Adaptive Quality Estimation for Machine Translation

Antonis Anastasopoulos
Advisors: Yanis Maistros\textsuperscript{1}, Marco Turchi\textsuperscript{2}, Matteo Negri\textsuperscript{2}

\textsuperscript{1}School of Electrical and Computer Engineering, NTUA, Greece
\textsuperscript{2}Fondazione Bruno Kessler, MT Group

April 9, 2014
OUTLINE

1 INTRODUCTION
   • Machine Translation
   • The Quality Estimation Task
   • Motivation

2 IMPLEMENTATION
   • System Overview
   • Machine Learning Component

3 EXPERIMENTS
   • General Framework
   • English-Spanish
   • English-Italian

4 CONCLUSION
   • Synopsis
OUTLINE

1 INTRODUCTION
   - Machine Translation
   - The Quality Estimation Task
   - Motivation

2 IMPLEMENTATION
   - System Overview
   - Machine Learning Component

3 EXPERIMENTS
   - General Framework
   - English-Spanish
   - English-Italian

4 CONCLUSION
   - Synopsis
1. **Introduction**
   - Machine Translation
   - The Quality Estimation Task
   - Motivation

2. **Implementation**
   - System Overview
   - Machine Learning Component

3. **Experiments**
   - General Framework
   - English-Spanish
   - English-Italian

4. **Conclusion**
   - Synopsis
OUTLINE

1 INTRODUCTION
   - Machine Translation
   - The Quality Estimation Task
   - Motivation

2 IMPLEMENTATION
   - System Overview
   - Machine Learning Component

3 EXPERIMENTS
   - General Framework
   - English-Spanish
   - English-Italian

4 CONCLUSION
   - Synopsis
Various approaches:

- Word-for-word translation
- Rule Based approach:

\[
\text{source} \xrightarrow{\text{transform}} \text{intermediate representation} \xrightarrow{\text{transform}} \text{target}
\]

- Interlingua
Various approaches:

- Word-for-word translation
- Rule Based approach:

\[
\text{source} \xrightarrow{\text{transform}} \text{intermediate representation} \xrightarrow{\text{transform}} \text{target}
\]

- Interlingua

Diagram:

- Source language analysis
- Semantic transfer
- Syntactic transfer
- Direct
- Target language generation

- Interlingua

Anastasopoulos

Online QE
Various approaches:

- Word-for-word translation
- Rule Based approach:
  
  \[
  \text{source} \xrightarrow{\text{transform}} \text{intermediate representation} \xrightarrow{\text{transform}} \text{target}
  \]

- Interlingua
**Statistical MT**

Given a foreign language $\mathcal{F}$ and a sentence $f$, find the most probable sentence $\hat{s}$ in the translation target language $S$, out of all possible translations $s$.

$$\hat{s} = \arg \max_s p(s|f)$$

From the Bayes rule:

$$\hat{s} = \arg \max_s p(s)p(f|s)$$
Statistical MT

Given a foreign language $\mathcal{F}$ and a sentence $f$, find the most probable sentence $\hat{s}$ in the translation target language $S$, out of all possible translations $s$.

$$\hat{s} = \arg \max_s p(s|f)$$

From the Bayes rule:

$$\hat{s} = \arg \max_s p(s)p(f|s)$$
Statistical MT

Given a foreign language $\mathcal{F}$ and a sentence $f$, find the most probable sentence $\hat{s}$ in the translation target language $\mathcal{S}$, out of all possible translations $s$.

\[ \hat{s} = \arg \max_s p(s|f) \]

From the Bayes rule:

\[ \hat{s} = \arg \max_s p(s)p(f|s) \]
MT Evaluation

- Reference-based: BLEU, NIST, Meteor
  
  \((\text{Modifications of ML precision or recall})\)

- Metrics of Post-Editing Effort:
  - Human Annotations
  - Post-Editing time
  - Human Translation Edit Rate (\(HTER\))

\[
HTER = \frac{\text{#edits}}{\text{#postedited words}}
\]

edits = insertions, deletions, substitutions, shifts
MT Evaluation

- Reference-based: BLEU, NIST, Meteor
  *(Modifications of ML precision or recall)*
- Metrics of Post-Editing Effort:
  - Human Annotations
  - Post-Editing time
  - Human Translation Edit Rate (*HTER*)

\[
HTER = \frac{\#\text{edits}}{\#\text{postedited words}}
\]

edits = insertions, deletions, substitutions, shifts
MT Evaluation

- Reference-based: BLEU, NIST, Meteor
  *(Modifications of ML precision or recall)*

- Metrics of Post-Editing Effort:
  - Human Annotations
  - Post-Editing time
  - Human Translation Edit Rate (*HTER*)

  \[
  HTER = \frac{\#\text{edits}}{\#\text{postedited words}}
  \]

  edits = insertions, deletions, substitutions, shifts
MT Evaluation

- Reference-based: BLEU, NIST, Meteor

*(Modifications of ML precision or recall)*

- Metrics of Post-Editing Effort:
  - Human Annotations
  - Post-Editing time
  - Human Translation Edit Rate (HTER)

\[
HTER = \frac{\#\text{edits}}{\#\text{postedited words}}
\]

edits = insertions, deletions, substitutions, shifts
MT Evaluation

- Reference-based: BLEU, NIST, Meteor
  
  \((\text{Modifications of ML precision or recall})\)

- Metrics of Post-Editing Effort:
  - Human Annotations
  - Post-Editing time
  - Human Translation Edit Rate (HTER)

  \[
  HTER = \frac{\#\text{edits}}{\#\text{postedited words}}
  \]

  edits = insertions, deletions, substitutions, shifts
source:

Because I also have a penchant for tradition, manners and customs.

produced translation:

Porque también tengo una inclinación por tradición, modales y costumbres.

post-edited:

Porque también tengo una inclinación por la tradición, los modales y las costumbres.

\[ HTER = \frac{3}{15} = 0.20 \]
1. **Introduction**
   - Machine Translation
   - The Quality Estimation Task
   - Motivation

2. **Implementation**
   - System Overview
   - Machine Learning Component

3. **Experiments**
   - General Framework
   - English-Spanish
   - English-Italian

4. **Conclusion**
   - Synopsis
The QE task

Definition

The task of estimating the quality of a system’s output for a given input, without information about the expected output.

- Initially a classification task: “good” and “bad” translations
- Now a regression task: Quality score (e.g. HTER)
- Evaluation campaigns @WMT
- Current focus on feature engineering
THE QE TASK

DEFINITION

The task of estimating the quality of a system’s output for a given input, without information about the expected output.

- Initially a classification task: “good” and “bad” translations
- Now a regression task: Quality score (eg. HTER)
- Evaluation campaigns @WMT
- Current focus on feature engineering
THE QE TASK

DEFINITION

The task of estimating the quality of a system’s output for a given input, without information about the expected output.

- Initially a classification task: “good” and “bad” translations
- Now a regression task: Quality score (eg. HTER)
- Evaluation campaigns @WMT
- Current focus on feature engineering
Machine Translation
The Quality Estimation Task

Motivation

THE QE TASK

Definition

The task of estimating the quality of a system’s output for a given input, without information about the expected output.

- Initially a classification task: “good” and “bad” translations
- Now a regression task: Quality score (eg. HTER)
- Evaluation campaigns @WMT
- Current focus on feature engineering
Connection with Industry

Vanilla CAT Tool

But I must explain to you how all this mistaken idea of denouncing pleasure and praising pain was born and I will give you a complete account of the system, and expound the actual teachings of the great explorer of the truth, the master builder of human happiness.

No one rejects, dislikes, or avoids pleasure itself, because it is pleasure...

Terminology is automatically extracted from the MT phrase-table (self-tuning and informative MT).

Collaboration between customers, translators and MT provider.
CAT-tool Scenario

CAT: Computer Assisted Translation

Source Segments

Statistical Machine Translation System

Translator

Target Segments

Post-Edited Segments

Anastasopoulos

Online QE
CAT-tool Scenario

- Source Segments
- Statistical Machine Translation System
- Target Segments
- Quality Estimation System
- Online QE
CAT-tool Scenario

Source Segments -> Statistical Machine Translation System -> Target Segments

Translator -> Post-Edited Segments

Prediction -> Quality Estimation System
CAT-tool Scenario

- Source Segments
  - Statistical Machine Translation System
    - Target Segments
      - Translator
        - Prediction
          - Quality Estimation System
            - Post-Edited Segments

Motivation

- Machine Translation
- The Quality Estimation Task

Conclusion

- Anastasopoulos
- Online QE
CAT-tool Scenario

Source Segments

Statistical Machine Translation System

Target Segments

Translator

Prediction

Quality Estimation System

Post-Edited Segments

Why Online?
Table of Contents

1 Introduction
   - Machine Translation
   - The Quality Estimation Task
   - Motivation

2 Implementation
   - System Overview
   - Machine Learning Component

3 Experiments
   - General Framework
   - English-Spanish
   - English-Italian

4 Conclusion
   - Synopsis
GOAL: Increase the productivity of the translator

This can be done by:

- Increasing the quality of the translations provided by the SMT systems
- Providing the translator with information about the quality of the suggested translations

In this direction...

- Small amount of data
  - How much data do we need for good quality predictions?
- Notion of quality is subjective
  - Can we adapt to an individual user?
- Different translation jobs
  - Can we adapt to domain changes?
Motivation and Open Questions

**GOAL: Increase the productivity of the translator**

This can be done by:

- Providing the translator with information about the quality of the suggested translations

In this direction...

- Small amount of data
  - How much data do we need for good quality predictions?
- Notion of quality is subjective
  - Can we adapt to an individual user?
- Different translation jobs
  - Can we adapt to domain changes?
Table of Contents

1 Introduction
   - Machine Translation
   - The Quality Estimation Task
   - Motivation

2 Implementation
   - System Overview
   - Machine Learning Component

3 Experiments
   - General Framework
   - English-Spanish
   - English-Italian

4 Conclusion
   - Synopsis
**System Overview**

Diagram showing the flow of data between components:

- **Quality Estimation System**
  - receive <src><trg>
  - send <src><trg>
  - receive Features
  - send Features
  - receive Prediction
  - send Prediction
  - receive <pe>
  - send <trg><pe>
  - receive HTER
  - send true HTER

- **Online Learning Algorithm**
  - Receive Confirmation that Training is Finished
Table of Contents

1 Introduction
   • Machine Translation
   • The Quality Estimation Task
   • Motivation

2 Implementation
   • System Overview
   • Machine Learning Component

3 Experiments
   • General Framework
   • English-Spanish
   • English-Italian

4 Conclusion
   • Synopsis
Learning Algorithms

- Online SVR
- Passive-Aggressive Alg.
- Sparse Online Gaussian Processes
**Definition**

Given a training set \( \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \subset X \times \mathbb{R} \) of \( n \) training points, were \( x_i \) is a vector of dimensionality \( d \) (so \( X = \mathbb{R}^d \)), and \( y_i \in \mathbb{R} \) is the target, find a hyperplane (function) \( f(x) \) that has at most \( \epsilon \) deviation from the target \( y_i \), and at the same time it is as flat as possible.
Support Vector Regression

Linear regression function:

\[ f(x) = W^T \Phi(x) + b \]

Convex optimization problem by requiring:

\[
\minimize \frac{1}{2} \|W\|^2 \\
\text{subject to } \begin{cases} 
  y_i - W^T \Phi(x) - b \leq \epsilon \\
  W^T \Phi(x) + b - y_i \leq \epsilon
\end{cases}
\]

Solution found through the dual optimization problem, using a kernel function, as long as the KKT conditions hold.
Online Support Vector Regression

- Idea: update the coefficient of the margin of the new sample \( x_c \) in a finite number of steps until it meets the KKT conditions.
- In the same time it must be ensured that also the rest of the existing samples continue to satisfy the KKT conditions.
Passive-Aggressive Algorithms

- Same idea as SVR: $\epsilon$-insensitive loss function that creates a hyper-slab of width $2\epsilon$
- Update:

$$l_\epsilon \mathbf{W}; (x, y) = \begin{cases} 0, & \text{if } |\mathbf{W} \cdot x - y| \leq \epsilon \\ |\mathbf{W} \cdot x - y| - \epsilon, & \text{otherwise} \end{cases}$$

- Passive: if $l_\epsilon$ is 0, $\mathbf{W}_{t+1} = \mathbf{W}_t$.
- Aggressive: if $l_\epsilon$ is not 0, $\mathbf{W}_{t+1} = \mathbf{W}_t + \text{sign}(y_t - \hat{y}_t) T_t x_t$, where $T_t = \min(C, \frac{l_t}{\|x_t\|^2})$. 
Passive-Aggressive Algorithms

- Same idea as SVR: $\epsilon$-insensitive loss function that creates a hyper-slab of width $2\epsilon$
- Update:

$$l_\epsilon W; (x, y) = \begin{cases} 0, & \text{if } |W \cdot x - y| \leq \epsilon \\ |W \cdot x - y| - \epsilon, & \text{otherwise} \end{cases}$$

- Passive: if $l_\epsilon$ is 0, $W_{t+1} = W_t$.
- Aggressive: if $l_\epsilon$ is not 0, $W_{t+1} = W_t + \text{sign}(y_t - \hat{y_t}) T_t x_t$, where $T_t = \min(C, \frac{l_t}{\|x_t\|^2})$. 
PASSIVE-AGGRESSIVE ALGORITHMS

- Same idea as SVR: $\epsilon$-insensitive loss function that creates a hyper-slab of width $2\epsilon$
- Update:

$$l_\epsilon W; (x, y) = \begin{cases} 0, & \text{if } |W \cdot x - y| \leq \epsilon \\ |W \cdot x - y| - \epsilon, & \text{otherwise} \end{cases}$$

- Passive: if $l_\epsilon$ is 0, $W_{t+1} = W_t$.
- Aggressive: if $l_\epsilon$ is not 0, $W_{t+1} = W_t + \text{sign}(y_t - \hat{y}_t) T_t x_t$, where $T_t = \min(C, \frac{l_t}{\|x_t\|^2})$. 

Anastasopoulos Online QE
Passive-Aggressive Algorithms

- Same idea as SVR: $\epsilon$-insensitive loss function that creates a hyper-slab of width $2\epsilon$
- Update:

$$l_\epsilon W; (x, y) = \begin{cases} 0, & \text{if } |W \cdot x - y| \leq \epsilon \\ |W \cdot x - y| - \epsilon, & \text{otherwise} \end{cases}$$

- Passive: if $l_\epsilon$ is 0, $W_{t+1} = W_t$.
- Aggressive: if $l_\epsilon$ is not 0, $W_{t+1} = W_t + \text{sign}(y_t - \hat{y}_t) T_t x_t$, where $T_t = \min(C, \frac{l_t}{\|x_t\|^2})$. 
**Gaussian Processes**

**Definition**

...a collection of random variables, any finite number of which have a joint Gaussian distribution (Rasmussen 2006)

Any Gaussian Process can be completely defined by its mean function $m(x)$ and the covariance function $k(x, x')$: 

$$ \mathcal{GP}(m(x), k(x, x')) $$

The Gaussian Process assumes that every target $y_i$ is generated from the corresponding data $x_i$ and an added white noise $\eta$ as:

$$ y_i = f(x_i) + \eta, \text{ where } \eta \sim \mathcal{N}(0, \sigma_n^2) $$

This function $f(x)$ is drawn from a GP prior:

$$ f(x) \sim \mathcal{GP}(m(x), k(x, x')) $$

where the covariance is encoded using the kernel function $k(x, x')$. 

Anastasopoulos Online QE
Any Gaussian Process can be completely defined by its mean function \( m(x) \) and the covariance function \( k(x, x') \):

\[
\mathcal{GP}(m(x), k(x, x')).
\]

The Gaussian Process assumes that every target \( y_i \) is generated from the corresponding data \( x_i \) and an added white noise \( \eta \) as:

\[
y_i = f(x_i) + \eta, \quad \text{where} \quad \eta \sim \mathcal{N}(0, \sigma^2_n)
\]

This function \( f(x) \) is drawn from a GP prior:

\[
f(x) \sim \mathcal{GP}(m(x), k(x, x')).
\]

where the covariance is encoded using the kernel function \( k(x, x') \).
Using RBF kernel and *automatic relevance determination* kernel, smoothness of the functions can be encoded. Current state-of-the-art for regression and QE.

Online GPs (Csato and Opper, 2002):

- *Basis Vector set* \( \mathcal{B} \mathcal{V} \) with pre-defined capacity.
- Online update based on properties of Gaussian distribution.
Online Gaussian Processes

Using RBF kernel and *automatic relevance determination* kernel, smoothness of the functions can be encoded. Current state-of-the-art for regression and QE. Online GPs (Csato and Opper, 2002):

- *Basis Vector set* $B\mathcal{V}$ with pre-defined capacity.
- Online update based on properties of Gaussian distribution.
Basic Features

We use 17 features. Indicatively:

- source and target sentence length (in tokens)
- source and target sentence 3-gram language model probabilities and perplexities
- average source word length
- percentage of 1 to 3-grams in the source sentence belonging to each frequency quartile of a monolingual corpus
- number of mismatching opening/closing brackets and quotation marks in the target sentence
- number of punctuation marks in the source and target sentences
- average number of translations per source word in the sentence (as given by IBM 1 table thresholded so that \( \text{prob}(t|s) > 0.2) \)
TABLE OF CONTENTS

1 INTRODUCTION
   • Machine Translation
   • The Quality Estimation Task
   • Motivation

2 IMPLEMENTATION
   • System Overview
   • Machine Learning Component

3 EXPERIMENTS
   • General Framework
   • English-Spanish
   • English-Italian

4 CONCLUSION
   • Synopsis
We compare:
- the *adaptive* approach (for all online algorithms)
- the *batch* approach, implemented with simple SVR
- the *empty* adaptive approach, starting with an empty model without training.

Performance measured with Mean Absolute Error (MAE)

\[
MAE = \frac{\sum_{i=1}^{n} |\hat{y}_i - y_i|}{n}
\]
We compare:

- the *adaptive* approach (for all online algorithms)
- the *batch* approach, implemented with simple SVR
- the *empty* adaptive approach, starting with an empty model without training.

Performance measured with Mean Absolute Error (MAE)

\[
\text{MAE} = \frac{\sum_{i=1}^{n} |\hat{y}_i - y_i|}{n}
\]
**EXPERIMENT FRAMEWORK**

We compare:

- the *adaptive* approach (for all online algorithms)
- the *batch* approach, implemented with simple *SVR*
- the *empty* adaptive approach, starting with an empty model without training.

Performance measured with Mean Absolute Error (MAE)

\[
MAE = \frac{\sum_{i=1}^{n} |\hat{y}_i - y_i|}{n}
\]
1 INTRODUCTION
   • Machine Translation
   • The Quality Estimation Task
   • Motivation

2 IMPLEMENTATION
   • System Overview
   • Machine Learning Component

3 EXPERIMENTS
   • General Framework
   • English-Spanish
   • English-Italian

4 CONCLUSION
   • Synopsis
**Data from WMT-2012 (2254 instances)**

- Shuffled and split into:
  - **TRAIN** (first 1500 instances)
  - **TEST** (last 754 instances)

- 3 sub-experiments:
  - Train on 200 instances
  - Train on 600 instances
  - Train on 1500 instances

<table>
<thead>
<tr>
<th>Training</th>
<th>Avg. HTER</th>
<th>St. Dev.</th>
<th>Test Labels</th>
<th>Avg. HTER</th>
<th>St. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>32.71</td>
<td>14.99</td>
<td></td>
<td>32.32</td>
<td>17.32</td>
</tr>
<tr>
<td>600</td>
<td>33.64</td>
<td>16.72</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1500</td>
<td>33.54</td>
<td>18.56</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**En-Es Data (Experiment 1)**

- Data from WMT-2012 (2254 instances)
- Shuffled and split into:
  - TRAIN (first 1500 instances)
  - TEST (last 754 instances)
- GridSearch with 10-fold Cross Validation for optimization of the initial parameters
- 3 sub-experiments:
  - Train on 200 instances
  - Train on 600 instances
  - Train on 1500 instances

<table>
<thead>
<tr>
<th>Training Labels</th>
<th>Test Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Training</strong></td>
<td><strong>Avg. HTER</strong></td>
</tr>
<tr>
<td>200</td>
<td>32.71</td>
</tr>
<tr>
<td>600</td>
<td>33.64</td>
</tr>
<tr>
<td>1500</td>
<td>33.54</td>
</tr>
</tbody>
</table>
**EN-ES Data (Experiment 1)**

- Data from WMT-2012 (2254 instances)
- Shuffled and split into:
  - TRAIN (first 1500 instances)
  - TEST (last 754 instances)
- 3 sub-experiments:
  - Train on 200 instances
  - Train on 600 instances
  - Train on 1500 instances

<table>
<thead>
<tr>
<th>Training</th>
<th>Avg. HTER</th>
<th>St. Dev.</th>
<th>Test Labels</th>
<th>Avg. HTER</th>
<th>St. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>32.71</td>
<td>14.99</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>600</td>
<td>33.64</td>
<td>16.72</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1500</td>
<td>33.54</td>
<td>18.56</td>
<td></td>
<td>32.32</td>
<td>17.32</td>
</tr>
</tbody>
</table>
**EN-ES Data (Experiment 1)**

- Data from WMT-2012 (2254 instances)
- Shuffled and split into:
  - TRAIN (first 1500 instances)
  - TEST (last 754 instances)
- 3 sub-experiments:
  - Train on 200 instances
  - Train on 600 instances
  - Train on 1500 instances

<table>
<thead>
<tr>
<th>Training</th>
<th>Avg. HTER</th>
<th>St. Dev.</th>
<th>Test Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>32.71</td>
<td>14.99</td>
<td></td>
</tr>
<tr>
<td>600</td>
<td>33.64</td>
<td>16.72</td>
<td>32.32</td>
</tr>
<tr>
<td>1500</td>
<td>33.54</td>
<td>18.56</td>
<td>17.32</td>
</tr>
</tbody>
</table>
# Results for Experiment 1

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Kernel</th>
<th>MAE ((i = 200))</th>
<th>MAE ((i = 600))</th>
<th>MAE ((i = 1500))</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Batch</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(SVR_i)</td>
<td>Linear</td>
<td>13.5</td>
<td>13.0</td>
<td>12.8</td>
</tr>
<tr>
<td></td>
<td>RBF</td>
<td>13.2*</td>
<td>12.7*</td>
<td>12.7*</td>
</tr>
<tr>
<td><strong>Adaptive</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(OSVR_i)</td>
<td>Linear</td>
<td>13.2*</td>
<td>12.9</td>
<td>12.8</td>
</tr>
<tr>
<td></td>
<td>RBF</td>
<td>13.6</td>
<td>13.7</td>
<td>13.5</td>
</tr>
<tr>
<td>(PA_i)</td>
<td>-</td>
<td>14.0</td>
<td>13.4</td>
<td>13.3</td>
</tr>
<tr>
<td>(OGP_i)</td>
<td>RBF</td>
<td>13.2*</td>
<td>12.9</td>
<td>12.8</td>
</tr>
</tbody>
</table>

Anastasopoulos Online QE
## Results for Experiment 1

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Kernel</th>
<th>MAE ((i = 200))</th>
<th>MAE ((i = 600))</th>
<th>MAE ((i = 1500))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{Empty})</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\text{OSVR}_0)</td>
<td>Linear RBF</td>
<td></td>
<td>13.5</td>
<td>13.7</td>
</tr>
<tr>
<td>(\text{PA}_0)</td>
<td></td>
<td></td>
<td></td>
<td>14.4</td>
</tr>
<tr>
<td>(\text{OGP}_0)</td>
<td>RBF</td>
<td></td>
<td></td>
<td>13.3</td>
</tr>
</tbody>
</table>
**TIME PERFORMANCE AND COMPLEXITY**

![Graph showing time performance and complexity](image)

Legend:
- OSVR
- PA
- OGP

- Training instances
- Time (ms)
Time performance and complexity

Given a number of seen samples $n$ and a number of features $f$ for each sample, the computational complexity of updating a trained model with a new instance is:

- $O(n^2 f)$ for training standard (not online) Support Vector Machines.
- $O(n^3 f)$ (average case: $O(n^2 f)$) for updating a trained model with OSVR.
- $O(f)$ for the Passive-Aggressive algorithm.
- $O(nd^2 f)$ (on run-time: $\Theta(nd^2 f)$) for an Online GP method with bounded $BV$ vector with maximum capacity $d$, where $\hat{d}$ is the actual number of vectors in the $BV$ vector.
**EN-ES Data (Experiment 2)**

- Data from WMT-2012 (2254 instances)
- Sorted according to the label and split into:
  - *Bottom* (first 600 instances)
  - *Top* (last 600 instances)
- 2 sub-experiments:
  - Train on *Bottom*, test on *Top*
  - Train on *Top*, test on *Bottom*.

<table>
<thead>
<tr>
<th>Set</th>
<th>Average HTER</th>
<th>HTER St. Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top</td>
<td>56.27</td>
<td>12.59</td>
</tr>
<tr>
<td>Bottom</td>
<td>12.35</td>
<td>6.43</td>
</tr>
</tbody>
</table>
**En-Es Data (Experiment 2)**

- Data from WMT-2012 (2254 instances)
- Sorted according to the label and split into:
  - *Bottom* (first 600 instances)
  - *Top* (last 600 instances)
- 2 sub-experiments:
  - Train on *Bottom*, test on *Top*
  - Train on *Top*, test on *Bottom*.

<table>
<thead>
<tr>
<th>Set</th>
<th>Average HTER</th>
<th>HTER St. Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top</td>
<td>56.27</td>
<td>12.59</td>
</tr>
<tr>
<td>Bottom</td>
<td>12.35</td>
<td>6.43</td>
</tr>
</tbody>
</table>
**En-Es Data (Experiment 2)**

- Data from WMT-2012 (2254 instances)
- Sorted according to the label and split into:
  - *Bottom* (first 600 instances)
  - *Top* (last 600 instances)
- 2 sub-experiments:
  - Train on *Bottom*, test on *Top*
  - Train on *Top*, test on *Bottom*.

<table>
<thead>
<tr>
<th>Set</th>
<th>Average HTER</th>
<th>HTER St. Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Top</em></td>
<td>56.27</td>
<td>12.59</td>
</tr>
<tr>
<td><em>Bottom</em></td>
<td>12.35</td>
<td>6.43</td>
</tr>
</tbody>
</table>
### Results for Experiment 2

<table>
<thead>
<tr>
<th>Test on Top</th>
<th>Test on Bottom</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Algorithm</strong></td>
<td><strong>Kernel</strong></td>
</tr>
<tr>
<td><strong>Batch</strong></td>
<td></td>
</tr>
<tr>
<td>$SVR_{Top}^{Bottom}$</td>
<td>Linear</td>
</tr>
<tr>
<td>RBF</td>
<td>43.2</td>
</tr>
<tr>
<td><strong>Adaptive</strong></td>
<td></td>
</tr>
<tr>
<td>$OSVR_{Top}^{Bottom}$</td>
<td>Linear</td>
</tr>
<tr>
<td>RBF</td>
<td>31.1</td>
</tr>
<tr>
<td>$PA_{Top}^{Bottom}$</td>
<td>-</td>
</tr>
<tr>
<td>$OGP_{Top}^{Bottom}$</td>
<td>RBF</td>
</tr>
</tbody>
</table>

Anastasopoulos Online QE
## Results for Experiment 2

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Kernel</th>
<th>MAE on Top</th>
<th>MAE on Bottom</th>
</tr>
</thead>
<tbody>
<tr>
<td>Empty</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$OSVR_0$</td>
<td>Linear</td>
<td>8.42</td>
<td>5.67</td>
</tr>
<tr>
<td></td>
<td>RBF</td>
<td>8.55</td>
<td>5.37</td>
</tr>
<tr>
<td>$PA_0$</td>
<td>-</td>
<td>8.37</td>
<td>5.30</td>
</tr>
<tr>
<td>$OGP_0$</td>
<td>RBF</td>
<td>8.83</td>
<td>5.22</td>
</tr>
</tbody>
</table>

Anastasopoulos | Online QE
Table of Contents

1 Introduction
   - Machine Translation
   - The Quality Estimation Task
   - Motivation

2 Implementation
   - System Overview
   - Machine Learning Component

3 Experiments
   - General Framework
   - English-Spanish
   - English-Italian

4 Conclusion
   - Synopsis
EN-IT DATA

- Data from a Field-Test @FBK (2012)
- Two domains: IT and Legal
- Same document for each domain: 4 Translators
  - 280 sentences for IT dataset
  - 160 sentences for Legal dataset
- Split into:
  - TRAIN: Day 1 of Field Test
  - TEST: Day 2 of Field Test
- All combinations of translators
Modelling Translator Behaviour

We rank translator pairs and compare:

- Average HTER
- Common vocabulary size
- Common n-grams percentage
- Average overlap
- Distribution difference (Hellinger distance)
- Reordering (Kendall’s $\tau$ metric)
- Instance-wise Difference

HTER correlates better with all the other possible metrics.
We rank translator pairs and compare:

- Average HTER
- Common vocabulary size
- Common n-grams percentage
- Average overlap
- Distribution difference (Hellinger distance)
- Reordering (Kendall’s $\tau$ metric)
- Instance-wise Difference

HTER correlates better with all the other possible metrics.
Legal domain:

<table>
<thead>
<tr>
<th>Post-editor</th>
<th>Avg HTER</th>
<th>HTER St. Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>29.04</td>
<td>16.84</td>
</tr>
<tr>
<td>2</td>
<td>32.33</td>
<td>18.87</td>
</tr>
<tr>
<td>3</td>
<td>43.25</td>
<td>14.86</td>
</tr>
<tr>
<td>4</td>
<td>23.52</td>
<td>15.80</td>
</tr>
</tbody>
</table>
## Translator Behaviour

**IT domain:**

<table>
<thead>
<tr>
<th>Post-editor</th>
<th>Avg HTER</th>
<th>HTER St. Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>39.32</td>
<td>21.03</td>
</tr>
<tr>
<td>2</td>
<td>47.77</td>
<td>20.49</td>
</tr>
<tr>
<td>3</td>
<td>37.72</td>
<td>20.05</td>
</tr>
<tr>
<td>4</td>
<td>36.60</td>
<td>19.71</td>
</tr>
</tbody>
</table>
In-domain Results

In general:

- When post-editors behave similarly, eg. (IT 1,3), *batch* and *adaptive* both work well.
- When post-editors are more different, eg (IT 3,2 or L 3,4), the *adaptive* approach significantly outperforms *batch*.

Learning Algorithm comparison:

- *OnlineGP* $>>$ *OnlineSVR* $>>$ *PA*

Algorithms perform well also in *Empty* mode.
In-domain Results

In general:

- When post-editors behave similarly, eg. (IT 1,3), batch and adaptive both work well.
- When post-editors are more different, eg (IT 3,2 or L 3,4), the adaptive approach significantly outperforms batch.

Learning Algorithm comparison:

- OnlineGP $>>$ OnlineSVR $>>$ PA

Algorithms perform well also in Empty mode.
IT domain

Cumulative MAE

Translator 1

Translator 2

Translator 3

Translator 4
We select the most different translators from each domain (Low, High).
8 combinations:

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Training Set</th>
<th>Test Set</th>
<th>HTER Diff.</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Low,L</td>
<td>High,IT</td>
<td>24.5</td>
</tr>
<tr>
<td>4.2</td>
<td>High,IT</td>
<td>Low,L</td>
<td>24</td>
</tr>
<tr>
<td>4.3</td>
<td>Low,IT</td>
<td>Low,L</td>
<td>13.5</td>
</tr>
<tr>
<td>4.4</td>
<td>Low,L</td>
<td>Low,IT</td>
<td>12.7</td>
</tr>
<tr>
<td>4.5</td>
<td>Low,IT</td>
<td>High,L</td>
<td>8.3</td>
</tr>
<tr>
<td>4.6</td>
<td>High,L</td>
<td>High,IT</td>
<td>6.8</td>
</tr>
<tr>
<td>4.7</td>
<td>High,L</td>
<td>Low,IT</td>
<td>5</td>
</tr>
<tr>
<td>4.8</td>
<td>High,IT</td>
<td>High,L</td>
<td>2.2</td>
</tr>
</tbody>
</table>
### Experiments

<table>
<thead>
<tr>
<th>Exp.</th>
<th>HTER Diff.</th>
<th>MAE Batch</th>
<th>MAE Adaptive</th>
<th>MAE Empty</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>24.5</td>
<td>27.00</td>
<td>19.77</td>
<td>16.55</td>
</tr>
<tr>
<td>4.2</td>
<td>24.0</td>
<td>25.37</td>
<td>19.96</td>
<td>12.46</td>
</tr>
<tr>
<td>4.3</td>
<td>13.5</td>
<td>17.54</td>
<td>15.73</td>
<td>12.46</td>
</tr>
<tr>
<td>4.4</td>
<td>12.7</td>
<td>17.58</td>
<td>15.50</td>
<td>15.45</td>
</tr>
<tr>
<td>4.5</td>
<td>8.3</td>
<td>13.00</td>
<td>10.51</td>
<td>11.28</td>
</tr>
<tr>
<td>4.6</td>
<td>6.8</td>
<td>16.89</td>
<td>16.38</td>
<td>16.55</td>
</tr>
<tr>
<td>4.7</td>
<td>5.0</td>
<td>16.15</td>
<td>14.40</td>
<td>15.45</td>
</tr>
<tr>
<td>4.8</td>
<td>2.2</td>
<td>10.84</td>
<td>10.64</td>
<td>11.28</td>
</tr>
</tbody>
</table>

#### Correlation of performance and hter difference:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>batch</td>
<td>0.945</td>
</tr>
<tr>
<td>adaptive</td>
<td>0.812</td>
</tr>
<tr>
<td>empty</td>
<td>0.190</td>
</tr>
</tbody>
</table>
Discussion:

- *Adaptive* approaches perform significantly better even with change in user or domain.
- *Batch* approaches are only good when post-editing behaviour is the same between train and test.
- *Empty* adaptive models also achieve outstanding results with very little data.

Learning Algorithms comparison:

- *OSVR* and *OGP* are more robust to domain and user change than *PA*.
Discussion:

- *Adaptive* approaches perform significantly better even with change in user or domain.
- *Batch* approaches are only good when post-editing behaviour is the same between train and test.
- *Empty* adaptive models also achieve outstanding results with very little data.

Learning Algorithms comparison:

- *OSVR* and *OGP* are more robust to domain and user change than *PA*.
# Table of Contents

1. **Introduction**
   - Machine Translation
   - The Quality Estimation Task
   - Motivation

2. **Implementation**
   - System Overview
   - Machine Learning Component

3. **Experiments**
   - General Framework
   - English-Spanish
   - English-Italian

4. **Conclusion**
   - Synopsis
We introduce the use of *online* learning techniques for the QE task.

- We show that they can deal with data scarcity and user and domain change, better than *batch* approaches.
- The AQET (Adaptive QE Tool) is suitable for commercial use and will be integrated into the MateCat-tool.

*Default alg: Online GP with RBF kernel*

- The code is available in [https://bitbucket.org/antonis/adaptiveqe](https://bitbucket.org/antonis/adaptiveqe).
We introduce the use of online learning techniques for the QE task.

We show that they can deal with data scarcity and user and domain change, better than batch approaches.

The AQET (Adaptive QE Tool) is suitable for commercial use and will be integrated into the MateCat-tool.

Default alg: Online GP with RBF kernel

The code is available in https://bitbucket.org/antonis/adaptiveqe.
We introduce the use of *online* learning techniques for the QE task.

We show that they can deal with data scarcity and user and domain change, better than *batch* approaches.

The *AQET* (Adaptive QE Tool) is suitable for commercial use and will be integrated into the MateCat-tool. *Default alg: Online GP with RBF kernel*

The code is available in https://bitbucket.org/antonis/adaptiveqe.
We introduce the use of online learning techniques for the QE task.

We show that they can deal with data scarcity and user and domain change, better than batch approaches.

The AQET (Adaptive QE Tool) is suitable for commercial use and will be integrated into the MateCat-tool.

*Default alg: Online GP with RBF kernel*

The code is available in [https://bitbucket.org/antonis/adaptiveqe](https://bitbucket.org/antonis/adaptiveqe).
Further Work

- Incorporate more features, following recent developments.
- Create and work on different datasets.
- **Personalization**
  - Keep "history" of certain user
  - New features for personalization
Thank you!!