Discriminative Training of Translation Models
Noisy Channels Again

\[ p(e) \]

source \(\rightarrow\) English
Noisy Channels Again

\[ p(e) \quad \rightarrow \quad \text{English} \quad \rightarrow \quad p(g \mid e) \quad \rightarrow \quad \text{German} \]
Noisy Channels Again

\[
p(e) \quad \rightarrow \quad \text{English} \quad \rightarrow \quad \text{German}
\]

\[
e^* = \arg \max_e p(e \mid g)
\]

\[
= \arg \max_e \frac{p(g \mid e) \times p(e)}{p(g)}
\]

\[
= \arg \max_e p(g \mid e) \times p(e)
\]
Noisy Channels Again

\[ e^* = \arg \max_e p(e \mid g) \]

\[ = \arg \max_e \frac{p(g \mid e) \times p(e)}{p(g)} \]

\[ = \arg \max_e p(g \mid e) \times p(e) \]
Noisy Channels Again

\[ e^* = \arg \max_e p(e \mid g) \]
\[ = \arg \max_e \frac{p(g \mid e) \times p(e)}{p(g)} \]
\[ = \arg \max_e p(g \mid e) \times p(e) \]
\[ = \arg \max_e \log p(g \mid e) + \log p(e) \]
Noisy Channels Again

\[ e^* = \arg \max_e p(e \mid g) \]

\[ = \arg \max_e \frac{p(g \mid e) \times p(e)}{p(g)} \]

\[ = \arg \max_e p(g \mid e) \times p(e) \]

\[ = \arg \max_e \log p(g \mid e) + \log p(e) \]

\[ = \arg \max_e \begin{bmatrix} 1 & \top \\ 1 \end{bmatrix}^\top \begin{bmatrix} \log p(g \mid e) \\ \log p(e) \end{bmatrix} \]

\[ \mathbf{w}^\top \begin{bmatrix} \log p(g \mid e) \\ \log p(e) \end{bmatrix} \]

h(g,e)
Noisy Channels Again

\[
\begin{align*}
\mathbf{e}^* &= \arg \max_{\mathbf{e}} p(\mathbf{e} | \mathbf{g}) \\
&= \arg \max_{\mathbf{e}} \frac{p(\mathbf{g} | \mathbf{e}) \times p(\mathbf{e})}{p(\mathbf{g})} \\
&= \arg \max_{\mathbf{e}} p(\mathbf{g} | \mathbf{e}) \times p(\mathbf{e})
\end{align*}
\]

This is a linear combination

\[
\begin{align*}
\mathbf{e}^* &= \arg \max_{\mathbf{e}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}^T \begin{bmatrix} \log p(\mathbf{g} | \mathbf{e}) \\ \log p(\mathbf{e}) \end{bmatrix} \\
&= \arg \max_{\mathbf{e}} \begin{bmatrix} \log p(\mathbf{g} | \mathbf{e}) \\ \log p(\mathbf{e}) \end{bmatrix}^T \begin{bmatrix} \mathbf{w}^T \\ h(\mathbf{g}, \mathbf{e}) \end{bmatrix}
\end{align*}
\]
The Noisy Channel

\[-\log p(g|e)\]

\[-\log p(e)\]
As a Linear Model

\[-\log p(g | e)\]

\[-\log p(e)\]
As a Linear Model

\[-\log p(g|e)\]

\[-\log p(e)\]
As a Linear Model

Improvement 1:
change $\vec{w}$ to find better translations
As a Linear Model

\[-\log p(g|e)\]

\[-\log p(e)\]
As a Linear Model

\[-\log p(g|e)\]

\[-\log p(e)\]
As a Linear Model
As a Linear Model

\[-\log p(g|e)\]

**Improvement 2:**

Add dimensions to make points **separable**
Linear Models

\[ e^* = \arg \max_e w^\top h(g, e) \]

- Improve the modeling capacity of the noisy channel in two ways
- Reorient the weight vector
- Add new dimensions (new features)

Questions

- What features?
- How do we set the weights?
Mann  beißt  Hund
Mann beißt Hund

\[ x \text{ BITES } y \]
Mann beißt Hund
man bites cat
man chase dog
man bite cat
man bite dog
man bites man
man bites dog
Man beißt Hund

$\textit{x BITES y}$

Man bite cat
Man chase dog

Man bites cat
Man bite dog
Man bites dog
Mann beißt Hund

man bites cat

man chase dog

dog bites man

\[ x \text{ BITES } y \]
Mann beißt Hund

man bites cat

Mann beißt Hund

man chase dog

Mann beißt Hund

dog bites man
Mann beißt Hund

$\mathbf{x \text{ BITES } y}$

man bites cat

man chase dog

man bite cat

man bite dog

dog bites man
Feature Classes

**Lexical**

Are lexical choices appropriate?

*bank* = “River bank” vs. “Financial institution”
Feature Classes

Lexical

Are lexical choices appropriate?

\[ \text{bank} = \text{“River bank” vs. “Financial institution”} \]

Configurational

Are semantic/syntactic relations preserved?

“Dog bites man” vs. “Man bites dog”
Feature Classes

**Lexical**

Are lexical choices appropriate?

bank = “River bank” vs. “Financial institution”

**Configurational**

Are semantic/syntactic relations preserved?

“Dog bites man” vs. “Man bites dog”

**Grammatical**

Is the output fluent / well-formed?

“Man *bites* dog” vs. “Man *bite* dog”
What do lexical features look like?

<table>
<thead>
<tr>
<th>Mann</th>
<th>beißt</th>
<th>Hund</th>
</tr>
</thead>
<tbody>
<tr>
<td>man</td>
<td>bites</td>
<td>cat</td>
</tr>
</tbody>
</table>
What do lexical features look like?

Mann  beißt  Hund

man  bites  cat
What do lexical features look like?

First attempt:

\[ score(g, e) = w^\top h(g, e) \]

\[ h_{15,342}(g, e) = \begin{cases} 
1, & \exists i, j : g_i = \text{Hund}, e_j = \text{cat} \\
0, & \text{otherwise} 
\end{cases} \]
What do lexical features look like?

First attempt:

\[
\text{score}(g, e) = w^\top h(g, e)
\]

\[
h_{15,342}(g, e) = \begin{cases} 
1, & \exists i, j : g_i = \text{Hund}, e_j = \text{cat} \\
0, & \text{otherwise}
\end{cases}
\]

But what if a \textbf{cat} is being chased by a \textbf{Hund}?
What do lexical features look like?

Latent variables enable more precise features:

\[
\text{score}(g, e, a) = w^\top h(g, e, a)
\]

\[
h_{15,342}(g, e, a) = \sum_{(i,j)\in a} \begin{cases} 
1, & \text{if } g_i = \text{Hund}, e_j = \text{cat} \\
0, & \text{otherwise}
\end{cases}
\]
Standard Features

• Target side features
  • log $p(e)$ [n-gram language model ]
  • Number of words in hypothesis

• Source + target features
  • log relative frequency $e|f$ of each rule [ log #(e,f) - log #(f) ]
  • log relative frequency $f|e$ of each rule [ log #(e,f) - log #(e) ]
  • “lexical translation” log probability $e|f$ of each rule [ $\approx \log p_{model1}(e|f)$ ]
  • “lexical translation” log probability $f|e$ of each rule [ $\approx \log p_{model1}(f|e)$ ]

• Other features
  • Count of rules/phrases used
  • Reordering pattern probabilities
Parameter Learning
Hypothesis Space
Hypothesis Space
Hypothesis Space

$h_1$

$h_2$

Hypotheses
Hypothesis Space
Preliminaries

We assume a **decoder** that computes:

\[
\langle e^*, a^* \rangle = \arg \max_{\langle e, a \rangle} w^\top h(g, e, a)
\]

And **K-best lists** of, that is:

\[
\{ \langle e_i^*, a_i^* \rangle \}_{i=1}^K = \arg \text{i}^{\text{th}} - \max_{\langle e, a \rangle} w^\top h(g, e, a)
\]

**Standard, efficient algorithms exist for this.**
Learning Weights

- Try to match the reference translation exactly
  - **Conditional random field**
    - Maximize the conditional probability of the reference translations
    - “Average” over the different latent variables
Learning Weights

• Try to match the reference translation exactly

• Conditional random field
  • Maximize the conditional probability of the reference translations
  • “Average” over the different latent variables

• Max-margin
  • Find the weight vector that separates the reference translation from others by the maximal margin
  • Maximal setting of the latent variables
Problems

• These methods give “full credit” when the model \textit{exactly} produces the reference and no credit otherwise

• \textbf{What is the problem with this?}
Problems

• These methods give “full credit” when the model *exactly* produces the reference and no credit otherwise

• **What is the problem with this?**
  • There are many ways to translate a sentence
  • What if we have multiple reference translations?
  • **What about partial credit?**
Cost-Sensitive Training

• Assume we have a **cost function** that gives a score for how good/bad a translation is

\[ \ell(\hat{e}, E) \mapsto [0, 1] \]

• Optimize the weight vector by making reference to this function

• We will talk about two ways to do this
K-Best List Example
K-Best List Example

![Diagram with points and vectors labeled #1 to #10]
K-Best List Example

$h_1$ vs. $h_2$

- $0.8 \leq \ell < 1.0$
- $0.6 \leq \ell < 0.8$
- $0.4 \leq \ell < 0.6$
- $0.2 \leq \ell < 0.4$
- $0.0 \leq \ell < 0.2$
Training as Classification

• **Pairwise Ranking Optimization**
  
  • Reduce training problem to **binary classification** with a **linear model**

• **Algorithm**
  
  • For $i=1$ to $N$
    
    • Pick random pair of hypotheses (A,B) from $K$-best list
    
    • Use cost function to determine if is A or B better
    
    • Create $i$th training instance
    
    • Train binary linear classifier
Worse!

- $0.8 \leq \ell < 1.0$
- $0.6 \leq \ell < 0.8$
- $0.4 \leq \ell < 0.6$
- $0.2 \leq \ell < 0.4$
- $0.0 \leq \ell < 0.2$
Better!
Better!

- 0.8 \leq \ell < 1.0
- 0.6 \leq \ell < 0.8
- 0.4 \leq \ell < 0.6
- 0.2 \leq \ell < 0.4
- 0.0 \leq \ell < 0.2
Fit a linear model
Fit a linear model
K-Best List Example

\[ 0.8 \leq \ell < 1.0 \]
\[ 0.6 \leq \ell < 0.8 \]
\[ 0.4 \leq \ell < 0.6 \]
\[ 0.2 \leq \ell < 0.4 \]
\[ 0.0 \leq \ell < 0.2 \]
MERT

- **Minimum Error Rate Training**
- Directly target an automatic evaluation metric
  - BLEU is defined at the corpus level
  - MERT optimizes at the corpus level
- **Downsides**
  - Does not deal well with > ~20 features
Given weight vector $w$, any hypothesis $\langle e, a \rangle$ will have a (scalar) score $m = w^\top h(g, e, a)$

Now pick a search vector $v$, and consider how the score of this hypothesis will change:

$$w_{\text{new}} = w + \gamma v$$
Given weight vector $w$, any hypothesis $\langle e, a \rangle$ will have a (scalar) score $m = w^\top h(g, e, a)$

Now pick a **search vector** $v$, and consider how the score of this hypothesis will change:

$$w_{\text{new}} = w + \gamma v$$

$$m = (w + \gamma v)^\top h(g, e, a)$$
MERT

Given weight vector \( w \), any hypothesis \( \langle e, a \rangle \) will have a (scalar) score \( m = w^\top h(g, e, a) \)

Now pick a **search vector** \( v \), and consider how the score of this hypothesis will change:

\[
\begin{align*}
\mathbf{w}_{\text{new}} &= \mathbf{w} + \gamma \mathbf{v} \\
\quad m &= (\mathbf{w} + \gamma \mathbf{v})^\top h(g, e, a) \\
&= w^\top h(g, e, a) + \gamma v^\top h(g, e, a)
\end{align*}
\]
Given weight vector $w$, any hypothesis $\langle e, a \rangle$ will have a (scalar) score $m = w^\top h(g, e, a)$

Now pick a search vector $v$, and consider how the score of this hypothesis will change:

$$w_{\text{new}} = w + \gamma v$$

$$m = (w + \gamma v)^\top h(g, e, a)$$

$$= w^\top h(g, e, a) + \gamma v^\top h(g, e, a)$$

$$m = a\gamma + b$$
Given weight vector $w$, any hypothesis $\langle e, a \rangle$ will have a (scalar) score $m = w^\top h(g, e, a)$

Now pick a search vector $v$, and consider how the score of this hypothesis will change:

$$w_{\text{new}} = w + \gamma v$$

$$m = (w + \gamma v)^\top h(g, e, a)$$

$$= w^\top h(g, e, a) + \gamma v^\top h(g, e, a)$$

$$= a \gamma + b$$
MERT

Given weight vector \( w \), any hypothesis \( \langle e, a \rangle \) will have a (scalar) score \( m = w^\top h(g, e, a) \)

Now pick a search vector \( v \), and consider how the score of this hypothesis will change:

\[
w_{\text{new}} = w + \gamma v
\]

\[
m = (w + \gamma v)^\top h(g, e, a) = w^\top h(g, e, a) + \gamma v^\top h(g, e, a)
\]

\[
\boxed{m = a\gamma + b}
\]

Linear function in 2D!
MERT
Recall our k-best set \( \{ \langle e_i^*, a_i^* \rangle \}_{i=1}^K \)
Recall our $k$-best set $\{\langle e_i^*, a_i^* \rangle\}_{i=1}^K$
\[ \langle e_{\ast 162}, a_{\ast 162} \rangle \quad \langle e_{\ast 28}, a_{\ast 28} \rangle \quad \langle e_{\ast 73}, a_{\ast 73} \rangle \]
MERT

\[ \langle e_{162}^*, a_{162}^* \rangle \]

\[ \langle e_{28}^*, a_{28}^* \rangle \]

\[ \langle e_{73}^*, a_{73}^* \rangle \]

\[ m \]

\[ \gamma \]

errors
Let \( w_{\text{new}} = \gamma^* v + w \)
MERT

• In practice “errors” are sufficient statistics for evaluation metrics (e.g., BLEU)
  • Can maximize or minimize!

• Envelope can also be computed using dynamic programming
  • Interesting complexity bounds

• How do you pick the search direction?
Summary

• Evaluation metrics
  • Figure out how well we’re doing
  • Figure out if a feature helps
  • But ALSO: train your system!

• What’s a great way to improve translation?
  • Improve evaluation!
Thank You!

\[ \langle e_{162}^*, a_{162}^* \rangle \]

\[ \langle e_{28}^*, a_{28}^* \rangle \]

\[ \langle e_{73}^*, a_{73}^* \rangle \]