Adaptive Quality Estimation for Machine Translation

Antonis Anastasopoulos

Advisors: Yanis Maistros[1], Marco Turchi[2], Matteo Negri[2]

[1]School of Electrical and Computer Engineering, NTUA, Greece
[2]Fondazione Bruno Kessler, MT Group

April 9, 2014

Outline

Outline

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

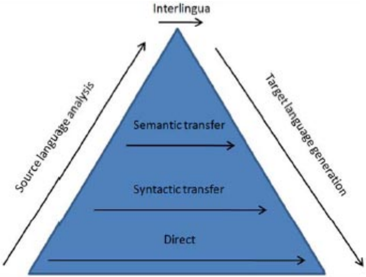
Machine Translation Overview

Various approaches:

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

$$source \xrightarrow{transform} intermediate\ representation \xrightarrow{transform} target$$

- Interlingua



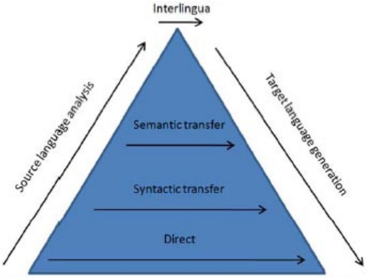
Machine Translation Overview

Various approaches:

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

$$source \xrightarrow{transform} intermediate\ representation \xrightarrow{transform} target$$

- Interlingua



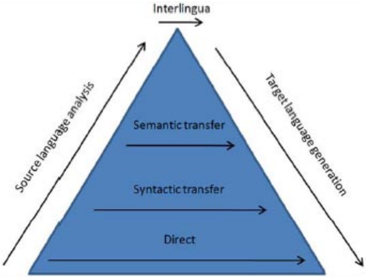
Machine Translation Overview

Various approaches:

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

$$source \xrightarrow{transform} intermediate\ representation \xrightarrow{transform} target$$

- 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 $\mathcal{S}$, out of all possible translations s.

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

From the Bayes rule:

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

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:

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

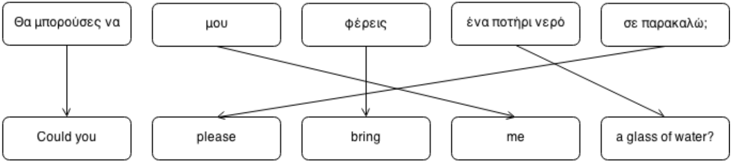
Statistical MT

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
  (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
  (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

HTER Example

source:

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

produced translation:

`Porque tambien tengo una inclinacion por tradicion , modales y costumbres .`

post-edited:

`Porque tambien tengo una inclinacion por la tradicion , los modales y las costumbres .`

$$HTER = \frac{3}{15} = 0.20$$

Table of Contents

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

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

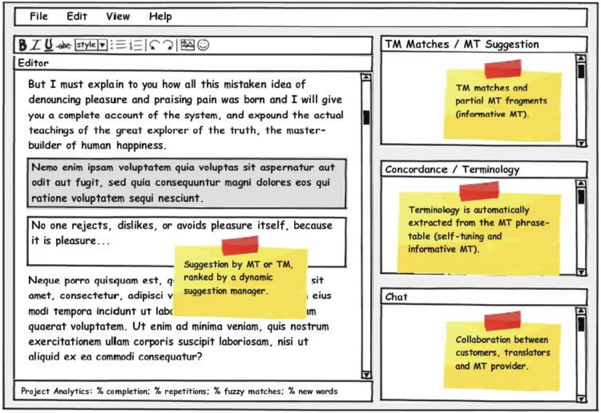
DEFINITION

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

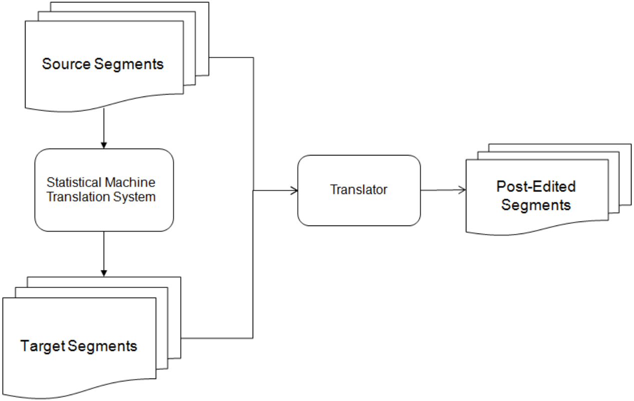
Connection with industry

Vanilla CAT Tool

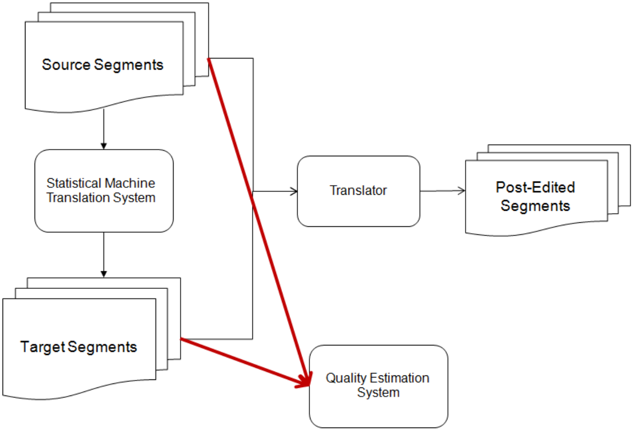


CAT-tool Scenario

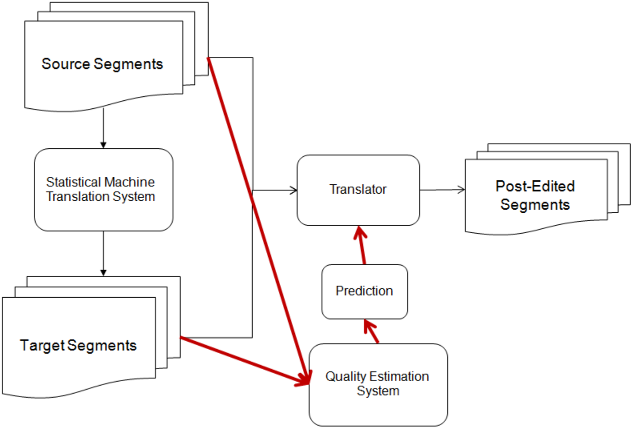
CAT: Computer Assisted Translation



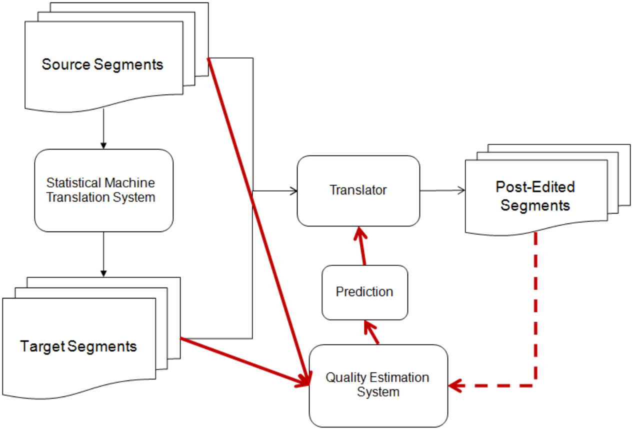
CAT-tool Scenario



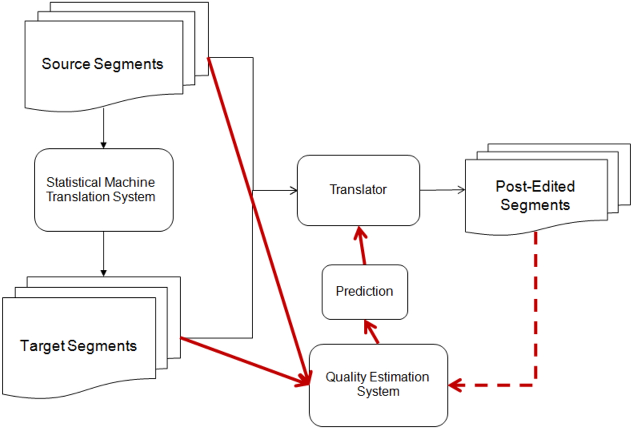
CAT-tool Scenario



CAT-tool Scenario



CAT-tool Scenario



Why Online?

Table of Contents

Motivation and Open Questions

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

System Overview

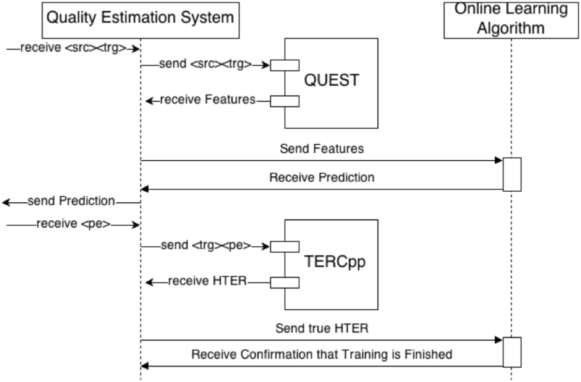


Table of Contents

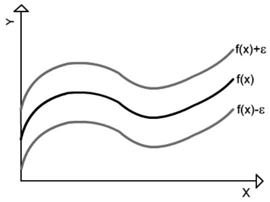
Learning Algorithms

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

Support Vector Regression

Definition

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



Support Vector Regression

Linear regression function:

$$f(\mathbf{x}) = \mathbf{W}^T \Phi(\mathbf{x}) + b$$

Convex optimization problem by requiring:

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

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

Online Support Vector Regression

- Introduced by Ma et al (2003).
- 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: ϵ-insensitive loss function that creates a hyper-slab of width 2ϵ
- Update:

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

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

Passive-Aggressive Algorithms

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

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

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

Passive-Aggressive Algorithms

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

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

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

Passive-Aggressive Algorithms

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

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

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

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 $m(\mathbf{x})$ and the covariance function $k(\mathbf{x}, \mathbf{x}')$:

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

The Gaussian Process assumes that every target y_i is generated from the corresponding data $\mathbf{x}_i$ and an added white noise η as:

$$y_i = f(\mathbf{x}_i) + \eta, \quad where \quad \eta \sim \mathcal{N}(0, \sigma_n^2)$$

This function $f(\mathbf{x})$ is drawn from a GP prior:

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

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

Gaussian Processes

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

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

The Gaussian Process assumes that every target y_i is generated from the corresponding data $\mathbf{x}_i$ and an added white noise η as:

$$y_i = f(\mathbf{x}_i) + \eta, \quad where \quad \eta \sim \mathcal{N}(0, \sigma_n^2)$$

This function $f(\mathbf{x})$ is drawn from a GP prior:

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

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

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* $\mathcal{BV}$ 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* $\mathcal{BV}$ 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 $prob(t|s) > 0.2$)

Table of Contents

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)

$$MAE = \frac{\sum_{i=1}^{n} |\hat{y}_i - y_i|}{n}$$

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)

$$MAE = \frac{\sum_{i=1}^{n} |\hat{y}_i - y_i|}{n}$$

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)

$$MAE = \frac{\sum_{i=1}^{n} |\hat{y}_i - y_i|}{n}$$

Table of Contents

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

Training Labels			Test Labels	
Training	Avg. HTER	St. Dev.	Avg. HTER	St. Dev.
200	32.71	14.99	32.32	17.32
600	33.64	16.72		
1500	33.54	18.56		

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

	Training Labels		Test Labels	
Training	Avg. HTER	St. Dev.	Avg. HTER	St. Dev.
200	32.71	14.99	32.32	17.32
600	33.64	16.72		
1500	33.54	18.56		

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

Training Labels			Test Labels	
Training	Avg. HTER	St. Dev.	Avg. HTER	St. Dev.
200	32.71	14.99	32.32	17.32
600	33.64	16.72		
1500	33.54	18.56		

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

Training Labels			Test Labels	
Training	Avg. HTER	St. Dev.	Avg. HTER	St. Dev.
200	32.71	14.99	32.32	17.32
600	33.64	16.72		
1500	33.54	18.56		

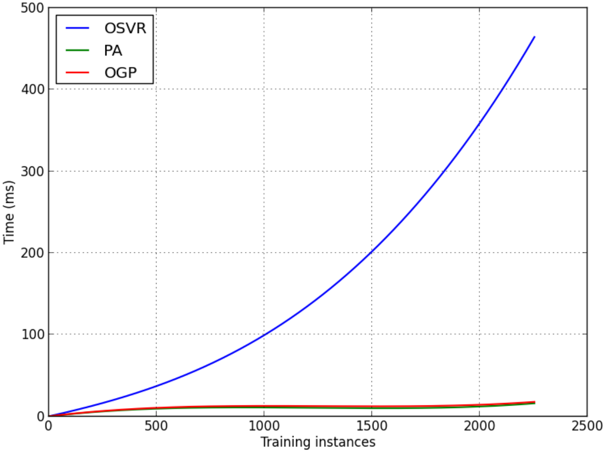
Results for experiment 1

Algorithm	Kernel	MAE ($i = 200$)	MAE ($i = 600$)	MAE ($i = 1500$)
		Batch		
SVR_i	Linear	13.5	13.0	12.8
	RBF	**13.2***	**12.7***	**12.7***
		Adaptive		
$OSVR_i$	Linear	**13.2***	12.9	12.8
	RBF	13.6	13.7	13.5
PA_i	-	14.0	13.4	13.3
OGP_i	RBF	**13.2***	12.9	12.8

Results for experiment 1

Algorithm	Kernel	MAE ($i = 200$)	MAE ($i = 600$)	MAE ($i = 1500$)
Empty				
$OSVR_0$	Linear RBF	13.5 13.7		
PA_0		14.4		
OGP_0	RBF	13.3		

Time performance and complexity



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:

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

Set	Average HTER	HTER St. Deviation
Top	56.27	12.59
Bottom	12.35	6.43

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*.

Set	Average HTER	HTER St. Deviation
Top	56.27	12.59
Bottom	12.35	6.43

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*.

Set	Average HTER	HTER St. Deviation
Top	56.27	12.59
Bottom	12.35	6.43

Results for experiment 2

Test on *Top*		
Algorithm	Kernel	MAE
Batch		
SVR^{Top}_{Bottom}	Linear	43.7
	RBF	43.2
Adaptive		
$OSVR^{Top}_{Bottom}$	Linear	28.7
	RBF	31.1
PA^{Top}_{Bottom}	-	28.2
OGP^{Top}_{Bottom}	RBF	**27.2**

Test on *Bottom*		
Algorithm	Kernel	MAE
Batch		
SVR^{Bottom}_{Top}	Linear	39.3
	RBF	40.7
Adaptive		
$OSVR^{Bottom}_{Top}$	Linear	**27.0**
	RBF	29.5
PA^{Bottom}_{Top}	-	31.0
OGP^{Bottom}_{Top}	RBF	28.3

Results for experiment 2

Algorithm	Kernel	MAE on *Top*	MAE on *Bottom*
Empty			
$OSVR_0$	Linear	8.42	5.67
	RBF	8.55	5.37
PA_0	-	**8.37**	5.30
OGP_0	RBF	8.83	**5.22**

Table of Contents

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 τ metric)
- Instance-wise Difference

HTER correlates better with all the other possible metrics.

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 τ metric)
- Instance-wise Difference

HTER correlates better with all the other possible metrics.

Translator Behaviour

Legal domain:

Post-editor	Avg HTER	HTER St. Deviation
1	29.04	16.84
2	32.33	18.87
3	43.25	14.86
4	23.52	15.80

Translator Behaviour

IT domain:

Post-editor	Avg HTER	HTER St. Deviation
1	39.32	21.03
2	47.77	20.49
3	37.72	20.05
4	36.60	19.71

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:

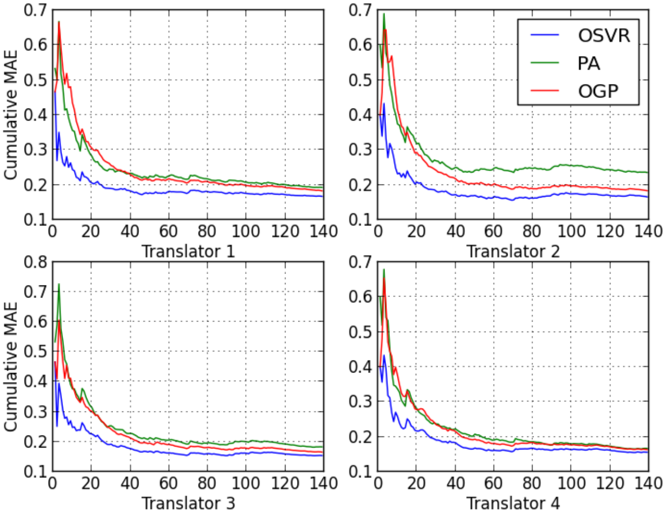
- When post-editors behave similarly, eg. (IT 1,3), *batch* and *adaptive* both work well.
- When post-editors are more different, eg (IT 3,2 or L 3,4), the *adaptive* approach significantly outperforms *batch*.

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
OSVR
PA
OGP

Out-domain Results

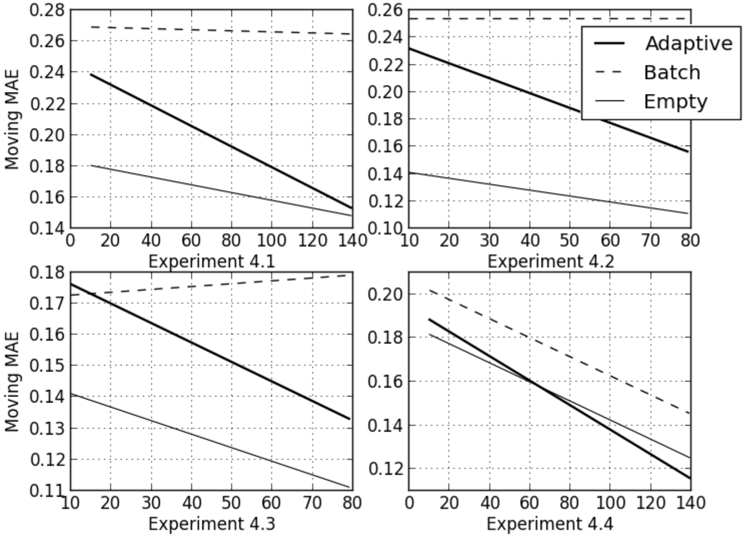
We select the most different translators from each domain (*Low*, *High*).
8 combinations:

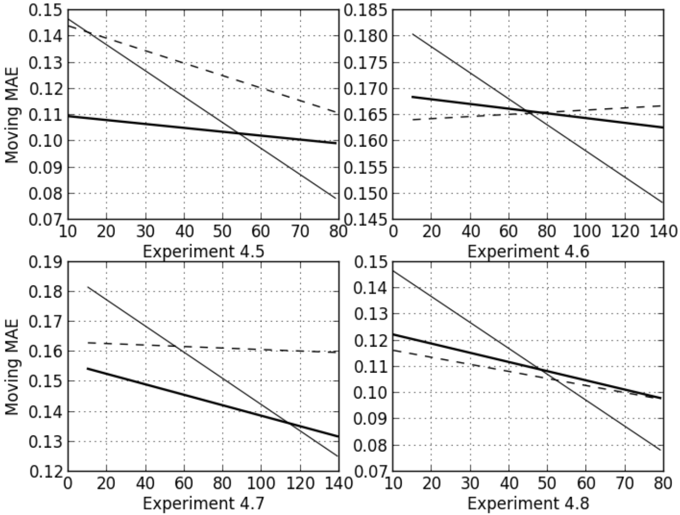
Experiment	*Training* Set	*Test* Set	HTER Diff.
4.1	Low,L	High,IT	24.5
4.2	High,IT	Low,L	24
4.3	Low,IT	Low,L	13.5
4.4	Low,L	Low,IT	12.7
4.5	Low,IT	High,L	8.3
4.6	High,L	High,IT	6.8
4.7	High,L	Low,IT	5
4.8	High,IT	High,L	2.2

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

Correlation of performance and hter difference:

Mode	Correlation
batch	0.945
adaptive	0.812
empty	0.190


Moving MAE
Adaptive
Batch
Empty
Experiment 4.1
Experiment 4.2
Experiment 4.3
Experiment 4.4


Moving MAE
Experiment 4.5
Experiment 4.6
Moving MAE
Experiment 4.7
Experiment 4.8

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

Synopsis

- We introduce the use of *online* learning techniques for the QE task.
- We show that they can deal with data scarsity 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`.

SYNOPSIS

- We introduce the use of *online* learning techniques for the QE task.
- We show that they can deal with data scarsity 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`.

SYNOPSIS

- We introduce the use of *online* learning techniques for the QE task.
- We show that they can deal with data scarsity 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`.

Synopsis

- We introduce the use of *online* learning techniques for the QE task.
- We show that they can deal with data scarsity 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`.

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!!