given a k-best list,

randomly choose a feature (e.g., reverse translation model) and vary a step size $\delta$:

$$\theta^\top f(x, y) + \delta \log P(y | x)$$

find value of $\delta$ that maximizes BLEU
more generally, randomly choose a “search direction” $\psi$ and vary a scalar multiplier $\delta$:

$$\theta^T f(x, y) + \delta \psi^T f(x, y)$$
MERT

more generally,

randomly choose a “search direction” $\psi$ and vary a scalar multiplier $\delta$:

$$\theta^\top f(x, y) + \delta \psi^\top f(x, y)$$

single-feature special case: let $\psi$ be a “one-hot” vector (all zeroes except a single 1)

for the reverse translation model feature one-hot vector $\psi$, this becomes:

$$\theta^\top f(x, y) + \delta \log P(y \mid x)$$
each line is a translation from the k-best list:

credit: Chris Dyer
$\theta^T f(x, y) + \delta \log P(y | x)$

MERT

as we vary $\delta$, the model score of the translation changes

credit: Chris Dyer
each line is a translation from the k-best list:

MERT

model
score

opposition to sanctions against Zimbabwe's opposing credit: Chris Dyer

African National Congress opposition to sanctions against Zimbabwe

African sanctioning to Zimbabwe’s opposing

credit: Chris Dyer
MERT

\[ \max_y \theta^\top f(x, y) + \delta \log P(y \mid x) \]

credit: Chris Dyer
\[
\text{argmax}_y \theta^\top f(x, y) + \delta \log P(y \mid x)
\]

credit: Chris Dyer
\[ \langle y_{17}^*, h_{17}^* \rangle \]  
\[ \langle y_{38}^*, h_{38}^* \rangle \]  
\[ \langle y_4^*, h_4^* \rangle \]
compute error (negative BLEU) for each translation

credit: Chris Dyer
repeat for each sentence, add to “error surface”:

credit: Chris Dyer
repeat for each sentence, add to error surface:

credit: Chris Dyer
\[ \theta^{\text{new}} = \theta + \delta^* \psi \]
Error Surface (Smith and Eisner, 2006)

Figure 1: The loss surface for a machine translation system: while other parameters are held constant, we vary the weights on the distortion and word penalty features. Note the piecewise constant regions with several local maxima.
10,000-best list, default Moses weights

1-best: 28 BLEU
same sentence, 10,000-best list after MERT

1-best: 34 BLEU
another sentence, default Moses weights

1-best: 46 BLEU
same sentence, after MERT

1-best: 62 BLEU
same sentence, after MERT

MERT is implemented in Moses: mert-moses.pl

1-best: 62 BLEU
\[
\max_{\theta} \text{BLEU}(\text{references}, \{\text{translate}(x_i, \theta)\}_{i=1}^{N})
\]

Problems with the MERT objective function?

Discontinuous & non-convex $\rightarrow$ optimization relies on randomized search

No regularization $\rightarrow$ frequently overfits to tuning set

As a result, MERT is only effective for very small models (<40 parameters)
Smoothing the Objective

\[
\max_{\theta} \text{BLEU} \left( \text{references, } \left\{ \arg\max_y \theta^\top f(x^{(i)}, y) \right\}_{i=1}^N \right)
\]

first, convert to sentence-level objective:

\[
\max_{\theta} \sum_{i=1}^N \text{BLEU}^{+1} \left( \text{references}^{(i)}, \arg\max_y \theta^\top f(x^{(i)}, y) \right)
\]

then, smooth using expectation under log-linear distribution:

\[
\max_{\theta} \sum_{i=1}^N \sum_{y} \text{BLEU}^{+1} \left( \text{references}^{(i)}, y \right) \frac{\exp\{\theta^\top f(x^{(i)}, y)\}^\alpha}{\sum_{y'} \exp\{\theta^\top f(x^{(i)}, y')\}^\alpha}
\]

alpha = “smoothness factor”
Smoothing the Objective

\[
\max_\theta \text{BLEU}\left(\text{references, } \{\arg\max_y \theta^\top f(x^{(i)}, y)\}_{i=1}^N\right)
\]

first, convert to sentence-level objective:

\[
\max_\theta \sum_{i=1}^N \text{BLEU}\left(\text{references}^{(i)}, y\right)
\]

This is often called “risk” or “Bayes risk”

\[
\max_\theta \sum_{i=1}^N \sum_y \text{BLEU}^{+1}\left(\text{references}^{(i)}, y\right) \frac{\exp\{\theta^\top f(x^{(i)}, y)\}^\alpha}{\sum_{y'} \exp\{\theta^\top f(x^{(i)}, y')\}^\alpha}
\]

alpha = “smoothness factor”
Smoothing Error Surfaces (Och, 2003)
Many other researchers tried to improve MERT:

Regularization and Search for MERT (Cer et al., 2008)
Random Restarts in MERT for MT (Moore & Quirk, 2008)
Stabilizing MERT (Foster & Kuhn, 2009)

Issues remain:

Better Hypothesis Testing for Statistical MT: Controlling for Optimizer Instability (Clark et al., 2011)

They suggest running MERT 3-5 times due to its instability
MERT doesn’t scale

Synthetic weight learning of MERT

The synthetic experiment in ideal conditions validates what has long been accepted as truth.
MERT doesn’t scale

Synthetic weight learning of MERT

Tuning as Ranking

Mark Hopkins and Jonathan May
SDL Language Weaver
Los Angeles, CA 90045
{mhopkins, jmay}@sdl.com

Number of features

The synthetic experiment in ideal conditions validates what has long been accepted as truth.
Pairwise Ranking Optimization
(Hopkins & May, 2011)

- generate k-best lists
Pairwise Ranking Optimization
(Hopkins & May, 2011)

- generate k-best lists
- sample translation pairs from k-best list
Pairwise Ranking Optimization
(Hopkins & May, 2011)

- generate k-best lists
- sample translation pairs from k-best list
- train a binary classifier to make the pairs’ model score difference have same sign as the BLEU difference
Pairwise Ranking Optimization
(Hopkins & May, 2011)

- generate k-best lists
- sample translation pairs from k-best list

implemented in Moses:
mert-moses.pl --pairwise-ranked
How about standard machine learning algorithms for structured prediction?
Structured Perceptron?
(Collins, 2002)
Structured Perceptron
(Collins, 2002)

reference

model prediction
Structured Perceptron for MT?

Model prediction

Reference
k-Best Perceptron for MT

(Liang et al., 2006)
k-Best Perceptron for MT
(Liang et al., 2006)
k-Best Perceptron for MT

(Liang et al., 2006)
“Hope-Fear” MIRA

(Chiang et al., 2008; 2009; Cherry & Foster, 2012; Chiang, 2012)
“Hope-Fear” MIRA
(Chiang et al., 2008; 2009; Cherry & Foster, 2012; Chiang, 2012)

max \left( \theta^\top f(x, y) + \text{BLEU}(\text{reference}, y) \right)

max \left( \theta^\top f(x, y) - \text{BLEU}(\text{reference}, y) \right)
“Hope-Fear” MIRA

(Chiang et al., 2008; 2009; Cherry & Foster, 2012; Chiang, 2012)

implemented in Moses:
`mert-moses.pl --batch-mira`

$max (\theta^T f(x, y) + \text{BLEU}(\text{reference}, y))$

$max (\theta^T f(x, y) - \text{BLEU}(\text{reference}, y))$

"hope" translation

"fear" translation
## MT Experiments

*(Gimpel, 2012)*

Averages over 8 test sets across 3 language pairs

<table>
<thead>
<tr>
<th></th>
<th>Moses %BLEU</th>
<th>Hiero %BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERT</td>
<td>35.9</td>
<td>37.0</td>
</tr>
<tr>
<td>PRO</td>
<td>35.9</td>
<td>36.9</td>
</tr>
<tr>
<td>Bayes Risk</td>
<td>35.6</td>
<td>36.4</td>
</tr>
<tr>
<td>Fear MIRA</td>
<td>34.9</td>
<td>34.2</td>
</tr>
<tr>
<td>Hope MIRA</td>
<td>35.2</td>
<td>36.0</td>
</tr>
<tr>
<td>Hope-Fear MIRA</td>
<td>35.7</td>
<td><strong>37.0</strong></td>
</tr>
</tbody>
</table>
Questions?
2002:
conditional log-likelihood

Discriminative Training and Maximum Entropy Models for Statistical Machine Translation

Franz Josef Och and Hermann Ney
Lehrstuhl für Informatik VI, Computer Science Department
RWTH Aachen - University of Technology
D-52056 Aachen, Germany
{och,ney}@informatik.rwth-aachen.de
2002:
conditional log-likelihood

Discriminative Training and Maximum Entropy Models for Statistical Machine Translation

Table 2: Effect of maximum entropy training for alignment template approach (WP: word penalty feature, CLM: class-based language model (five-gram), MX: conventional dictionary).

<table>
<thead>
<tr>
<th></th>
<th>objective criteria [%]</th>
<th>subjective criteria [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SER</td>
<td>WER</td>
</tr>
<tr>
<td>Baseline($\lambda_m = 1$)</td>
<td>86.9</td>
<td>42.8</td>
</tr>
<tr>
<td>ME</td>
<td>81.7</td>
<td>40.2</td>
</tr>
<tr>
<td>ME+WP</td>
<td>80.5</td>
<td>38.6</td>
</tr>
<tr>
<td>ME+WP+CLM</td>
<td>78.1</td>
<td>38.3</td>
</tr>
<tr>
<td>ME+WP+CLM+MX</td>
<td>77.8</td>
<td>38.4</td>
</tr>
</tbody>
</table>
Hinge Loss

gold standard

cost-augmented prediction