Tuning As Ranking

Mark Hopkins
Jonathan May
SDL Language Weaver

EMNLP
July 29, 2011
What we did

We replaced MERT’s linear optimization with a linear binary classifier, and fed it pairs of translations, effecting a ranking.

What we found

- **Scalable** to many features
- **Consistent** results
- **Parity** with leading techniques
- Very **fast**

Any Questions?

Wednesday, August 3, 2011
Which is best?

-il ne va pas

he goes not

he doesn't go

she not go

(Image credit: Silverstein, 1981)
Which is best?

A good scoring function can tell us...
Which is best?

We should avoid bad functions
Which is best?

How do we ensure “proper” scores?

il ne va pas → he goes not

→ he doesn’t go

→ she not go
Properties of the translation
Properties of the translation

he goes not

literal meaning?
Properties of the translation

literal meaning?
fluency?

he goes not
Properties of the translation

- literal meaning?
- fluency?
- word count?
- count of “he”?
- count of “coffee”?
- alliterative?
- $2 \leq \text{count(“o”)} \leq 3$?
- how do you feel?

he goes not
Properties of the translation

Features!

- literal meaning?
- fluency?
- word count?
- count of “he”? 2 ≤ count(“o”) ≤ 3?
- count of “coffee”?
- alliterative?
- how do you feel? 32.8

he goes not
Properties of the translation

Features!

he goes not
Properties of the translation

Features!

Form a weighted sum

\[-2 \times 2 + 3 \times 4 = 8\]

Weights!

he goes not
Translations are feature vectors

il ne va pas

he goes not

he doesn’t go

she not go

Wednesday, August 3, 2011
Weight vector determines the score

\[ w: \begin{bmatrix} -2 \\ 3 \end{bmatrix} \]

\[ \begin{align*}
2 & \quad 4 & = & & 8 \\
3 & \quad 8 & = & & 18 \\
6 & \quad 1 & = & & -9 \\
\end{align*} \]

\[ \text{il ne va pas} \]
Weight vector determines the score

model
score

features

weights

w: -2 3

h = f \bullet w

2 4 = 8

3 8 = 18

6 1 = -9

il ne va pas
Weight vector determines the score

\[ il\text{ ne va pas } \]

\[ h = f \odot w \]

\[ \begin{array}{c}
2 \\
3 \\
6 \\
\end{array} \begin{array}{c}
4 \\
8 \\
1 \\
\end{array} = \begin{array}{c}
-2 \\
-8 \\
27 \\
\end{array} \]

\[ \text{model} \quad \text{score} \quad \text{features} \quad \text{weights} \]

\[ \begin{array}{c}
\text{w:} \\
5 \\
-3 \\
\end{array} \]

Wednesday, August 3, 2011
Weight vector determines the score

Tuning is all about choosing this vector

w: 5 -3

features  
2 4 = -2
3 8 = -8
6 1 = 27

model score
Weight vector determines the score

We should choose a vector that matches an extrinsic score

<table>
<thead>
<tr>
<th>features</th>
<th>model score</th>
<th>extrinsic score</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 4</td>
<td>-2</td>
<td>.28</td>
</tr>
<tr>
<td>3 8</td>
<td>-8</td>
<td>.44</td>
</tr>
<tr>
<td>6 1</td>
<td>27</td>
<td>.12</td>
</tr>
</tbody>
</table>

w: 5 -3

(BLEU+1) (Lin & Och, '04)
Weight vector determines the score

We should choose a vector that matches an extrinsic score

| w: | 5 | -3 |

<table>
<thead>
<tr>
<th>ranks</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
</tr>
<tr>
<td>-2</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>-8</td>
</tr>
<tr>
<td>A</td>
</tr>
<tr>
<td>27</td>
</tr>
</tbody>
</table>

Bad match!
Weight vector determines the score

We should choose a vector that matches an extrinsic score

$w$: 

$\begin{align*}
&\text{features} \\
&2 \quad 4 \\
&3 \quad 8 \\
&6 \quad 1 \\
\end{align*}$

$\begin{align*}
&\text{B} \quad \text{B} \\
&0 \quad .28 \\
&A \quad A \\
&2 \quad .44 \\
&C \quad C \\
&-11 \quad .12 \\
\end{align*}$

Good match!
The tuning framework that everybody uses

MERT framework

(Och, 2003)
The tuning framework that everybody uses

\[\text{(almost)}\] * Not David Chiang

MERT framework

* Not David Chiang

\[(\text{Och}, 2003)\]
The tuning framework that everybody uses

(Almost)*

Candidate Generation

MERT framework

Generate n-best per input sentence

Add to Candidate Pool

* Not David Chiang

(Och, 2003)
The tuning framework that everybody uses

Candidate Generation

MERT framework

Generate n-best per input sentence

Add to Candidate Pool

* Not David Chiang

(Och, 2003)
The tuning framework that everybody uses (almost)*

* Not David Chiang

MERT framework

Candidate Generation

Learned weight vector

Generate n-best per input sentence

Add to Candidate Pool

Weight Optimization

w:
How MERT works

The MERT algorithm works by varying one weight at a time to find a value for that weight that aligns the best model score with the best extrinsic score.
How MERT works

The MERT algorithm works by varying one weight at a time to find a value for that weight that aligns the best model score with the best extrinsic score.

<table>
<thead>
<tr>
<th>feats</th>
<th>model</th>
<th>extrins</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>-3</td>
<td>-3</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>-5</td>
<td>-3</td>
<td>0</td>
</tr>
</tbody>
</table>

W: 0 0 

Total extrinsic: .43
The MERT algorithm works by varying one weight at a time to find a value for that weight that aligns the best model score with the best extrinsic score.

How MERT works:

(MERT can optimize the non-decomposable BLEU; swap these for n-gram component values and determine total with the BLEU equation)
The MERT algorithm works by varying one weight at a time to find a value for that weight that aligns the best model score with the best extrinsic score.
How MERT works

The MERT algorithm works by varying one weight at a time to find a value for that weight that aligns the best model score with the best extrinsic score.

<table>
<thead>
<tr>
<th>feats</th>
<th>model</th>
<th>extrins</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>-2</td>
<td>.28</td>
</tr>
<tr>
<td>3</td>
<td>-3</td>
<td>.44</td>
</tr>
<tr>
<td>6</td>
<td>-6</td>
<td>.12</td>
</tr>
<tr>
<td>-3</td>
<td>3</td>
<td>.15</td>
</tr>
<tr>
<td>1</td>
<td>-1</td>
<td>.18</td>
</tr>
<tr>
<td>-5</td>
<td>5</td>
<td>.32</td>
</tr>
</tbody>
</table>

Total extrinsic: .60
How MERT works

The MERT algorithm works by varying one weight at a time to find a value for that weight that aligns the best model score with the best extrinsic score.
How MERT works

The MERT algorithm works by varying one weight at a time to find a value for that weight that aligns the best model score with the best extrinsic score.

### How MERT works

<table>
<thead>
<tr>
<th>feats</th>
<th>model score</th>
<th>extrinsic score</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>-3</td>
<td>-5</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>-5</td>
<td>2</td>
</tr>
<tr>
<td>S2</td>
<td>-3</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>-3</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

**Total extrinsic**: .62
How MERT works

The MERT algorithm works by varying one weight at a time to find a value for that weight that aligns the best model score with the best extrinsic score.

The table shows the model and extrinsic scores for two different sets, S1 and S2. The weights (w) for varying and holding are indicated for each set.

For S1:
- Model score: 4
- Extrinsic score: 0
- Total extrinsic: 0.28

For S2:
- Model score: 3
- Extrinsic score: -11
- Total extrinsic: 0.12

The best extrinsic score is 0.32, achieved by varying the weights as indicated.
How MERT works

This works well for small feature sets, but as the feature space grows, it is hard to find a good position.
Synthetic Experiment

“Candidate pool” of randomly drawn “feature” vectors
Synthetic Experiment

“Candidate pool” of randomly drawn “feature” vectors

How to determine “extrinsic score”??
Synthetic Experiment

“Candidate pool” of randomly drawn “feature” vectors

Secret “goal weights” used to calculate extrinsic score
Synthetic Experiment

Now use MERT to try and learn the goal weights back.

This is linear equation solving.

It’s much easier than MT tuning.
Synthetic Experiment

Now use MERT to try and learn the goal weights back

This is linear equation solving

It’s much easier than MT tuning
MERT doesn’t scale

The synthetic experiment in ideal conditions validates what has long been accepted as truth.
MERT only cares about the top-scoring translation
MERT only cares about the top-scoring translation.

MERT doesn’t care about these feats.

Wednesday, August 3, 2011
It doesn’t care about matching the overall ranking

S1

S2

mismatch!
This could lead to poor generalization

<table>
<thead>
<tr>
<th>feats</th>
<th>model</th>
<th>extrins</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>D</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>A</td>
</tr>
<tr>
<td>6</td>
<td>I</td>
<td>C</td>
</tr>
<tr>
<td>7</td>
<td>-4</td>
<td>J</td>
</tr>
<tr>
<td>12</td>
<td>-2</td>
<td>B</td>
</tr>
</tbody>
</table>

... ... ... ...

| S1 | -12 | 5 | Z | D |

not good but liked by model

good but disliked
We should focus on rank

Recognize that these are different solutions!
We should focus on rank

Recognize that these are different solutions! (To MERT they are the same)
We can describe rank from a \textbf{pairwise} perspective

For any two translations $a$ and $b$ of the same sentence

(Herbrich et al., ’99)
We can describe rank from a pairwise perspective.

Model and extrinsic score order should agree.
We can describe rank from a pairwise perspective.

translation a

\[
\begin{align*}
&f_a \quad h_a \quad g_a \\
&\text{extrinsic} \\
&g_a > g_b
\end{align*}
\]

translation b

\[
\begin{align*}
&f_b \quad h_b \quad g_b \\
&\text{model} \\
&h_a > h_b
\end{align*}
\]

\[
\begin{align*}
&h_a - h_b > 0
\end{align*}
\]
We can describe rank from a \textbf{pairwise} perspective

\begin{align*}
\text{translation a} & \quad \text{translation b} \\
\begin{array}{c}
\hat{f}_a \\
\hat{h}_a \\
\hat{g}_a
\end{array} & \quad \begin{array}{c}
\hat{f}_b \\
\hat{h}_b \\
\hat{g}_b
\end{array}
\end{align*}

\textbf{extrinsic}
\begin{align*}
\hat{g}_a & \succ \hat{g}_b \\
\hat{h}_a & \succ \hat{h}_b \\
\hat{w} \cdot \hat{f}_a & \succ \hat{w} \cdot \hat{f}_b
\end{align*}

\textbf{model}
\begin{align*}
\hat{h}_a & \succ \hat{h}_b \\
\hat{w} \cdot \hat{f}_a & \succ \hat{w} \cdot \hat{f}_b
\end{align*}
We can describe rank from a pairwise perspective.

**translation a**

\[ f_a \quad h_a \quad g_a \]

**extrinsic**

\[ g_a > g_b \]

**translation b**

\[ f_b \quad h_b \quad g_b \]

**model**

\[ h_a > h_b > 0 \]

\[ w \cdot f_a - w \cdot f_b > 0 \]

\[ w \cdot (f_a - f_b) > 0 \]

Wednesday, August 3, 2011
This is a binary classification problem.

extrinsic

\[ g_a > g_b \]

\[ w \cdot (f_a - f_b) > 0 \]
This is a binary classification problem.

Extrinsic:
\[ g_a > g_b \]

Model:
\[ w \cdot (f_a - f_b) > 0 \]

Label:
(+ if a is better, - if b is better)

Training instance:
(difference vector)
Find the separating vector

\[ \Delta f_1 \]

\[ \Delta f_2 \]
Find the separating vector $\mathbf{w}$ such that $\mathbf{w} \cdot (\mathbf{f}_a - \mathbf{f}_b) = 0$. 

\[
\Delta f_1 \\
\Delta f_2
\]

Wednesday, August 3, 2011
Find the separating vector $w$ such that $fa - fb = 0$.

- $w \cdot (fa - fb) < 0$,
- $w \cdot (fa - fb) > 0$,
- $w \cdot (fa - fb) = 0$.
Find the separating vector

\[ w \cdot (f_a - f_b) = \begin{cases} > 0 & \text{if} \quad w \cdot (f_a - f_b) > 0 \\ < 0 & \text{if} \quad w \cdot (f_a - f_b) < 0 \\ = 0 & \text{if} \quad w \cdot (f_a - f_b) = 0 \end{cases} \]

“I have a good tool for finding that vector!” (Daumé III, ’04)
Find the separating vector

\[ \mathbf{w} \cdot (f_a - f_b) = 0 \]

\[ \mathbf{w} \cdot (f_a - f_b) > 0 \]

\[ \mathbf{w} \cdot (f_a - f_b) < 0 \]

“\text{I have a good tool for finding that vector!}” (Daumé III, ’04)
Find the separating vector

\[ w \cdot (f_a - f_b) > 0 \]

\[ w \cdot (f_a - f_b) < 0 \]

\[ w \cdot (f_a - f_b) = 0 \]

Daumé III, ’04  Manning & Klein, ’03
Find the separating vector $w \cdot (f_a - f_b) = 0$

$w \cdot (f_a - f_b) > 0$

$w \cdot (f_a - f_b) < 0$

Daumé III, ’04  Manning & Klein, ’03  Hall et al., ’09
Find the separating vector 

\[ w_{fa} - w_{fb} = 0 \]

\[ w_{fa} - w_{fb} > 0 \]

\[ w_{fa} - w_{fb} < 0 \]
Avoid Intractability

- Sample from the pool to avoid blowup
- Focus on difference vectors with large differences
- Add evil twins to ensure balance
Avoid Intractability

- Sample from the pool to avoid blowup
- Focus on difference vectors with large differences
- Add evil twins to ensure balance

(Nimoy, ’68)
MERT Tuning

MERT framework (Och, 2003)

Generate n-best per input sentence

Add to Candidate Pool

Learned weight vector

MERT Optimization

Candidate Generation

w:
Pairwise Ranking Optimization (PRO) Tuning

Candidate Generation

MERT framework

Pairwise Ranking Optimization

Learned weight vector

Generate n-best per input sentence

Add to Candidate Pool

Sample pairs
Unlike MERT, PRO is unfazed by a large number of features in the synthetic test.
Adding noise to the synthetic test makes it more difficult but PRO still does quite well compared to MERT.
MIRA also scales... but it’s **hard** to implement

- Like PRO, a discriminative learning algorithm
- Unlike PRO, requires online, simultaneous optimization and decoding
- MIRA tuning must be customized to compute environment (cluster, inter-process communication, reliability concerns)

(Watanabe et al., ’07)  (Chiang et al., ‘08, ‘09)
Unavoidable slide detailing the configuration and data of the experimental conditions...

<table>
<thead>
<tr>
<th>Language</th>
<th>Data (words)</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Train</td>
<td>Tune</td>
</tr>
<tr>
<td>Arabic-English</td>
<td>175M (NIST 2008)</td>
<td>65K (NIST 03-06/GALE)</td>
</tr>
<tr>
<td>Chinese-English</td>
<td>173M (GALE 2008)</td>
<td>42K (NIST 03-06)</td>
</tr>
</tbody>
</table>

Wednesday, August 3, 2011
Unavoidable slide detailing the configuration and data of the experimental conditions.

<table>
<thead>
<tr>
<th>Language</th>
<th>Data (words)</th>
<th>Features</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Arabic-English</td>
<td>175M (NIST 2008)</td>
<td>65K (NIST 03-06/GALE)</td>
<td>47K (NIST 2008)</td>
<td>15</td>
</tr>
<tr>
<td>Chinese-English</td>
<td>173M (GALE 2008)</td>
<td>42K (NIST 03-06)</td>
<td>37K (NIST 2008)</td>
<td>15</td>
</tr>
</tbody>
</table>
Unavoidable slide detailing the configuration and data of the experimental conditions....zzzzz

<table>
<thead>
<tr>
<th>Language</th>
<th>Data (words)</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Train</td>
<td>Tune</td>
</tr>
<tr>
<td>Urdu-English</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Arabic-English</td>
<td>175M</td>
<td>65K</td>
</tr>
<tr>
<td>Chinese-English</td>
<td>173M</td>
<td>42K</td>
</tr>
</tbody>
</table>

State-of-the-art decoders

Wednesday, August 3, 2011
Unavoidable slide detailing the configuration and data of the experimental conditions...

Two feature configurations per decoder

<table>
<thead>
<tr>
<th>Language</th>
<th>Train</th>
<th>Tune</th>
<th>Test</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Language</td>
<td></td>
<td></td>
<td>PBMT</td>
</tr>
<tr>
<td>Arabic-English</td>
<td>175M (NIST 2008)</td>
<td>65K (NIST 03-06/GALE)</td>
<td>47K (NIST 2008)</td>
<td>base 15</td>
</tr>
<tr>
<td>Chinese-English</td>
<td>173M (GALE 2008)</td>
<td>42K (NIST 03-06)</td>
<td>37K (NIST 2008)</td>
<td>base 15</td>
</tr>
</tbody>
</table>
Unavoidable slide detailing the configuration and data of the experimental conditions....zzzzz

<table>
<thead>
<tr>
<th>Language</th>
<th>Data (words)</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>PBMT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>base</td>
</tr>
<tr>
<td><strong>Ran MERT, MIRA, PRO</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Arabic-English</td>
<td>175M (NIST 2008)</td>
<td>65K (NIST 03-06/GALE)</td>
</tr>
<tr>
<td>Chinese-English</td>
<td>173M (GALE 2008)</td>
<td>42K (NIST 03-06)</td>
</tr>
</tbody>
</table>
Unavoidable slide detailing the configuration and data of the experimental conditions....zzzzz

<table>
<thead>
<tr>
<th>Language</th>
<th>Data (words)</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>PBMT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>base</td>
</tr>
<tr>
<td>Urdu-English</td>
<td>2.2M (NIST 2009)</td>
<td>16K</td>
</tr>
<tr>
<td></td>
<td>(NIST 2008)</td>
<td></td>
</tr>
<tr>
<td>Arabic-English</td>
<td>175M (NIST 2008)</td>
<td>65K</td>
</tr>
<tr>
<td></td>
<td>(NIST 03-06/GALE)</td>
<td></td>
</tr>
<tr>
<td>Chinese-English</td>
<td>173M (GALE 2008)</td>
<td>42K</td>
</tr>
<tr>
<td></td>
<td>(NIST 03-06)</td>
<td></td>
</tr>
</tbody>
</table>

**Ran MIRA, PRO (MERT doesn’t scale)**
Unavoidable slide detailing the configuration and data of the experimental conditions...

<table>
<thead>
<tr>
<th>Language</th>
<th>Data (words)</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Train</td>
<td>Tune</td>
</tr>
<tr>
<td></td>
<td>Train</td>
<td>Tune</td>
</tr>
<tr>
<td>Arabic-English</td>
<td>175M (NIST 2008)</td>
<td>65K (NIST 03-06/GALE)</td>
</tr>
<tr>
<td></td>
<td>Train</td>
<td>Tune</td>
</tr>
<tr>
<td>Chinese-English</td>
<td>173M (GALE 2008)</td>
<td>42K (NIST 03-06)</td>
</tr>
</tbody>
</table>

PBMT
- base: 15
- ext: 2250
- base: 15
- ext: 6333
- base: 15
- ext: 1828

SBMT
- base: 19
- ext: 277
- base: 19
- ext: 352
- base: 19
- ext: 517

Report 4-reference, detokenized, mixed-case BLEU

Wednesday, August 3, 2011
PRO is **fast**

*Your implementation of MIRA may be faster*
MERT is **unstable**

Result from five identical runs

(Clarck et al., 2011)
PRO is stable

Urdu-English PBMT tuning stability

Result from five identical runs

Wednesday, August 3, 2011
MERT vs. MIRA vs. PRO

PBMT Urdu-English

<table>
<thead>
<tr>
<th></th>
<th>MERT</th>
<th>MIRA</th>
<th>PRO</th>
<th>MERT</th>
<th>MIRA</th>
<th>PRO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>20.5</td>
<td>17.7</td>
<td>20.5</td>
<td>20.4</td>
<td>17.9</td>
<td>20.4</td>
</tr>
<tr>
<td>Extended</td>
<td>21.8</td>
<td>18.2</td>
<td>21.6</td>
<td>17.8</td>
<td>18.1</td>
<td>18.1</td>
</tr>
</tbody>
</table>

SBMT Urdu-English

<table>
<thead>
<tr>
<th></th>
<th>MERT</th>
<th>MIRA</th>
<th>PRO</th>
<th>MIRA</th>
<th>PRO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>23.4</td>
<td>23.6</td>
<td>23.4</td>
<td>22.3</td>
<td>22.2</td>
</tr>
<tr>
<td>Extended</td>
<td>25.2</td>
<td>24.2</td>
<td>25.2</td>
<td>22.8</td>
<td>22.8</td>
</tr>
</tbody>
</table>

- Red bars represent baseline features.
- Blue bars represent extended features.
MERT vs. MIRA vs. PRO

PBMT Arabic-English

<table>
<thead>
<tr>
<th></th>
<th>MERT</th>
<th>MIRA</th>
<th>PRO</th>
<th>MERT</th>
<th>MIRA</th>
<th>PRO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline features</td>
<td>46.8</td>
<td>47</td>
<td>46.9</td>
<td>47.5</td>
<td>48.5</td>
<td>41.2</td>
</tr>
<tr>
<td>Extended features</td>
<td>47</td>
<td>41.1</td>
<td>41.1</td>
<td>41.7</td>
<td>41.9</td>
<td>44.7</td>
</tr>
</tbody>
</table>

SBMT Arabic-English

<table>
<thead>
<tr>
<th></th>
<th>MERT</th>
<th>MIRA</th>
<th>PRO</th>
<th>MERT</th>
<th>MIRA</th>
<th>PRO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline features</td>
<td>39</td>
<td>39</td>
<td>39</td>
<td>39</td>
<td>39</td>
<td>39</td>
</tr>
<tr>
<td>Extended features</td>
<td>45.8</td>
<td>45.9</td>
<td>45.8</td>
<td>45.9</td>
<td>45.9</td>
<td>40.3</td>
</tr>
</tbody>
</table>

- baseline features
- extended features
PRO is comparable to all

PBMT Urdu-English

26
22
17
MERT MIRA PRO MIRA PRO

PBMT Arabic-English

49
44
39
MERT MIRA PRO MIRA PRO

PBMT Chinese-English

26
24
22
MERT MIRA PRO MIRA PRO

SBMT Urdu-English

26
22
17
MERT MIRA PRO MIRA PRO

SBMT Arabic-English

49
44
39
MERT MIRA PRO MIRA PRO

SBMT Chinese-English

26
24
22
MERT MIRA PRO MIRA PRO
PRO is comparable to all

- **PBMT Urdu-English**
  - MERT: 17, 22, 26
  - MIRA: 17, 22, 26
  - PRO: 17, 22, 26

- **SBMT Urdu-English**
  - MERT: 22, 26
  - MIRA: 22, 26
  - PRO: 22, 26

- **PBMT Arabic-English**
  - MERT: 39, 44, 49
  - MIRA: 39, 44, 49
  - PRO: 39, 44, 49

- **SBMT Arabic-English**
  - MERT: 39, 44, 49
  - MIRA: 39, 44, 49
  - PRO: 39, 44, 49

- **PBMT Chinese-English**
  - MERT: 22, 24, 26
  - MIRA: 22, 24, 26
  - PRO: 22, 24, 26

- **SBMT Chinese-English**
  - MERT: 22, 24, 26
  - MIRA: 22, 24, 26
  - PRO: 22, 24, 26
Related Work

**SampleRank**
(Culotta, ’08, Wick et al., ’09, Roth et al., ’10)

Similar approach, with guided search through pool space (See Haddow et al. in WMT)

**Classifier-based Weight Learning**
(Tillmann & Zhang, ’05, Och & Ney, ’02
Ittycheriah & Roukos, ’05, Xiong et al., ’06)

Various approaches using classifiers to learn MT feature weights -- these do not use the difference vector approach

**Discriminative Re-ranking**
(Shen et al., ’04, Cowan et al., ’06,
Watanabe et al., ’06)

Changing the n-best list after decoding using similar techniques to ours
Why Use **PRO**?
Why Use **PRO**?

It’s **scalable**
Why Use **PRO**?

It’s **scalable**

![Graph showing scalability comparison between PRO and MERT](image1)

It’s **stable**

![Graph showing stability comparison between PRO and MERT](image2)
Why Use **PRO**?

**It's scalable**

**It's stable**

**It's fast**
Why Use **PRO**?

- **It’s scalable**
- **It’s stable**
- **It’s fast**
- **It’s easy**

At least **three** external implementations prior to this talk.
Why Use **PRO**?

- **It’s scalable**
- **It’s fast**
- **It’s stable**
- **It’s easy**

**“Including mine!”**

(Dyer, P.C.)

https://github.com/redpony/cdec/tree/master/pro-train
Why Use PRO?

It's scalable

It's fast

It's stable

It's easy

"Including mine!"  
(Dyer, P.C.)

https://github.com/redpony/cdec/tree/master/pro-train