Tuning as Ranking
Pairwise Ranking Optimisation (PRO)
HOPKINS, M. & MAY, J.
2011
An SMT system translates from one human language to another

Such systems typically have a lot of parameters that need to be tuned
Current Tuning Solutions

- **MERT**
  - Well-understood, easy to implement, and runs quickly
  - Does not scale beyond a handful of features

- **MIRA**
  - Shown to perform well on large-scale tasks
  - Complex and architecturally different from MERT
Pairwise Ranking Optimisation (PRO)

- Adapts the MERT system
- Provides comparable performance to both
- Scales comparably to MIRA but is much simpler
- Should take about 2 hours to implement (supposedly)
Set-up
(Definitions!)
Candidate Space \( \langle \Delta, I, J, e, x \rangle \)

- \( \Delta \), the space’s **dimensionality** (a positive integer)
- \( I \), **sentence indices** (a set of positive integers)
- J maps
  - Each sentence index
  - To a set of **candidate indices** (positive integers)
Candidate Space \langle \Delta, \ I, \ J, \ e, \ x \rangle

- \ e(i, j) \ maps
  - Each pair \ (i, j) \in I \times J(i)
  - To the j^{th} target-language candidate translation of source sentence i

- \ x(i, j) \ maps
  - Each pair \ (i, j) \in I \times J(i)
  - To a \ \Delta\text{-dimension feature vector} \ representation of \ e(i, j)
Policy \( p(i) \)

- A function corresponding to a candidate space
- It maps
  - Each source sentence index \( (i \in I) \)
  - To a candidate sentence index \( (\in J(i)) \)
Scoring Function, \( h_w(i, j) = w \cdot x(i, j) \)

- Indicates how good candidate \( j \) is for source sentence \( i \)
- \( w \) is a weight vector that must be learnt
- Typically returns positive real numbers (higher \( \Rightarrow \) better)
- Can extend this idea to policy \( p \) by summing the costs of each candidate translation

\[
H_w(p) = \sum_{i \in I} h_w(i, p(i))
\]
A Gold Scoring Function, G

- An idealised equivalent of $H_w(p)$
- Maps
  - Each policy
  - To a real-valued score
- Typically calculated by a library, such as IBM Bleu
Goal of Tuning

- Goal is to find a weight vector $\mathbf{w}$
- For space $s$, we want a $\mathbf{w}$ that, equivalently
  - Gives an $H_\mathbf{w}$ which behaves “similarly” to $G$ on $s$
  - Minimises a loss function $l_s(H_\mathbf{w}, G)$
MERT
Two-Stage Feedback Loop

- **Candidate Generation**
  - Candidate translations are selected from a base candidate space $s$
  - Translations are added to the candidate pool, $s'$

- **Optimisation**
  - The weight vector $w$ is optimised to minimise a loss function $l_{s'}(H_w, G)$
  - Loss defined to prefer weight vectors such that the gold function $G$ scores $H_w$’s best policy as highly as possible (0 loss if equal to $G$’s best)
  - Implemented by line optimisation
Issues

- Does not scale well with dimensionality
- MERT optimisation focuses on $H_w$'s best policy, and not on its overall ability to rank policies
Pairwise Ranking Optimisation (PRO)
Assume the gold scoring function $G$ decomposes to:

$$G(p) = \sum_{i \in I} g(i, p(i))$$

Here, $g$ is a local scoring function
- It is equivalent to $h_w$ for $H_w$
- It can be used to rank candidate translations for each source sentence
Example

<table>
<thead>
<tr>
<th>Source Sentence</th>
<th>Candidate Translations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>i</strong></td>
<td><strong>j</strong></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>“he does not go”</td>
</tr>
<tr>
<td>3</td>
<td>“she not go”</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>“we do not go”</td>
</tr>
<tr>
<td>3</td>
<td>“I do not go”</td>
</tr>
</tbody>
</table>
The task is to classify candidate pairs, \( \langle e(i, j), e(i, j') \rangle \), into two categories:

- Correctly ordered (the first is better than the second)
- Incorrectly ordered (the second is better than the first)
Reframing the Learning Task with $g$

Thus, for a translations $e(i, j)$ and $e(i, j')$, we want $w$ such that

$$g(i, j) > g(i, j') \iff h_w(i, j) > h_w(i, j')$$

We can algebraically turn this into a binary classification problem!

$$g(i, j) > g(i, j') \iff h_w(i, j) > h_w(i, j')$$
$$\iff h_w(i, j) - h_w(i, j') > 0$$
$$\iff w \cdot x(i, j) - w \cdot x(i, j') > 0$$
$$\iff w \cdot (x(i, j) - x(i, j')) > 0$$
To Create Training Instances

1. Compute the difference vector $\mathbf{x}(i, j) - \mathbf{x}(i, j')$
2. Label it:
   - ‘Positive’ if the first vector is superior, according to $g$
   - ‘Negative’ if the second vector is superior, according to $g$

- Consider both difference vectors from a pair
- Randomly sample these vectors to create training data
Dimensional Scalability Evaluation
Set-up

1. Define $G = H_{w^*}(p)$ for some gold weight vector $w^*$
2. Generate a $\Delta$-dimensionality candidate pool
   - 500 source “sentences”, each with 100 candidate “translations”
   - Draw, at random, $\Delta$-dimensional feature vector values
3. Run the tuners
4. Repeat 1-3 with different $\Delta$ values
5. Repeat 1-4 with Gaussian noise added to feature vectors
Results

![Graph showing synthetic parameter learning of MERT and PRO](image)
Translation Evaluation
SBMT vs PBMT

- **Syntax-Based systems (SBMT)**
  - Based on the idea of translating syntactic units
  - Rather than single words or sequences of words

- **Phrase-Based systems (PBMT)**
  - Based on idea of translating whole sequences of words
  - Reduces the restrictions of word-based translation
  - The sequence lengths may differ
Evaluation Feature Sets

- **Baseline feature set**
  - Correspond to a typical small feature set in MT literature
  - Gives a low (around 20) dimensional candidate space

- **Extended feature set**
  - Only used with MIRA and PRO
  - Gives a high (thousands) dimensional candidate space
<table>
<thead>
<tr>
<th>Language</th>
<th>PBMT</th>
<th></th>
<th>SBMT</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>BLEU</td>
<td></td>
<td>BLEU</td>
</tr>
<tr>
<td></td>
<td>feats</td>
<td>tune  test</td>
<td>feats</td>
<td>tune  test</td>
</tr>
<tr>
<td>Urdu-English</td>
<td>base</td>
<td>MERT 20.5</td>
<td></td>
<td>MERT 23.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>17.7</td>
<td></td>
<td>21.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MIRA 20.5</td>
<td></td>
<td>MIRA 23.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>17.9</td>
<td></td>
<td>22.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRO 20.4</td>
<td></td>
<td>PRO 23.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>18.2</td>
<td></td>
<td>22.2</td>
</tr>
<tr>
<td></td>
<td>ext</td>
<td>MIRA 21.8</td>
<td></td>
<td>MIRA 25.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>17.8</td>
<td></td>
<td>22.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRO 21.6</td>
<td></td>
<td>PRO 24.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>18.1</td>
<td></td>
<td>22.8</td>
</tr>
<tr>
<td>Arabic-English</td>
<td>base</td>
<td>MERT 46.8</td>
<td></td>
<td>MERT 44.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>41.2</td>
<td></td>
<td>39.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MIRA 47.0</td>
<td></td>
<td>MIRA 44.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>41.1</td>
<td></td>
<td>39.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRO 46.9</td>
<td></td>
<td>PRO 44.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>41.1</td>
<td></td>
<td>39.0</td>
</tr>
<tr>
<td></td>
<td>ext</td>
<td>MIRA 47.5</td>
<td></td>
<td>MIRA 45.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>41.7</td>
<td></td>
<td>39.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRO 48.5</td>
<td></td>
<td>PRO 45.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>41.9</td>
<td></td>
<td>40.3</td>
</tr>
<tr>
<td>Chinese-English</td>
<td>base</td>
<td>MERT 23.8</td>
<td></td>
<td>MERT 25.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>22.2</td>
<td></td>
<td>22.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MIRA 24.1</td>
<td></td>
<td>MIRA 25.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>22.5</td>
<td></td>
<td>22.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRO 23.8</td>
<td></td>
<td>PRO 25.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>22.5</td>
<td></td>
<td>22.9</td>
</tr>
<tr>
<td></td>
<td>ext</td>
<td>MIRA 24.8</td>
<td></td>
<td>MIRA 26.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>22.6</td>
<td></td>
<td>23.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRO 24.9</td>
<td></td>
<td>PRO 25.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>22.7</td>
<td></td>
<td>23.5</td>
</tr>
</tbody>
</table>
Monotonicity
Summary
Successes of this Publication

- Thorough explanation of background and concepts
- Appears to perform comparably to contemporary systems
- Illustrates idea of mapping to a well-solved problem
- Surprisingly good results by solving an apparently simpler problem
- Source code not released, which is a pity
- Comparisons to alternative baselines might be interesting