Quality Estimation of Machine Translation
Recent advances, Challenges and Software

Lucia Specia and Carolina Scarton

{l.specia, c.scarton}@sheffield.ac.uk

Alicante, 21 January 2016
Outline

1 Introduction
   - Structure of the tutorial
   - Quality estimation

2 Variants
   - Sentence-level QE
   - Word-level QE
   - Document-level QE

3 Conclusions

4 Practice
   - QuEst
   - QuEst++
   - Feature Extractor module
   - Machine Learning module
Theory

- Introduction to QE
- Sentence, word, document-level QE
  - Labels
  - Features
  - ML algorithms
  - Results
  - Challenges and future work
- Does QE help improve productivity?
Practice

- Code and installation (dependencies)
- How to run code on existing example
- How to set up a new experiment
- How to implement a new feature
- How to add a new machine learning algorithm
Definition

- Approaches to **predict** the quality of a language output application – no access to “true” output for comparison

- **(Machine) translation**: predict the quality of translation without access to a reference translation
  - **Quality**: fluency, adequacy, post-editing effort, etc.
  - **General method**: supervised machine learning on quality features + labels

- Circa 2001 - **Confidence Estimation**
  - How **confident** MT system is in a translation
  - Mostly word-level prediction from SMT internal features
Definition

- Approaches to **predict** the quality of a language output application – no access to “true” output for comparison

- **(Machine) translation**: predict the quality of translation without access to a reference translation
  - **Quality**: fluency, adequacy, post-editing effort, etc.
  - **General method**: supervised machine learning on quality features + labels

- Circa 2001 - **Confidence Estimation**
  - How **confident** MT system is in a translation
  - Mostly word-level prediction from SMT internal features

- **Now**: popular area, challenging research, commercial interest
Motivations

**MT**: The King closed hearings Monday with Deputy Canary Coalition Ana Maria Oramas González-Moro, who said, in line with the above, that “there is room to have government in the coming months,” although he did not disclose prints Rey about reports Francesco Manetto. Monarch Oramas transmitted to his conviction that ‘soon there will be an election’ because looks unlikely that Rajoy or Sanchez can form a government.
MT: The King closed hearings Monday with Deputy Canary Coalition Ana Maria Oramas González-Moro, who said, in line with the above, that “there is room to have government in the coming months,” although he did not disclose prints Rey about reports Francesco Manetto. Monarch Oramas transmitted to his conviction that ‘soon there will be an election” because looks unlikely that Rajoy or Sanchez can form a government.

SRC: El Rey cerró las audiencias del lunes con la diputada de Coalición Canaria Ana María Oramas González-Moro, quien aseguró, en la línea de los anteriores, que “no hay ambiente de tener Gobierno en los próximos meses”, aunque no desveló las impresiones del Rey al respecto, informa Francesco Manetto. Oramas transmitió al Monarca su convicción de que “pronto habrá un proceso electoral”, porque ve poco probable que Rajoy o Sánchez puedan formar Gobierno.
Motivations

Target:
site security should be included in sex education curriculum for students

Source:
场地安全性教育应纳入学生的课程

Reference:
site security requirements should be included in the education curriculum for students

By Google Translate
Motivations

Target:
the road boycotted a friend ... indian robin hood killed the poor after 32 years of prosecution.

Source:
مقتل روبن هود الهندي.. قاطع الطريق صديق الفقراء بعد 32 عاما من الملاحقة

Reference:
death of the indian robin hood, highway robber and friend of the poor, after 32 years on the run.

By Google Translate
Uses

Quality = **Can we publish it as is?**

Quality = **Can a reader get the gist?**

Quality = **Is it worth post-editing it?**

Quality = **How much effort to fix it?**

Quality = **Which words need fixing?**

Quality = **Which version of the document is more reliable?**
General method

Source documents

Target documents
General method

Source documents -> Feature extractor -> Target documents
General method

Source documents → Feature extractor → Features for QE → Target documents
General method

Source documents → Feature extractor → Features for QE → QE model training → Target documents
General method

- **Source documents**
  - Feature extractor
  - Features for QE
- **Target documents**
- **Quality labels**
  - Likert
  - HTER
  - BLEU
  - ...

- **QE model training**
General method

Source documents → Feature extractor → Features for QE → QE model training → QE model → Quality labels

- Likert
- HTER
- BLEU
- ...

Target documents
General method

1. Source documents
2. Target documents
3. Feature extractor
4. Features for QE
5. QE model training
6. QE model
7. Predictions
8. Quality labels:
   - Likert
   - HTER
   - BLEU
   - ...

Structure of the tutorial
- Introduction
- Variants
- Conclusions
- Practice

Quality estimation

Lucia Specia and Carolina Scarton
Main components to build a QE system:

1. Definition of quality: **what to predict**
2. (Human) labelled **data** (for quality)
3. **Features**
4. Machine learning **algorithm**
Different **users**, different **needs**

Ref: Do **not** buy this product, it's their craziest invention!

MT: Do buy this product, it’s their craziest invention!
Quality

Different **users**, different **needs**

*Ref:* Do **not** buy this product, it's their craziest invention!
*MT:* Do buy this product, it’s their craziest invention!

- **Severe** if end-user does not speak source language
- **Trivial** to post-edit by translators
Quality

Different **users**, different **needs**

**Ref:** Do **not** buy this product, it's their craziest invention!

**MT:** Do buy this product, it’s their craziest invention!

- **Severe** if end-user does not speak source language
- **Trivial** to post-edit by translators

**Ref:** The **battery lasts 6 hours** and it can be **fully recharged** in **30 minutes**.

**MT:** **Six-hour battery**, **30 minutes** to **full charge last**.
Different users, different needs

Ref: Do not buy this product, it's their craziest invention!
MT: Do buy this product, it’s their craziest invention!

- **Severe** if end-user does not speak source language
- **Trivial** to post-edit by translators

Ref: The battery lasts 6 hours and it can be fully recharged in 30 minutes.
MT: Six-hour battery, 30 minutes to full charge last.

- **Ok** for gisting - meaning preserved
- **Very costly** for post-editing if style is to be preserved
Features

Adequacy indicators

Source text → MT system → Translation

Complexity indicators

Confidence indicators

Fluency indicators
Sentence-level QE

- Most popular level
  - MT systems work at sentence-level
  - PE is done at sentence-level

- Easier to get labelled data

- Practical mainly for **post-editing purposes**

- Wide range of features possible: sentence-wide, sentence within document, words in sentence
Old work: predict NIST, TER, BLEU against independently created reference

Explicit:
- Predict 1-N absolute scores for adequacy/fluency
- Predict 1-N absolute scores for post-editing effort
- Predict good/bad score for whatever purpose
- Predict relative rankings for same/different sources, by one or more MT systems

Implicit:
- Predict average post-editing time per word
- Predict percentage of edits needed for sentence
Features

**MT system-independent** features:

- **SF - Source complexity features:**
  - source sentence length
  - source sentence type/token ratio
  - average source word length
  - source sentence 3-gram LM score
  - percentage of source 1 to 3-grams seen in the MT training corpus
Features

**MT system-independent** features:

- **SF - Source complexity features:**
  - source sentence length
  - source sentence type/token ratio
  - average source word length
  - source sentence 3-gram LM score
  - percentage of source 1 to 3-grams seen in the MT training corpus

- **TF - Target fluency features:**
  - target sentence 3-gram LM score
  - translation sentence length
  - proportion of mismatching opening/closing brackets and quotation marks in translation
  - coherence of the target sentence
Features

- **AF - Adequacy features:**
  - ratio of number of tokens btw source & target and v.v.
  - absolute difference btw no tokens in source & target
  - absolute difference btw no brackets, numbers, punctuation symbols in source & target
  - ratio of no, content-/non-content words btw source & target
  - ratio of nouns/verbs/pronouns/etc btw source & target
  - proportion of dependency relations with constituents aligned btw source & target
  - difference btw depth of the syntactic trees of source & target
  - difference btw no pp(np)/vp/adjp/advp/conjp phrase labels in source & target
  - difference btw no 'person'/ 'location'/ 'organization' (aligned) entities in source & target
  - proportion of matching base-phrase types at different levels of source & target parse trees
Features

- **Confidence** features:
  - score of the hypothesis (MT global score)
  - size of nbest list
  - using n-best to build LM: sentence n-gram log-probability
  - individual model features (phrase probabilities, etc.)
  - maximum/minimum/average size of the phrases in translation
  - proportion of unknown/untranslated words
  - n-best list density (vocabulary size / average sentence length)
  - edit distance of the current hypothesis to the center hypothesis
  - Search graph info: total hypotheses, % discarded / pruned / recombined search graph nodes

- Other: **average quality prediction for words in sentence**

http://www.quest.dcs.shef.ac.uk/quest_files/features_blackbox
http://www.quest.dcs.shef.ac.uk/quest_files/features_glassbox
Algorithms

- Mostly **regression** algorithms (SVM, GP)
- Binary **classification**
- **Kernel** methods perform better
- **Tree kernel** methods for syntactic trees
- Advanced approaches: online learning, multi-task learning
WMT15:

- English → Spanish
- Predicting HTER [0-100]
- One MT system
- News
- Crowdsourced annotations
- Training/dev: 11,271/1,000 <source, MT, PE, HTER>
- Test: 1,817 <source, MT>
## Results

**WMT15:**

<table>
<thead>
<tr>
<th>System ID</th>
<th>MAE ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>English-Spanish</td>
<td></td>
</tr>
<tr>
<td>RTM-DCU/RTM-FS+PLS-SVR</td>
<td>13.25</td>
</tr>
<tr>
<td>LORIA/17+LSI+MT+FILTRE</td>
<td>13.34</td>
</tr>
<tr>
<td>RTM-DCU/RTM-FS-SVR</td>
<td>13.35</td>
</tr>
<tr>
<td>LORIA/17+LSI+MT</td>
<td>13.42</td>
</tr>
<tr>
<td>UGENT-LT3/SCATE-SVM</td>
<td>13.71</td>
</tr>
<tr>
<td>UGENT-LT3/SCATE-SVM-single</td>
<td>13.76</td>
</tr>
<tr>
<td>SHEF/SVM</td>
<td>13.83</td>
</tr>
<tr>
<td><strong>Baseline SVM</strong></td>
<td><strong>14.82</strong></td>
</tr>
<tr>
<td>SHEF/GP</td>
<td>15.16</td>
</tr>
</tbody>
</table>

- = winning submissions - top-scoring and those which are not significantly worse.
Gray area = systems that are not significantly different from the baseline.
## Results

Did we do better than **WMT14**?

<table>
<thead>
<tr>
<th>System ID</th>
<th>MAE  ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>English-Spanish</strong></td>
<td></td>
</tr>
<tr>
<td>• FBK-UPV-UEDIN/WP</td>
<td>12.89</td>
</tr>
<tr>
<td>• RTM-DCU/RTM-SVR</td>
<td>13.40</td>
</tr>
<tr>
<td>• USHEFF</td>
<td>13.61</td>
</tr>
<tr>
<td>RTM-DCU/RTM-TREE</td>
<td>14.03</td>
</tr>
<tr>
<td>DFKI/SVR</td>
<td>14.32</td>
</tr>
<tr>
<td>FBK-UPV-UEDIN/NOWP</td>
<td>14.38</td>
</tr>
<tr>
<td>SHEFF-lite/sparse</td>
<td>15.04</td>
</tr>
<tr>
<td>MULTILIZER</td>
<td>15.04</td>
</tr>
<tr>
<td><strong>Baseline SVM</strong></td>
<td>15.23</td>
</tr>
<tr>
<td>DFKI/SVRxdata</td>
<td>16.01</td>
</tr>
<tr>
<td>SHEFF-lite</td>
<td>18.15</td>
</tr>
</tbody>
</table>
WMT15:
Pearson correlation (Graham, 2015) = DeltaAvg’s ranking

<table>
<thead>
<tr>
<th>System ID</th>
<th>Pearson’s $r$ ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>• LORIA/17+LSI+MT+FILTRE</td>
<td>0.39</td>
</tr>
<tr>
<td>• LORIA/17+LSI+MT</td>
<td>0.39</td>
</tr>
<tr>
<td>• RTM-DCU/RTM-FS+PLS-SVR</td>
<td>0.38</td>
</tr>
<tr>
<td>RTM-DCU/RTM-FS-SVR</td>
<td>0.38</td>
</tr>
<tr>
<td>UGENT-LT3/SCATE-SVM</td>
<td>0.37</td>
</tr>
<tr>
<td>UGENT-LT3/SCATE-SVM-single</td>
<td>0.32</td>
</tr>
<tr>
<td>SHEF/SVM</td>
<td>0.29</td>
</tr>
<tr>
<td>SHEF/GP</td>
<td>0.19</td>
</tr>
<tr>
<td><strong>Baseline SVM</strong></td>
<td><strong>0.14</strong></td>
</tr>
</tbody>
</table>
Challenges and future work

- **Data**: how to obtain objective labels, for different languages and domains, which are comparable across translators? → **By product of PE**

- How to deal with **biases** from annotators (or domains)? → Multi-task learning

- How to **adapt** models over time? → Online (multitask) learning
Some **applications** require fine-grained information on quality:
- Highlight words that need fixing
- Inform readers of portions of sentence that are not reliable

Seemingly a **more challenging task**
- A quality label is to be predicted for each target word
- Sparsity is a serious issue
- Skewed distribution towards GOOD
- Errors are interdependent
Predict binary **GOOD/BAD** labels

Predict general **types of edits**: Shift, Replacement, Insertion, Deletion is an issue

Predict specific errors. E.g. **MQM** in WMT14
Features

- target token, its left & right token
- source token aligned to target token, its left & right tokens
- boolean dictionary flag: whether target token is a stopword, a punctuation mark, a proper noun, a number
- dangling token flag (null link)
- LM of n-grams with target token $t_i$: $(t_{i-2}, t_{i-1}, t_i)$, $(t_{i-1}, t_i, t_{i+1})$, $(t_i, t_{i+1}, t_{i+2})$
- order of the highest order n-gram which starts/ends with the source/target token
- POS tag of target/source token
- number of senses of target/source token in WordNet
- pseudo-reference flag: 1 if token belongs to pseudo-reference, 0 otherwise
Algorithms

- **Sequence labelling** algorithms, like CRF

  ![Sequence labelling diagram](attachment:sequence_labeling.png)

- **Classification** algorithms: each word tagged independently

  ![Classification diagram](attachment:classification.png)

- **DNN**: multilayer perceptron with bilingual word embeddings
  - Linear combination with classifier on standard features ([WMT15](#))
WMT15:

- English → Spanish, one MT system, News
- Labelling done with TERCOM:
  - OK = unchanged
  - BAD = insertion, substitution
- Data: <source word, MT word, OK/BAD label>

<table>
<thead>
<tr>
<th></th>
<th>Sentences</th>
<th>Words</th>
<th>% of BAD words</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>11,271</td>
<td>257,548</td>
<td>19.14</td>
</tr>
<tr>
<td>Dev</td>
<td>1,000</td>
<td>23,207</td>
<td>19.18</td>
</tr>
<tr>
<td>Test</td>
<td>1,817</td>
<td>40,899</td>
<td>18.87</td>
</tr>
</tbody>
</table>

Challenge: skewed class distribution
Results

**WMT15:**
- Mostly interested in finding errors
- Precision/recall preferences depend on application
- Rare classes should not dominate

**Evaluation metric:** weighted average $F_1$ of “BAD” class

**Baseline** introduced:
- CRF classifier with 25 features
## Results

<table>
<thead>
<tr>
<th>System ID</th>
<th>weighted $F_1$ All ↑</th>
<th>$F_1$ BAD ↑</th>
<th>$F_1$ OK ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>UAlacant/OnLine-SBI-Baseline</td>
<td>71.47</td>
<td>43.12</td>
<td>78.07</td>
</tr>
<tr>
<td>HDCL/QUETCHPLUS</td>
<td>72.56</td>
<td>43.05</td>
<td>79.42</td>
</tr>
<tr>
<td>UAlacant/OnLine-SBI</td>
<td>69.54</td>
<td>41.51</td>
<td>76.06</td>
</tr>
<tr>
<td>SAU/KERC-CRF</td>
<td>77.44</td>
<td>39.11</td>
<td>86.36</td>
</tr>
<tr>
<td>SAU/KERC-SLG-CRF</td>
<td>77.4</td>
<td>38.91</td>
<td>86.35</td>
</tr>
<tr>
<td>SHEF2/W2V-BI-2000</td>
<td>65.37</td>
<td>38.43</td>
<td>71.63</td>
</tr>
<tr>
<td>SHEF2/W2V-BI-2000-SIM</td>
<td>65.27</td>
<td>38.40</td>
<td>71.52</td>
</tr>
<tr>
<td>SHEF1/QuEst++-AROW</td>
<td>62.07</td>
<td>38.36</td>
<td>67.58</td>
</tr>
<tr>
<td>UGEN/SCATE-HYBRID</td>
<td>74.28</td>
<td>36.72</td>
<td>83.02</td>
</tr>
<tr>
<td>DCU-SHEFF/BASE-NGRAM-2000</td>
<td>67.33</td>
<td>36.60</td>
<td>74.49</td>
</tr>
<tr>
<td>HDCL/QUETCH</td>
<td>75.26</td>
<td>35.27</td>
<td>84.56</td>
</tr>
<tr>
<td>DCU-SHEFF/BASE-NGRAM-5000</td>
<td>75.09</td>
<td>34.53</td>
<td>84.53</td>
</tr>
<tr>
<td>SHEF1/QuEst++-PA</td>
<td>26.25</td>
<td>34.30</td>
<td>24.38</td>
</tr>
<tr>
<td>Baseline (always BAD)</td>
<td>0.599</td>
<td>31.76</td>
<td>0.00</td>
</tr>
<tr>
<td>UGEN/SCATE-MBL</td>
<td>74.17</td>
<td>30.56</td>
<td>84.32</td>
</tr>
<tr>
<td>RTM-DCU/s5-RTM-GLMd</td>
<td>76.00</td>
<td>23.91</td>
<td>88.12</td>
</tr>
<tr>
<td>RTM-DCU/s4-RTM-GLMd</td>
<td>75.88</td>
<td>22.69</td>
<td>88.26</td>
</tr>
<tr>
<td>Baseline CRF</td>
<td>75.31</td>
<td>16.78</td>
<td>88.93</td>
</tr>
<tr>
<td>Baseline (always OK)</td>
<td>72.67</td>
<td>0.00</td>
<td>89.58</td>
</tr>
</tbody>
</table>
Did we do better than **WMT14**?

<table>
<thead>
<tr>
<th>System ID</th>
<th>weighted $F_1$</th>
<th>$F_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>All ↑</td>
<td>BAD ↑</td>
</tr>
<tr>
<td>Baseline (always OK)</td>
<td>50.43</td>
<td>0.00</td>
</tr>
<tr>
<td>Baseline (always BAD)</td>
<td>18.71</td>
<td>52.53</td>
</tr>
<tr>
<td>FBK-UPV-UEDIN/RNN</td>
<td>62.00</td>
<td>48.73</td>
</tr>
<tr>
<td>LIMSI/RF</td>
<td>60.55</td>
<td>47.32</td>
</tr>
<tr>
<td>LIG/FS</td>
<td>63.55</td>
<td>44.47</td>
</tr>
<tr>
<td>LIG/BL ALL</td>
<td>63.77</td>
<td>44.11</td>
</tr>
<tr>
<td>FBK-UPV-UEDIN/CRF</td>
<td>62.17</td>
<td>42.63</td>
</tr>
<tr>
<td>RTM-DCU/RTM-GLM</td>
<td>60.68</td>
<td>35.08</td>
</tr>
<tr>
<td>RTM-DCU/RTM-GLMd</td>
<td>60.24</td>
<td>32.89</td>
</tr>
</tbody>
</table>
Challenges and future work

- **Data:**
  - Labelling is very expensive → by product of post-editing
  - Labelling from post-editing not reliable → need better alignment methods
  - Data sparsity and skewness are hard to overcome →
    - Injecting errors or filtering positive cases
    - DNNs for generalisation - but require large datasets

- Errors are rarely isolated – how to model **interdependencies**?
  → **Phrase-level QE - WMT16**
Document-level QE

- Prediction of a single label for **entire documents**

- **Assumption**: quality of a document is more than the simple aggregation of its sentence-level quality scores
  - While certain sentences are perfect in isolation, their combination in context may lead to an incoherent document
  - A sentence can be poor in isolation, but good in context as it may benefit from information in surrounding sentences

- **Feature engineering** is challenging: few processing tools for discourse-wide information
  - Topic and structure of document
  - Relationship between its sentences/paragraphs

- Parallel data with **doc-level markup** not commonly found
Notion of **quality** is very subjective
- Human labels are difficult and too expensive to get
- No datasets with human labels are available.

Predict **METEOR, BLEU** against independently created reference

Either as:
- Absolute score, or
- Relative ranking of translations by one or more MT systems
Features

- aggregation or doc-level counts of sentence-level features
- word/lemma/noun repetition in source/target doc
- ratio of word/lemma/noun repetition btw source & target docs
- number of pronouns in source/target doc
- number of discourse connectives of type **Expansion**, **Temporal**, **Contingency**, **Comparison** and **Non-discourse**
- number of EDU (elementary discourse units) breaks in source/target doc
- number of RST (Rhetorical Structure Theory) **Nucleus** relations in source/target doc
- number of RST **Satellite** relations in source/target doc
- average quality prediction for sentences in docs
Algorithms

- Same as for sentence-level
- Tree kernels for discourse parser trees
Results

WMT15:
- English → German, German → English
- **Paragraphs** from all WMT13 translation task MT systems
- 800 for training; 415 for test
- Average METEOR scores in data [0,1]:

<table>
<thead>
<tr>
<th></th>
<th>EN-DE</th>
<th></th>
<th>DE-EN</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>METEOR (↑)</td>
<td>AVG</td>
<td>STDEV</td>
<td>AVG</td>
<td>STDEV</td>
</tr>
<tr>
<td></td>
<td>0.35</td>
<td>0.14</td>
<td>0.26</td>
<td>0.09</td>
</tr>
</tbody>
</table>

ps.: it is possible that variation (STDEV) is more affected by the difference in MT systems than the difference in quality or the fact that it’s paragraphs. E.g. EN-DE doc-level 1 MT system, AVG = 0.39 and STDEV 0.058
## Results

<table>
<thead>
<tr>
<th>System ID</th>
<th>MAE ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>English-German</strong></td>
<td></td>
</tr>
<tr>
<td>RTM-DCU/RTM-FS-SVR</td>
<td>7.28</td>
</tr>
<tr>
<td>RTM-DCU/RTM-SVR</td>
<td>7.5</td>
</tr>
<tr>
<td>USAAR-USHEF/BFF</td>
<td>9.37</td>
</tr>
<tr>
<td>USHEF/QUEST-DISC-REP</td>
<td>9.55</td>
</tr>
<tr>
<td><strong>Baseline SVM</strong></td>
<td>10.05</td>
</tr>
<tr>
<td><strong>German-English</strong></td>
<td></td>
</tr>
<tr>
<td>RTM-DCU/RTM-FS-SVR</td>
<td>4.94</td>
</tr>
<tr>
<td>RTM-DCU/RTM-FS+PLS-SVR</td>
<td>5.78</td>
</tr>
<tr>
<td>USHEF/QUEST-DISC-BO</td>
<td>6.54</td>
</tr>
<tr>
<td>USAAR-USHEF/BFF</td>
<td>6.56</td>
</tr>
<tr>
<td><strong>Baseline SVM</strong></td>
<td>7.35</td>
</tr>
</tbody>
</table>
## Results

<table>
<thead>
<tr>
<th>System ID</th>
<th>Pearson’s $r$ ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>English-German</strong></td>
<td></td>
</tr>
<tr>
<td>RTM-DCU/RTM-SVR</td>
<td>0.59</td>
</tr>
<tr>
<td>RTM-DCU/RTM-FS-SVR</td>
<td>0.53</td>
</tr>
<tr>
<td>USHEF/QUEST-DISC-REP</td>
<td>0.30</td>
</tr>
<tr>
<td>USAAR-USHEF/BFF</td>
<td>0.29</td>
</tr>
<tr>
<td><strong>Baseline SVM</strong></td>
<td>0.12</td>
</tr>
<tr>
<td><strong>German-English</strong></td>
<td></td>
</tr>
<tr>
<td>RTM-DCU/RTM-FS-SVR</td>
<td>0.52</td>
</tr>
<tr>
<td>RTM-DCU/RTM-FS+PLS-SVR</td>
<td>0.39</td>
</tr>
<tr>
<td>USHEF/QUEST-DISC-BO</td>
<td>0.10</td>
</tr>
<tr>
<td>USAAR-USHEF/BFF</td>
<td>0.08</td>
</tr>
<tr>
<td><strong>Baseline SVM</strong></td>
<td>0.06</td>
</tr>
</tbody>
</table>
Challenges and future work

- BLEU-style metrics are not ideal to be used as labels
  - They have well known limitations as evaluation metrics
  - They were designed to evaluate different MT systems and not different documents produced by the same MT system

- Ideal quality label should take into account purpose of translation →
  - Two-stage post-editing: isolate document-level problems
  - Reading comprehension assessment: how much of the MT text is understandable by humans

- Better features are needed for discourse information at different levels
  - micro units (lexical, EDUs)
  - macro units (sentences, paragraphs)
Does QE help?

- (Specia, 2011) **Time to post-edit** subset of sentences predicted as “low PE effort” **vs** time to post-edit random subset of sentences

<table>
<thead>
<tr>
<th>Language</th>
<th>no QE</th>
<th>QE</th>
</tr>
</thead>
<tbody>
<tr>
<td>fr-en</td>
<td>0.75 words/sec</td>
<td>1.09 words/sec</td>
</tr>
<tr>
<td>en-es</td>
<td>0.32 words/sec</td>
<td>0.57 words/sec</td>
</tr>
</tbody>
</table>
QE in practice

Does QE help?

- (Huang et al., 2014) **Productivity increase** of 10% at IBM when translation is supported by the estimates of a QE system
- Comparison between using MT suggestions with predicted QE labels against not using MT at all
QE in practice

Does QE help?

- (Turchi et al., 2015) **Small productivity increase**
- Comparison btw post-editing with and without QE
- Predictions shown with binary colour codes (**green** vs **red**)

Lucia Specia and Carolina Scarton

Quality Estimation of Machine Translation
QE in practice

Does QE help?

- (Turchi et al., 2015) **Small productivity increase**
- Comparison btw post-editing with and without QE
- Predictions shown with binary colour codes (green vs red)

| Average PET (sec/word) | colored grey | 8.086 | 9.592 | $p = 0.33$
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>% Wins of colored</td>
<td>51.7</td>
<td></td>
<td></td>
<td>$p = 0.039$</td>
</tr>
</tbody>
</table>

![Graph showing % Wins of Colored vs HTER](chart.png)
Next round - WMT16

- Large datasets collected as part of QT21: 15K segments
- EN-DE as starting point
- Post-editing by professionals
- Document-level: new annotation schemes by humans

http://www.statmt.org/wmt16/quality-estimation-task.html
Conclusions

- Different metrics for: different purposes/users, different needs, different notions of **quality**
- **Quality estimation**: learning of these different notions, based on labelled data
- Estimates can be used in **real applications**
- Estimates have been shown to help
- **Commercial** interest: IBM, Multilizer, SLD, KantanMT, Xerox, ...
- **Advanced topics**: multi-task learning, phrase-level QE, pipelined prediction, DNNs, etc.
- **Utility of QE** in practice: needs to be further validated
**Goal**: framework to explore features for QE
**Goal**: framework to explore features for QE

- **Feature extractors**: for 150+ features of all types: Java
**Goal**: framework to explore features for QE

- **Feature extractors**: for $150+$ features of all types: Java
- **Machine learning**: wrappers for a number of algorithms in the scikit-learn toolkit, grid search, feature selection: Python
Definition

**Goal**: framework to explore features for QE

- **Feature extractors**: for 150+ features of all types: Java
- **Machine learning**: wrappers for a number of algorithms in the scikit-learn toolkit, grid search, feature selection: Python

Open source: [http://www.quest.dcs.shef.ac.uk/](http://www.quest.dcs.shef.ac.uk/)
**New release**: word and document-level feature extraction and machine learning added
**New release**: word and document-level feature extraction and machine learning added

- **Feature extractors**: 40 features for word-level and 79 features for document-level
**New release:** word and document-level feature extraction and machine learning added

- **Feature extractors:** 40 features for word-level and 79 features for document-level

- **Machine learning:** support to Conditional Random Fields (CRF) added for word-level models
**New release**: word and document-level feature extraction and machine learning added

- **Feature extractors**: 40 features for word-level and 79 features for document-level
- **Machine learning**: support to Conditional Random Fields (CRF) added for word-level models
- **Another important improvement**: changes on the core functionalities
System and baseline features required

- **Java 8** (OpenJDK or Oracle versions)
System and baseline features required

- **Java 8** (OpenJDK or Oracle versions)
- **Sentence and word-level baseline features**
  - Perl 5 (or above)
  - SRILM
  - Tokenizer and Truecaser (from Moses toolkit)
System and baseline features required

- **Java 8** (OpenJDK or Oracle versions)
- **Sentence and word-level baseline features**
  - Perl 5 (or above)
  - SRILM
  - Tokenizer and Truecaser (from Moses toolkit)
- **Word-level features**
  - Stanford Core NLP 3.5.1 models
  - Stanford Core NLP Spanish model
  - Universal WordNet plugin
Basic Usage - Word-level

java -cp QuEst++.jar:lib/* shef.mt.WordLevelFeatureExtractor
-lang <<lang_source>> <<lang_target>>
-input <<input_source>> <<input_target>>
-alignments <<alignment_file>>
-config <<config_file>>
Basic Usage - Sentence-level

java -cp QuEst++.jar shef.mt.SentenceLevelFeatureExtractor -tok -case true -lang <<lang_source>> <<lang_target>> -input <<input_source>> <<input_target>> -config <<config_file>>
Basic Usage - Document-level

```java
java -cp QuEst++.jar shef.mt.DocLevelFeatureExtractor
-tok -case true
-lang <<-lang_source>> <<-lang_target>>
-input <<-input_source>> <<-input_target>>
-config <<-config_file>>
```
Input files

- **Word and sentence levels**: file with one sentence per line.
Input files

- **Word and sentence levels**: file with one sentence per line
- **Document level**: file with paths for documents
Input files

- **Word and sentence levels**: file with one sentence per line
- **Document level**: file with paths for documents

**Files from source and target should have the same number of lines**
**Folders**

- **src**: source code
- **lang_resources**: folder containing all language resources required for the features
- **lib**: external libraries needed for feature extraction
- **config**: configuration files for running QuEst++
- **input**: auxiliary input folder
- **output**: output folder
Adding a new feature

- Example with **sentence-level** feature extractor
Adding a new feature

- Example with `sentence-level` feature extractor
- New feature: `complex words per sentence` (averaged by the length of sentence)
Adding a new feature

- Example with *sentence-level* feature extractor
- New feature: *complex words per sentence* (averaged by the length of sentence)
- Language Resource: *list of simple words* (LSW)
Adding a new feature

- Example with **sentence-level** feature extractor
- New feature: **complex words per sentence** (averaged by the length of sentence)
- Language Resource: **list of simple words** (LSW)
- **Idea**: count words not in the LSW and normalise by number of words in the sentence
Adding a new feature

- **Creating a processor for the new feature**
  - Package: `shef.mt.tools`
Adding a new feature

- Creating a processor for the new feature
  - Package: `shef.mt.tools`
  - Function: prepare resources to be used by features
Adding a new feature

Creating a processor for the new feature

- Package: `shef.mt.tools`
- Function: prepare resources to be used by features
- Extends `ResourceProcessor` class: add the resources to the sentence (`processNextSentence` method)
Adding a new feature

- **Creating a processor for the new feature**
  - Package: `shef.mt.tools`
  - Function: prepare resources to be used by features
  - Extends `ResourceProcessor` class: add the resources to the sentence (`processNextSentence` method)
  - It is useful because a **unique processor** can be used by several features
Adding a new feature

- Create a new Java class called `ComplexWordsProcessor.java`
  - Package: `shef.mt.tools`
Adding a new feature

- Create a new Java class called **ComplexWordsProcessor.java**
  - Package: *shef.mt.tools*
  - Extends: **ResourceProcessor** class
Adding a new feature

- **Create a new Java class called ComplexWordsProcessor.java**
  - Package: `shef.mt.tools`
  - Extends: `ResourceProcessor` class
  - Read the LSW and store it on a ArrayList
Adding a new feature

- Creating a class for the new feature
  - Package: `shef.mt.features.impl.bb`
Creating a class for the new feature
- Package: `shef.mt.features.impl.bb`
- Extends `Feature` class: `run` method - feature extraction itself
Adding a new feature

- Creating a class for the new feature
  - Package: `shef.mt.features.impl.bb`
  - Extends `Feature` class: `run` method - feature extraction itself
  - Feature classes are usually named following a number order (e.g. Feature1001, Feature1002)
Adding a new feature

- **Create a new Java class called Feature7001.java**
  - Package: `shef.mt.features.impl.bb`
Adding a new feature

- **Create a new Java class called Feature7001.java**
  - Package: `shef.mt.features.impl.bb`
  - Extends: `Feature` class
Adding a new feature

- **Create a new Java class called Feature7001.java**
  - Package: `shef.mt.features.impl.bb`
  - Extends: `Feature` class
  - Get the ArrayList from the `ComplexWordsProcessor` class
    and calculate the feature
Adding a new feature

- **Create a new Java class called Feature7001.java**
  - Package: `shef.mt.features.impl.bb`
  - Extends: `Feature` class
  - Get the ArrayList from the `ComplexWordsProcessor` class and calculate the feature
  - Also define the resource that will be required for this feature
Adding a new feature

- **Feature configuration file**
  - Folder: `config/features`
Adding a new feature

- **Feature configuration file**
  - Folder: `config/features`
  - XML file with the featureset that will be executed
Adding a new feature

- **Feature configuration file**
  - Create a file named `features_complex_words.xml` inside the folder `config/features`
Adding a new feature

- **Feature configuration file**
  - Create a file named `features_complex_words.xml` inside the folder `config/features`
  - Add the new feature to this file
Adding a new feature

- **Configuration file**
  - Folder: `config`
Adding a new feature

- **Configuration file**
  - Folder: `config`
  - For sentence-level: `config.sentence-level.properties`
Adding a new feature

**Configuration file**
- Folder: `config`
- For sentence-level: `config.sentence-level.properties`
- Contains basic configuration for the system and paths to resources and tools
Adding a new feature

- **Configuration file**
  - Add the resource `source.simplewords` to the configuration file
Adding a new feature

**Configuration file**

- Add the resource `source.simplewords` to the configuration file
- Change the option `featureConfig` to the path to `features_complex_words.xml`
Adding a new feature

- **SentenceLevelProcessorFactory.java**
  - Package: `shef.mt.tools`
Adding a new feature

- **SentenceLevelProcessorFactory.java**
  - Package: *shef.mt.tools*
  - Function: create all processors required by the features
Adding a new feature

- **SentenceLevelProcessorFactory.java**
  - Package: `shef.mt.tools`
  - Function: create all processors required by the features
  - Only generate processors that will be used (improvement of QuEst++)
Adding a new feature

- **SentenceLevelProcessorFactory.java**
- Package: `shef.mt.tools`
- Function: create all processors required by the features
- Only generate processors that will be used (improvement of QuEst++)
- It is the connection between features and configuration file
Adding a new feature

- **SentenceLevelProcessorFactory.java**
- Package: shef.mt.tools
Adding a new feature

- **SentenceLevelProcessorFactory.java**
  - Package: `shef.mt.tools`
  - Add an `if` block containing the calling to a method called `getComplexWordsProcessor`
Adding a new feature

- **SentenceLevelProcessorFactory.java**
  - Package: `shef.mt.tools`
  - Add an `if` block containing the calling to a method called `getComplexWordsProcessor`
  - Implement `getComplexWordsProcessor` method
Build

- NetBeans 8.1
- ant "-Dplatforms.JDK_1.8.home=/usr/lib/jvm/java-8-<<version>>>"
Run

java -cp QuEst++.jar shef.mt.SentenceLevelFeatureExtractor -tok -case true -lang <<lang_source>> <<lang_target>> -input <<input_source>> <<input_target>> -config <<config_file>>

Check the file output.txt inside output/test
System requirements

- **Python 2.7.6** (or above - only 2.7 stable distributions)
System requirements

- **Python 2.7.6** (or above - only 2.7 stable distributions)
- **SciPy** and **NumPy** (SciPy $\geq$0.9 and NumPy $\geq$1.6.1)
- **scikit-learn** (version 0.15.2)
- **PyYAML**
System requirements

- **Python 2.7.6** (or above - only 2.7 stable distributions)
- **SciPy** and **NumPy** (SciPy >=0.9 and NumPy >=1.6.1)
- **scikit-learn** (version 0.15.2)
- **PyYAML**
- **GPy**
- **CRFsuite**
Folders

- **learning**: main folder
- **config**: configuration files
- **src**: source code files
- **data**: example data (same format as output of feature extractor) + scores
Run

```
python src/learn_model.py config/<<config_file>>
```
Machine learning algorithms

- SVR
- SVC
- LassoCV
- LassorLars
- LassorLarsCV
- GP (implemented using GPy - need some code update)
- CRF (implemented using CRFsuite)
Adding a machine learning algorithm

Example using an algorithm from scikit-learn
Adding a machine learning algorithm

Example using an algorithm from scikit-learn

- Algorithm: **Ridge**: Linear least squares with l2 regularization
Adding a machine learning algorithm

Example using an algorithm from scikit-learn

- Algorithm: **Ridge**: Linear least squares with L2 regularization
- Package: `sklearn.linear.model.Ridge`
Exemple using an algorithm from scikit-learn

- **Algorithm:** Ridge: Linear least squares with l2 regularization
- **Package:** `sklearn.linear.model.Ridge`
- **Idea:** include the algorithm on the available code
Adding a machine learning algorithm

learn_model.py

- Main class of QuEst++ machine learning module
Adding a machine learning algorithm

learn_model.py
- Main class of QuEst++ machine learning module
- Method: `set_learning_method(config, X_train, y_train)`
Adding a machine learning algorithm

learn_model.py

- Main class of QuEst++ machine learning module
- Method: `set_learning_method(config, X_train, y_train)`
- Create estimators for the new algorithm
Folder: **config**
Configuration file

- Folder: `config`
- Files follow the YAML format
Configuration file

- Folder: `config`
- Files follow the YAML format
- Open the file `svr.cfg` to see an example
Configuration file

- Folder: `config`
- Files follow the YAML format
- Open the file `svr.cfg` to see an example

Create a new file called `ridge.cfg` and follow the structured YAML to provide parameters for the model
Run

python src/learn_model.py config/ridge.cfg
Quality Estimation of Machine Translation

Recent advances, Challenges and Software

Lucia Specia and Carolina Scarton

{l.specia, c.scarton}@sheffield.ac.uk

Alicante, 21 January 2016