EXPERT

Project funded by the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme FP7/2007-2013/ under REA grant agreement no. 317471.

Project reference: 317471

Project full title: EXPloiting Empirical appRoaches to Translation

D5.2: Framework for learning from human translators

Authors: Varvara Logacheva (USFD), Lucia Specia (USFD), Chris Hokamp (DCU)

***Document Number*:** EXPERT_D8.1_20150521

***Distribution Level*:** Public

***Contractual Date of Delivery*:** 30.10.14

***Actual Date of Delivery*:** 21.05.15

Contributing to the Deliverable: WP5

WP Task Responsible: USFD

EC Project Officer: Concepcion Perez-Camaras

D5.2: Framework for learning from human translators

Varvara Logacheva, Lucia Specia, Chris Hokamp

Contents

Abstract

The topic of this research is the incorporation of human feedback of different types into machine translation (MT). The aim is to reduce the human effort spent on feedback generation, i.e., to find effective ways of use of small amounts of feedback. The feedback is incorporated into MT indirectly: it is used to train additional systems that somehow contribute to an MT system quality. We train a system that performs quality estimation (QE) of automatically translated texts and explore two scenarios of its use: we exploit it for data selection and for translation improvement.

1 Introduction

The latest advances in statistical machine translation (**SMT**) have given the hope to use MT not only for gisting (getting the approximate content of a document written in the unknown language), but also for distribution of the translated text. Since statistical MT still has numerous flaws, before distribution it needs to be proofread and maybe edited by a professional translator.

Post-editing is supported by various computer-aided translation (**CAT**) tools. Their main purpose is to improve translator's productivity by translating some bits of text automatically. The first technique to be incorporated into CAT tools were translation memories, which matches new segments to translate against a database of translations, returning full or partial matches to translators [Lagoudaki, 2006].

A more recent direction is to supplement CAT tools with machine translation systems, so that the translator's work is to post-edit the provided translation suggestions. However, MT output, unlike that of translation memories, can be erroneous, and therefore the user usually has to correct errors. The organisation of this process is often criticised by the users of such systems: the translators have to repeatedly correct the same errors in translation systems output over time. The indirect feedback given by correcting translations is not taken into account. The work package 5 (**WP5**) of the **EXPERT** project is aimed at addressing the lack of interaction between a human translator and an MT system. In particular, this report deals with the task of incorporating user feedback into MT system in order to improve its quality and allow user to take better advantage of it.

The described problem has been partially solved by the online retraining paradigm for MT. An MT system with online retraining generates a translation variant and passes it to a CAT system, which fetches users' corrections and passes them back to the MT system, so they are taken into account immediately. However, this system has a number of limitations: it is effective only for adapting an MT system to a narrow domain, whereas general-purpose MT systems which receive feedback and corrections from many different users do not benefit from this technique.

Hence, in the scope of WP5 we aim at finding new ways of the incorporation of user feedback into MT systems. In our preliminary experiments we faced

the lack of human-corrected MT data: although sufficient to adapt a system to a new domain, it is not able to significantly improve the MT output by using human feedback as additional training data. Thus, we should incorporate the human data into MT indirectly, i.e. to use this data to train an additional system which will somehow contribute to MT quality. We chose to use the available data to train a quality estimation (**QE**) system. Quality estimation is the prediction of quality of an automatically translated sentence. This system does exactly what we would like to achieve: it simulates the human labelling of automatic translation, i.e. it deals with the lack of data.

In addition to that, a QE system can accept various feedback types as training. Post-edition is one of the most natural feedback types as it is acquired via the translation process and its collection does not require any additional arrangements. However, editing a text is time-consuming and requires an editor to be proficient in both source and target languages. Moreover, due to confidentiality issues the post-editions done by professional translators often cannot be made available for third-party organisations. Therefore, the post-editions of high quality cannot be generated in big number, and crowd-sourced post-editions are unreliable because we have no information about the editors.

QE systems relax the requirement to the training data. They do not need the corrected version of the translation, but are primarily interested in what words/phrases of the initial automatically translated sentences are correct or incorrect. Therefore, instead of post-editions a QE system can be trained on texts marked with word-, phrase- or sentence-level quality labels. These labels can be of different granularity depending on the task — from binary labels to fine-grained classes of errors (e.g. MQM [Lommel et al., 2014]). The acquisition of these labels takes much less effort and does not require annotators to be professional translators, so much larger amount of data can be collected.

The predicted quality scores can be incorporated into MT pipeline in a number of ways. In this report we describe two strategies their usage:

- the use of a QE system to choose the training data for an MT system,
- the use of a QE system to inform an MT system of low-quality segments in translation and thus prevent it from generating erroneous phrases.

The report is organised as follows. We describe our feedback-motivated active selection strategy and provide the results of initial experiments in section 2. In section 3 we consider the second scenario of incorporation of human feedback: the use of a quality estimation system to improve the automatic translation quality. We give the conclusions in section 4.

2 Active Learning

The most obvious way of the use of post-editions is to add them as training data to an MT system. However, the post-editing data usually exists in small amount and does not have any visible impact on the system quality. Therefore

we decided to use post-editions to define the segments that are the most difficult for translation and are usually translated incorrectly. If an MT system receives many examples of such sentences translated correctly, it will produce better translations.

In the described experiments we use the post-editions as training data for a quality estimation system. When trained on a relatively small amount of sentences post-edited by humans, it can define the quality of an arbitrary translation without having its reference. Therefore, we can choose the sentences that are likely to evoke errors, and provide their valid translations to eliminate these errors.

2.1 Active learning strategy

Our AL sentence selection strategy relies on quality estimation (QE). QE is aimed at predicting the quality of a translated text (in this case, a sentence) without resorting to reference translations. It considers features of the source and machine translated texts, and an often small number (a few hundreds) of examples of translations labelled for quality by humans to train a machine learning algorithm to predict such quality labels for new data.

We use the open source QE framework QuEst [Specia et al., 2013]. In our settings it was trained to predict an HTER score [Snover et al., 2006] for each sentence, i.e., the edit distance between the automatic translation and its human post-edited version. QuEst can extract a wide range of features. In our experiments we use only the 17 so-called *baseline features*, which have been shown to perform well in evaluation campaigns [Bojar et al., 2013]:

- number of tokens in sentences
- average token length
- language model probabilities for source and target sentences
- average number of translations per source word
- percentage of higher and lower frequency ngrams in source sentence based on MT training corpus
- number of punctuation marks in source and target sentences.

Similarly to [Ananthakrishnan et al., 2010], we assume that the most useful sentences are those that lead to larger translation errors. However, instead of looking at the ngrams that caused errors — a very sparse indicator requiring significantly larger amounts of training data, we account for errors in a more general way: the (QuEst predicted) percentage of edits (HTER) that would be necessary to transform the MT output into a correct sentence.

Therefore, for the retraining we choose the sentences which have the highest predicted HTER score, i.e. the lowest quality.

2.2 Experiments

2.2.1 Datasets and MT settings

For the AL data selection experiment, two datasets are necessary: parallel sentences to train an initial, baseline MT system, and an additional pool of parallel sentences to select from. Our goal was to study potential improvements in the baseline MT system in a realistic "human in the loop" scenario, where source sentences are translated by the baseline system and post-edited by humans before they are added to the system. As it has been shown in [Potet et al., 2012], post-editions tend to be closer to source sentences than freely created translations. One of the research questions was to investigate whether they would be more useful to improve MT quality.

We chose the biggest corpus with machine translations and post-editions available to date: the LIG French–English post-editions corpus [Potet et al., 2012]. It contains 10,881 quadruples of the type: <*source sentence, reference translation, automatic translation, post-edited automatic translation*>. Out of these, we selected 9,000 as the pool to be added to be baseline MT system, and the remaining 1,881 to train the QE system for the experiments with AL. For QE training, we use the HTER scores between MT and its post-edited version as computed by the TERp tool.[1]

We use the Moses toolkit with standard settings[2] to build the (baseline) statistical MT systems. As training data, we use the French–English News Commentary corpus released by the WMT13 shared task [Bojar et al., 2013]. For the AL experiments, the size of the pool of additional data (10,000) poses a limitation. To examine improvements obtained by adding fractions of up to only 9,000 sentences, we took a small random subset of the WMT13 data for these experiments (Table 1). Although these figures may seem small, the settings are realistic for many language pairs and text domains where larger data sets are simply not available.

Corpora	**Size (sentences)**
Initial data (baseline MT system)	
Training - subset of News Commentary corpus	10,000
Tuning - WMT newstest-2012	3,000
Test - WMT newstest-2013	3,000
Additional data (AL data)	
Post-editions corpus:	10,881
- **Training QE** system	1,881
- **AL pool**	9,000

Table 1: Datasets

[1] `http://www.umiacs.umd.edu/~snover/terp/`

[2] `http://www.statmt.org/moses/?n=Moses.Baseline`

We should also note that all the used data belongs to the same domain: the LIG system which produced sentences from the post-editions corpus was trained on Europarl and News commentary datasets [Potet et al., 2010], the post-edited sentences themselves were taken from test sets released for WMT shared tasks of different years. Our baseline system is also trained on a fraction of the News commentary corpus. Finally, we tune and test all our systems on WMT shared task test sets choosing ones that do not have intersections with post-editions corpus.

Furthermore, the fact that LIG system and our baseline system were trained on the same corpus justifies our experiments with post-edited data. Namely, although our baseline system is not the system whose output was post-edited, it is similar to the LIG system.

2.2.2 Post-editions vs References

In order to compare the impact of post-editions and reference translations on MT quality, we added these two variants of translations to baseline MT systems of different sizes, including the entire News Commentary corpus. The figures for BLEU [Papineni et al., 2002] scores in Table 2 show that adding post-editions results in significantly better quality than adding the same number of reference translations.[3] This effect can be seen even when the additional data corresponds to only a small fraction of the training data.

Baseline corpus (sentences)	Results (BLEU)		
	Baseline	**Ref**	**PE**
150,000	22.41	22.95	**23.21**
50,000	20.22	20.91	**22.01**
10,000	15.09	18.65	**20.44**

Table 2: Influence of post-edited and reference translations on MT quality. **Ref**: baseline system with added free references, **PE**: baseline system with added post-editions.

In addition to that, it does not matter which MT system produced the translations which were then edited to acquire the post-edition corpus. Even if the output of a third-party system was used (as in the present case), it improves the quality of machine translations. We assume, that since post-editions tend to be closer to original sentences, than free translations [Potet et al., 2012], they can help produce better alignments.

In fact, this usefulness of post-editions from a third-party MT system is not unconditional. The LIG system used to generate the translations that were then post-edited is similar to our system: it was trained with the same Moses toolkit

[3]These systems use the whole post-editions set (10,881 sentences) as opposed to 9,000-sentence subset which we use further in our AL experiments. Therefore the figures reported in this table are higher than those in subsequent sections.

with only minor differences from our version. The data used for training — Europarl corpus[4] — belongs to similar domain.

We believe that the similarity of training data and the similar technologies used for training are the sufficient conditions for a third-party MT system to generate useful post-editions.

2.2.3 AL settings

The experimental settings for all methods are as follows. First, a baseline MT system is trained. Then a batch of 1,000 sentences is selected from the data pool with an AL strategy, and the selected data is removed from the pool. The MT system is rebuilt using a concatenation of the initial training data and the new batch. The process is repeated until the pool is empty, with subsequent steps using the MT system trained on the previous step as a baseline. The performance of each MT system is measured in terms of BLEU scores.

We use the following AL strategies:

- **QuEst**: our confidence-based method described in previous section.
- **Random**: random selection of sentences.
- **Oracle-HTER**: oracle-based selection based on true HTER scores of sentences in the pool, instead of the QuEst estimated HTER scores.
- **Oracle-Error**: another oracle-based strategy that chooses the sentences with the largest amount of errors (HTER score without normalization by sentence length)
- **Length**: selection of the longest sentences
- **Ranking**: another confidence-based AL strategy described in [Ananthakrishnan et al., 2010].

2.2.4 AL results

Our initial results in Figure 1 show that our selection strategy (`QuEst`) consistently outperforms the `Random` selection baseline.

In comparison with previous work, we found that the error-based `Ranking` strategy performs closely to `Random` selection, although [Ananthakrishnan et al., 2010] reports it to be better. Compared to `QuEst`, we believe the lower figures of the `Ranking` strategy are due to the fact that the latter considers features of only one type (source ngrams), whereas QuEst uses a range of different features of the source and translation sentences.

Interestingly, the `Oracle-HTER` method under-performs our QE-based method, although we expected the use of real HTER scores to be more effective. In order to understand the reasons behind such behaviour, we examined the batches selected by `QuEst` and `Oracle-HTER` strategies more closely.

[4] `http://statmt.org/europarl/`

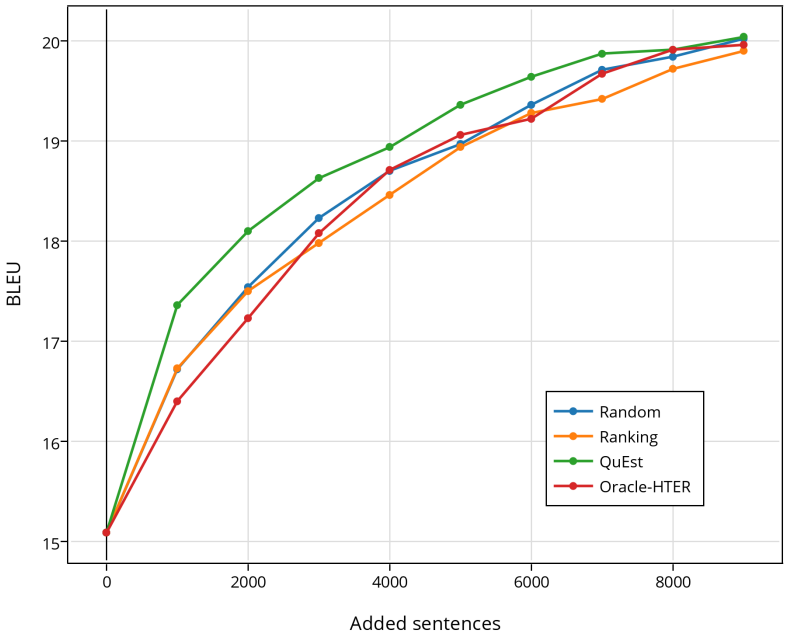

Figure 1: Performance of MT systems enhanced with data selected by different AL strategies

We found that the **distribution of sentence lengths** in batches by the two strategies is very different (see Figure 2). While in batches selected by `QuEst` the average sentence length steadily decreases as more data is added, in `Oracle-HTER` batches the average length was almost uniform for all batches, except the first one, which contains shorter sentences.

This is explained by HTER formulation: HTER is computed as the number of edits over the sentence length, and therefore in shorter sentences every edit is given more weight. For example, the HTER score of a 5-word sentence with one error is 0.2, whereas a sentence of 20 words with the same single error has a score of 0.05. However, it is doubtful that the former sentence will be more useful for an MT system than the latter. Regarding the nature of length bias in the predictions done by `QuEst` system, sentence length is used there as a feature, and longer sentences tend to be estimated as having higher HTER scores (i.e., lower translation quality).

Thus, sentences with the highest HTER can actually be not the most useful, which makes the **Oracle** strategy inferior to **QuEst**. Moreover, longer sentences

chosen by our strategy simply provide more data, so their addition might be more useful even regardless of the amount of errors.

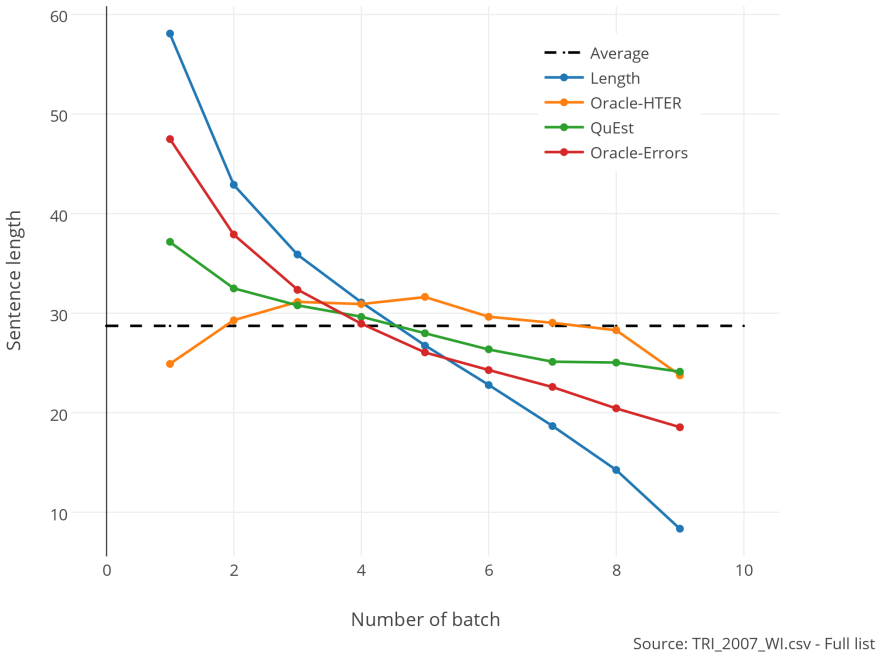

Figure 2: Number of words in batches selected by different AL strategies

This seems to indicate that the success of our strategy might not be related to the quality of the translations only, but to their length. Another guess is that sentences selected by **QuEst** might have more errors, which means that they can contribute more to an MT system.

2.2.5 Additional experiments

In order to check these hypotheses, we conduct two other sets of AL experiments: (i) a selection strategy that chooses longer sentences first (denoted as `Length`) and (ii) a selection strategy that chooses sentences with larger numbers of errors first (`Errors`).

Figure 3 shows that a simple length-based strategy yields better results than any of the other tested strategies. Therefore, in cases when the corpus has sufficient variation in sentence length, length-based selection might perform at least as well as other more sophisticated criteria. The experiments with

confidence-based selection described in [Ananthakrishnan et al., 2010] were free of this length bias, as sentences much longer or shorter than average were deliberately filtered out.

Interestingly, results for the `Errors` strategy are slightly worse than those for `QuEst`, although the former is guaranteed to choose sentences with the largest number of errors and has even stronger length bias than `QuEst` (see figure 2). Therefore, the reasons hypothesised to be behind the superiority of `QuEst` over `Oracle` (longer sentences and larger number of errors) are actually not the only factors that influence the quality of an AL strategy.

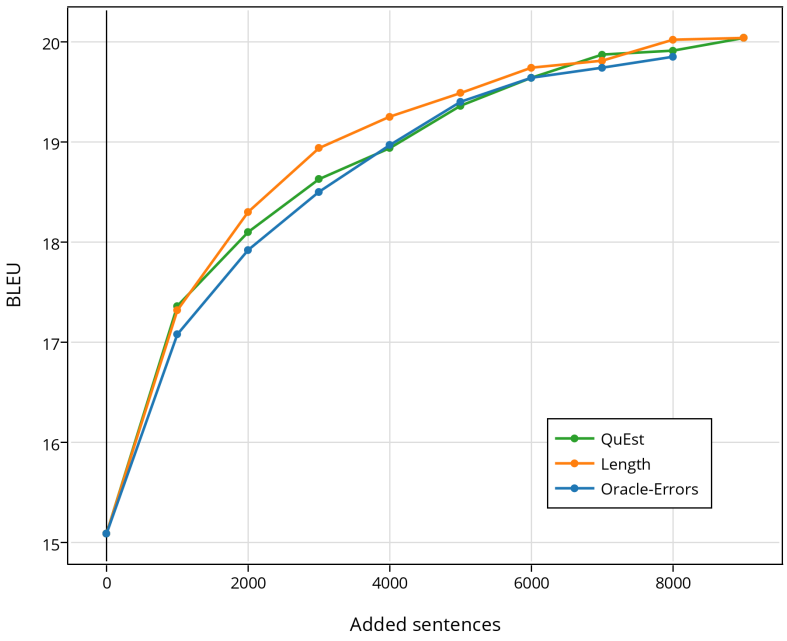

Figure 3: Comparison of our QuEst-based selection with a length-based selection

2.2.6 Length-independent results

The length bias of some AL strategies for MT has already been noticed by other researchers: [Bloodgood and Callison-burch, 2010] report the learning curve for length-based sentence selection analogous to one in figure 3. They also show that in length-independent representation the length-based strategy turns out to perform poorly.

The length bias could be eliminated by filtering out sentences that are longer or shorter than some threshold, as it was done in [Ananthakrishnan et al., 2010]. Unfortunately, the LIG corpus used here as a pool is too small to filter it out. Therefore we prefer to follow [Bloodgood and Callison-burch, 2010] and represent the result of the same experiments with respect to data size in words, not in sentences.

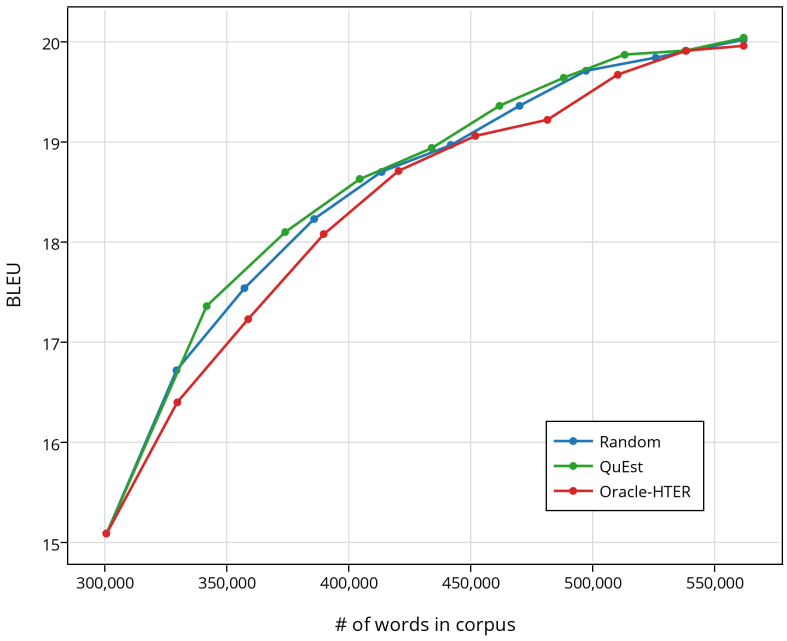

Figure 4: Active learning quality plotted with respect to data size in words: **QuEst** vs **Oracle** strategies.

We plot the results of **Length** and **Error** strategies separately to make the graphs readable. Figure 5 shows that in length-independent representation, these are very close and both underperform the **QuEst** strategy.

Here our experience echoes the results of [Mohit and Hwa, 2007], where the authors propose the idea of *difficult to translate phrases*. It is assumed that extending an MT system with phrases that can cause difficulties during translation is more effective than simply adding new data and re-building the system. Due to the lack of time and human annotators, the authors extracted difficult phrases automatically using a set of features: alignment features, syntactic features, model score, etc. Conversely, we had the human-generated information

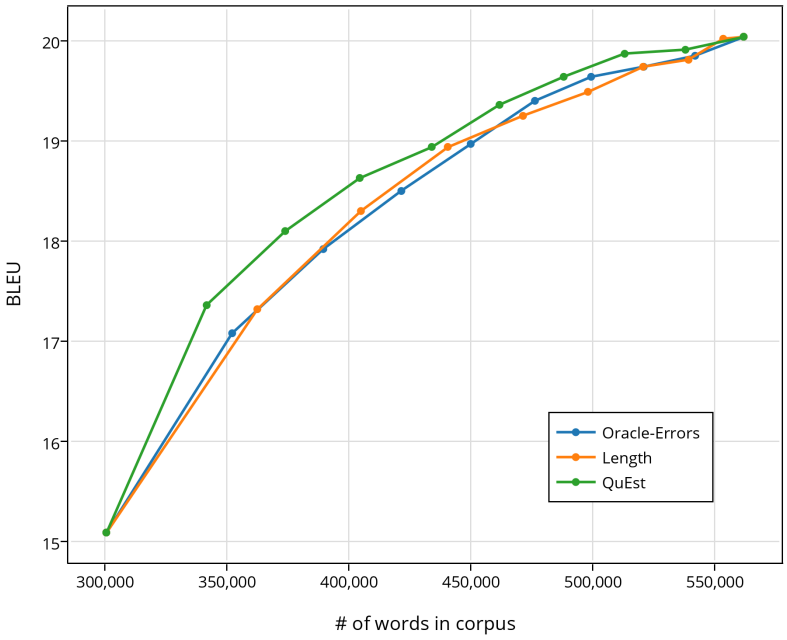

Figure 5: AL quality plotted with respect to data size in words: **QuEst** vs **Length** and **Errors** strategies.

on what segments have been translated incorrectly. We suppose that the use of this knowledge as part of our AL strategy would give us an upper bound for our AL method results. However, it turned out that prediction based on multiple features is more reliable than precise information on quality, which accounts for only one aspect of data.

2.3 Analysis

The experiments have shown several phenomena which we would like to explore:

- Post-editions are consistently better than references for MT training
- QuEst-based sentences selection performs better than random data selection
- predicted HTER scores (QuEst) are more useful for data selection than real HTER scores

2.3.1 Post-editions vs references

In order to understand the advantages of post-editions over references we examine the number of out-of-vocabulary (OOV) words in the test set when it is translated by the different systems. Namely, we considered two parameters:

- Number of words in the test set which do not occur in training data (referred to as *corpus-OOV*);
- Number of words in the test set which do not occur in phrase table (*phrase-OOV*).

The former is usually larger than the latter, because a word can be met in the training corpus only once, not be aligned with any word on the target side and not get to the phrase table.

The comparison of these parameters for systems trained on post-edited translations and those trained on reference translations led to interesting results. Figure 6 shows that while *corpus-OOV* is almost identical for corpora containing references and post-editions, the *pt-OOV* rate for systems built from post-editions is consistently lower than that of systems built from references.

2.3.2 QuEst vs Random

Figure 7 shows how the number of OOV words in the test set naturally decreases as more data is added to MT system. This reduction is faster for both *corpus-OOV* and *phrase-OOV* when our error-driven strategy is used as opposed to random selection. Therefore, `QuEst` seems to predict higher HTER scores for sentences that contain more uncovered words and thus we implicitly attempt to increase vocabulary variety, which is widely used in other active learning work, e.g. [Eck et al., 2005].

2.3.3 QuEst vs Oracle

The **Oracle** strategy was shown to perform worse than both **QuEst**-based strategy and random. In order to explain this result we should consider the essence of HTER score. HTER is the edit distance between the automatically translated sentence and its manual post-edition. So the sentences with high HTER need more post-editing and hence contain some phrases that are not represented in the phrase table of our MT system.

However, the list of most common substitutions met in the corpus does not give a clue to what phrases are absent in the system: the words corrected more often are function words (prepositions, determiners, pronouns). The table 3 gives the statistics of the most common substitution patterns in the post-edited data. These observations agree with other experiments with post-editions: [Wisniewski, 2013] gives very similar statistics of the most common substitutions extracted from post-editions of Trace corpus, [Groves and Schmidtke, 2009]

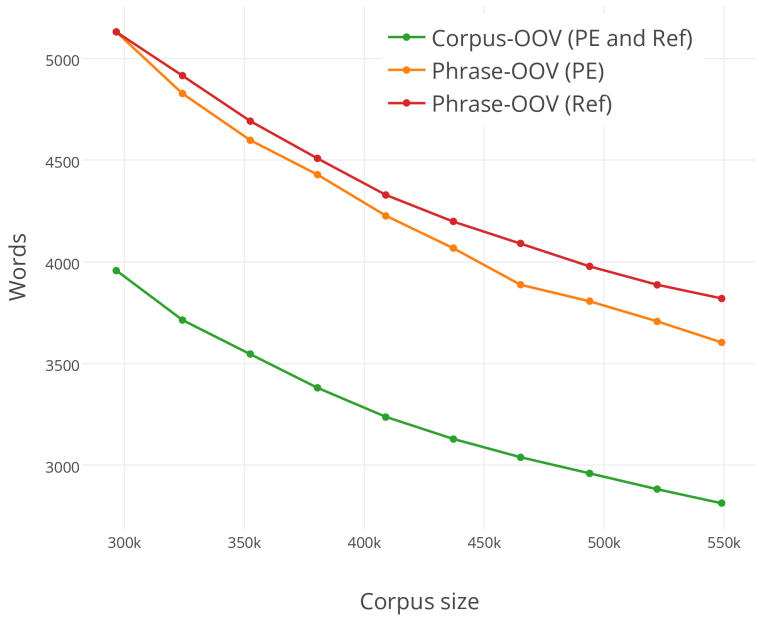

Figure 6: Comparison of the impact of post-editions and references on the OOV rate.

show that when translating into German and French the most frequent corrections are insertions, deletions or substitutions of determiners and prepositions. Based on these evidences we can deduce that the majority of corrections fix grammar errors and phrases structure. Such corrections are too heterogeneous to improve the quality of an MT system much if injected in small batches.

We have also noticed that the correlation of QuEst predictions with MT quality metrics is very low. The predictions do not even correlate with HTER score, although this is the metric QuEst is trying to predict. The table 4 shows the correlation coefficients between QuEst predictions for LIG corpus and HTER and sentence-level BLEU (sBLEU) scores.[5] The metric values were counted for both post-editions and reference translations (denoted as *pe* and *ref*, respectively), post-editions predictably give higher correlation.

The HTER and BLEU metrics correlate well with each other (two bottom rows of the table). Unfortunately, both of them have only weak correlation with QuEst. Surprisingly, BLEU correlates slightly better with QuEst predictions.

[5]The negative correlation values appear when we count the correlation of two metrics, one of which gives to the best sentences the value of 1 (BLEU), and the other metric – the value of 0 (HTER, QuEst).

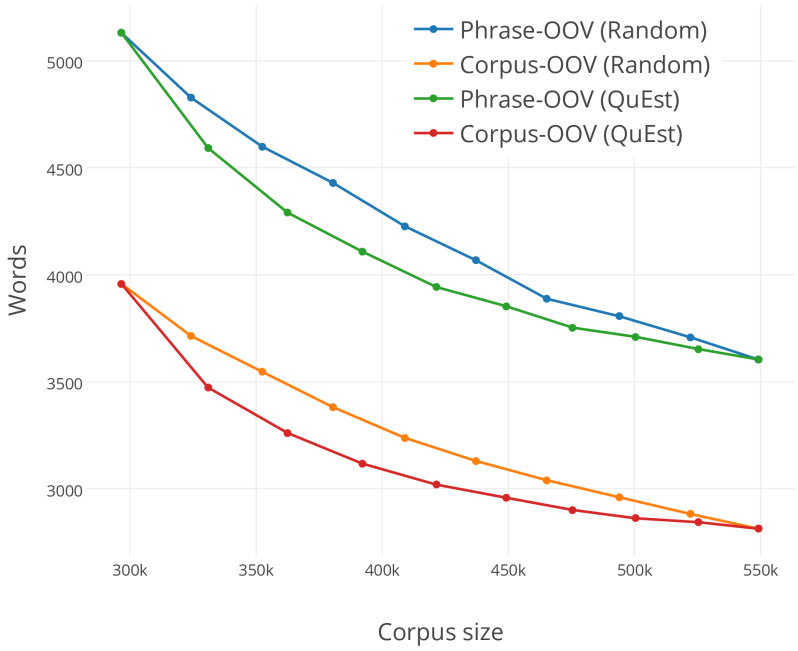

Figure 7: Comparison of reduction of OOV words for different active selection techniques.

The differences of sentence lengths statistics in QuEst-ranked and HTER-ranked lists, as well as the low correlation of QuEst and HTER scores give the evidence to the fact that QuEst system predicts scores that have little in common with the real quality labels. However, these predictions are useful for the MT system training.

Apparently, some of features used in QuEst can detect the sentences that are useful for the retraining.

2.4 Conclusions

We have introduced a data selection strategy for MT training data collection. It selects sentences which can potentially improve the performance of an MT system, in other words, sentences which are the most useful for MT system training. We assume that the most useful sentences are those which cause more errors in a baseline system, and to judge that we look at both source and machine translated sentences.

Substitution pattern			Count
it	$\longrightarrow$	he	214
the	$\longrightarrow$	's	167
of	$\longrightarrow$	from	123
its	$\longrightarrow$	his	123
'	$\longrightarrow$	"	115
of	$\longrightarrow$	in	112
to	$\longrightarrow$	of	109
in	$\longrightarrow$	to	106
in	$\longrightarrow$	of	105
a	$\longrightarrow$	an	102

Table 3: Substitution patterns

	Pearson correlation	Spearman rank correlation
QuEst vs HTER (pe)	0.317341	0.352838
QuEst vs HTER (ref)	0.184328	0.237578
QuEst vs sBLEU (ref)	-0.258091	-0.265386
QuEst vs sBLEU (pe)	-0.336811	-0.347072
HTER vs sBLEU (ref)	-0.712004	-0.835811
HTER vs sBLEU (pe)	-0.906598	-0.941111

Table 4: Correlations of metrics

The selection strategy is based on a quality estimation metric which predicts HTER scores. We choose sentences with the highest predicted HTER (the largest proportion of editing needed) and enhance the training data with their translations or post-editions. This strategy has been shown to outperform random selection.

However, we found that it was successful mostly due to its tendency to rate long sentences as having lower quality. A length-independent representation of the results showed that an oracle selection is less effective than our quality-based strategy, which we believe to be due to the nature of corrections and small size of the post-edition corpus. In addition to that, another oracle selection based on the amount of errors and length-based selection show poor results when displayed in length-independent mode.

We also show that an MT system with added post-edited sentences consistently outperforms and MT system with added reference translations for the same sentences. This finding suggests that we could reduce the translator's effort in creating data for active learning while getting even better improvements in the quality of the resulting MT system.

3 Quality estimation for the improvement of MT

In our initial experiments we explored the possibility of using automatic translations post-edited by humans as the additional training data for an MT system. The initial experiments with the post-editions showed that the existing post-editing data is principally limited in size: we showed that in order to significantly improve the quality of translation with additional post-edited sentences, their number should be comparable with the overall amount of training data. Since the MT systems usually use around 1,000,000 of sentence pairs, collection of the sufficient amount of post-edition is infeasible.

Therefore, we need to consider a different scenario of using the post-edits in MT framework. As it was outlined before, the post-editions can be used in MT either directly or indirectly. As the direct use didn't prove successful, we decided to use post-editions to train a separate system which will improve the quality of MT output.

We decided to use the available post-editions to train a quality estimation (**QE**) system. This system aims at evaluating the quality of an automatically translated sentence without having an oracle translation with which it could be compared. Instead, it uses a number of features of the sentence and the MT system which produced it.

The high-level sense of the incorporation of quality scores into an MT system is the following: a QE system is trained on a small amount of human-labelled data and then propagates these labels on the unseen data, thus simulating the human labelling. Therefore, in the ideal case a QE system should work as a human annotator and inform an MT system about the quality of generated words and sequences, so that the MT system can change the output accordingly.

The results of QE can be incorporated into MT in one of the following ways:

- quality score can be used for N-best list reordering on its own or in combination with the system score.
- quality score can be used N-best list re-ranking using the quality score as an additional feature.
- second-pass decoding: generation of a translation lattice consisting of the highest-scoring translations.
- incorporation of the quality score into the decoder.

While the former two scenarios are suitable for both sentence- and word-level QE, the latter two need the labels for every word in a translated sentence.

3.1 Word-level QE system

While there exists an open-source sentence level QE system QuEst [Specia et al., 2013], there are no publicly available QE systems that perform

analysis at the word level. Therefore, we developed a word-level QE toolkit Marmot[6]. This was the joint work of the teams of University of Sheffield and Dublin City University.

A QE system should be able to perform three main tasks: extract features from the data, train a model, and perform tagging of new data using this model. Marmot is designed to perform all of these tasks in a flexible, efficient and transparent manner. Marmot pipelines allow users to go from raw training data to trained and evaluated models with minimal configuration.

Marmot is written in Python. By integrating mature Python libraries that implement common machine learning algorithms and NLP tasks at the pre-processing and feature extraction stages, Marmot takes advantage of stable and well-maintained libraries. The main feature of the Marmot is its flexibility. Because of the modular structure, users can easily implement new parsers, data representations, and features which fit their particular use cases.

3.1.1 System architecture

Figure 8 shows the architecture of the pipeline used by Marmot. The grey fields in the picture denote the stages of pre-processing or training which require specification of pre-defined parameters and which are likely to be customised by a user. The white blocks inside the grey fields denote user-determined procedures.

Each phase in the pipeline for an experiment is specified in the configuration file as the path to a Python class or function. Therefore, the framework is agnostic about the actual functions that it calls, allowing the user to create a custom pipeline without writing any code, simply by specifying which parsers, feature extractors, and learning method(s) they wish to use. User-defined functions do not need to be included into the source code of the system, and no changes to the source code need to be made if a user provides a valid path to the function and makes sure it returns the output accepted by the next stages.

Parsing input data is the first of the configurable stages. Data can be provided in many different formats, such as parallel corpora in one or more files, JSON, or XML, and new parsers are straightforward to implement.

Feature extractors convert the input data into one or more features[7]. The extracted features can be passed directly into the model training module or persisted to a file for use with other machine learning frameworks.

The extractors are independent of each other, the set of used feature extractors can be varied according to user's needs. The current version of Marmot contains the most meaningful features from [Luong et al., 2014] and can be extended by a user. The extracted feature set can be either passed on to the training module or dumped to file in the format accepted by CRF++ and some other machine learning tools.

There is a wide range of possibilities of a model training. Classification can be performed with one of classifiers defined in `scikit-learn`[8]. Sequence

[6] http://github.com/qe-team/marmot

[7] features may be scalar and/or categorical

[8] `http://scikit-learn.org/`

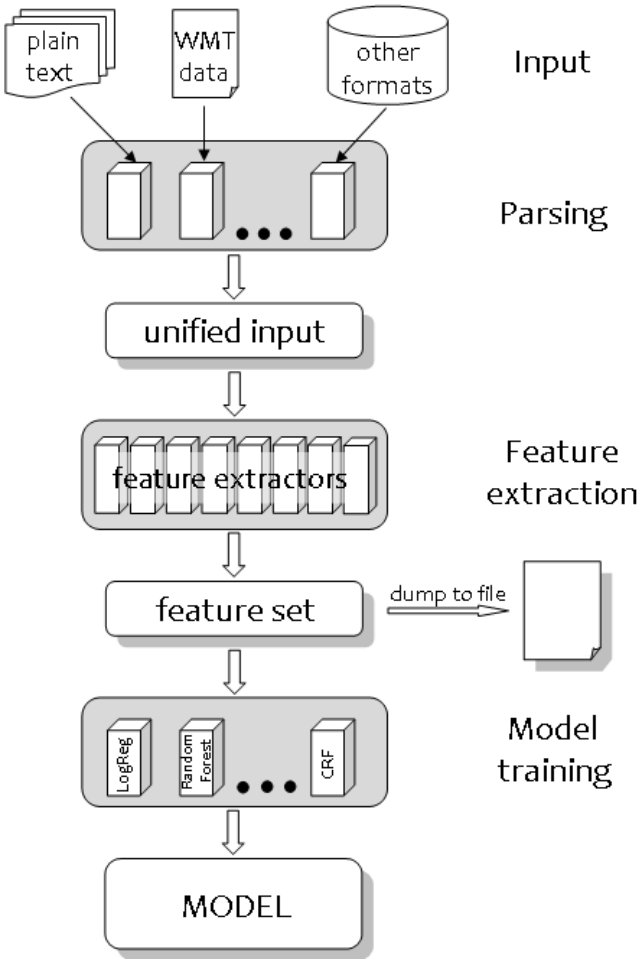

Figure 8: System architecture

labelling is done with functions of a Python module `pystruct`[9]. Analogously to parsing and feature extraction, the ML method used is specified in the config file, so any classifier from `scikit-learn` can training module can also easily integrate any other ML libraries if they are written in Python.

The standard approach is to train a classifier for all the training examples. In addition to that, our system is able to train a separate classifier for every token that occurs in the training data. This approach allows to take into account differences in distributions of errors throughout words and word classes.

[9] `https://pystruct.github.io/`

3.1.2 Experiments

We evaluated our tool on WMT-14 Quality estimation task[10] — classification of words in the translated sentence as "good" or "bad", and compared its performance with that of other systems that participated in the shared task. The results were evaluated in terms of F1 score, the systems were ranked according to the F1 score of the "bad" class.

We tested Marmot with three different settings:

- words are classified independently of each other with Random Forest classifier,
- sequences of words are labelled with Conditional Random Fields (CRFs)
- sequences of words are labelled with Maximum Entropy Markov Model (MEMM) using only the features that can be extracted from the current word and its left context. This setting was tested in order to understand if the QE can be used in an MT system at decoding time, when no right context of the evaluated sentence is available.

	F1-BAD ↑	F1-GOOD	Average F1
FBK-UPV-UEDIN/RNN	0.487283	0.693375	0.619968
LIMSI	0.473157	0.678724	0.605504
LIG feature selection	0.444735	0.740961	0.63545
LIG all features	0.441074	0.746503	0.637714
FBK-UPV-UEDIN/RNN+tandem+crf	0.426251	0.729763	0.621657
Marmot CRF	0.425964	**0.761358**	**0.64186**
Marmot classification	0.420294	0.702252	0.601823
Marmot MEMM, left-hand features	0.411212	0.707165	0.601751
DCU-GLM	0.350838	0.748412	0.606803
DCU-GLMd	0.328942	0.753697	0.602406

Table 5: Comparison of the Marmot tool with other word-level QE systems

The table 3.1.2 shows that the performance of Marmot is comparable to the quality of other existing systems. Moreover, in the sequence labelling setting it achieved the highest average F1 and F1 for the "good" class. These metrics are not considered when ranking the QE systems, but they should not be completely discarded.

Notice that even the MEMM version that uses only the left contexts performs just slightly worse than the systems that used the full set of features. This result is encouraging as it means that our QE system can be incorporated into the MT decoding process with only minor loss of quality.

[10] `http://statmt.org/wmt14/quality-estimation-task.html`

3.2 Data generation for the QE system training

We trained the QE system on the data provided for the WMT-14 shared task: 1,957 sentences for the English–Spanish language pair. This is a very small training set, so the performance of our system could be improved if we had more data available. However, achieving such data is expensive, so we decided to consider automatic generation of new sentences labelled with word-level quality labels.

The easiest choice for artificial data generation is to create a sentence by taking all or some of its words from a probability distribution of words in some monolingual corpus. The probability can be defined for unigrams only or conditioned on the previous words. This however is a target language-only method that does not suit the QE task as the "quality" of a target word or sentence is dependent on the source sentence, and disregarding it will certainly lead to generation of spurious data.

Random target sentences based on a given source sentence could be generated with bilingual LMs. However another limitation of this approach is the assumption that all words in such sentences are wrong, which makes the data useless for word-level QE.

Alternatively, the artificial sentences can be generated using MT systems for back-translation. The target sentences are first fed to a target–source MT system, and then its output is passed to a source–target system. However, according to our experiments, if both systems are statistical the back-translation is too similar to the original sentence, and the majority of their differences are interchangeable paraphrases. Rule-based systems could be more effective, but the number of rule-based systems freely available would limit the work to a small number of language pairs.

3.2.1 A two-stage error generation method

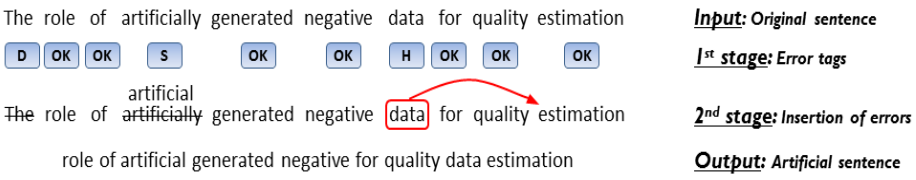

Figure 9: Example of the two-stage artificial data generation process

Methods that artificially generate entire sentences have drawbacks that make them difficult or impossible to use for QE. Therefore, following [Raybaud et al., 2011], our approach is to inject errors into otherwise correct texts. This process consists of two stages:

- labelling of a sentence with error tags,
- insertion of the errors into that sentence.

The first stage assigns an error tag to every word in a sentence. The output of this stage is the initial sentence where every word is assigned a tag denoting a type of error that needs to be incurred on this word. We use five tags corresponding to edit operations in the TERp tool: no error (**OK**), substitution (**S**), deletion (**D**), insertion (**I**) and shift (**H**). During the second stage the words in the sentence are changed according to their tag: substituted, deleted, shifted, or left in place if word has the tag **OK**. Figure 9 gives an example of the complete generation process.

Error tagging of sentences We generate errors based on a corpus of post-edited machine translations. We align translations and post-editions using the TERp tool (exact matching) and extract counts on the number of shifts, substitutions, insertions and deletions. TERp does not always capture the true errors, in particular, it fails to identify phrase substitutions (e.g. *was* $\rightarrow$ *has been*). However, since editors are usually asked to minimise the number of edits, translations and post-editions are often close enough and the TERp alignment provide a good proxy to the true error distribution.

The TERp alignments can be used to collect the statistics on errors alone or to combine the frequency of errors with the words they are incurred on. We suggest three methods of generation of an error string for a sentence:

- **bigramEG**: the *bigram* error generation that uses a bigram error model regardless of the actual words [Raybaud et al., 2011].
- **wordprobEG**: the conditional probability of an error given a word.
- **crfEG**: the combination of the bigram error model and error probability conditioned on a word. This generation method can be modelled with Hidden Markov Model (HMM) or conditional random fields (CRF).

The first model has the advantage of keeping the distribution of errors as in the training data, because the probability distributions used depend only on the frequency of errors themselves. The second model is more informed about which words commonly cause errors. Our implementation of the third method uses CRFs to train an error model. We use all unigrams, bigrams and trigrams that include the target word as features for training. This method is expected to produce more plausible error tags, but it can have the issue that the vocabulary we want to tag is not fully covered by the training data, so some words in the sentences to tag will be unknown to the trained model. If an unknown word needs to be tagged, it will more often be tagged with the most frequent tag, which is "Good" in our case. In order to avoid this problem we replace rare words in the training set with a default string or with the word class, i.e. the word's POS tag. We also consider the scenario where the POS tags are used as additional features.

Insertion of errors We consider errors of four types: **insertion**, **deletion**, **substitution** and **shift**. Word marked with the 'deletion' error tag are simply

removed. Shift errors require the distribution of shift distances which are computed based on the TERp-aligned training corpus. Substitutions and insertions require word insertion (WI) and the new words need to be drawn from some probability distribution. We suggest two methods for the generation of these distributions:

- **unigramWI**: word frequencies computed based on a large monolingual corpus.
- **paraphraseWI**: distributions of words that can be used instead of the current word in the translation. This computation is performed as follows: first all possible sources of a target word are extracted from an SMT system's translation table, then all possible targets for these sources. That gives us a confusion set for each target word.

3.2.2 Experiments

Sentence-level QE task We tested our artificially generated error on the task 1.1 of WMT-14 QE shared task — namely, the ternary classification of sentences as "good", "almost good" and "bad". The original dataset for this task contains 949 "good", 2010 "almost good", and 857 "bad" sentences, whereas the test set has 600 entries: 131 "good", 333 "almost good", 136 "bad". The results were evaluated using F1-score.

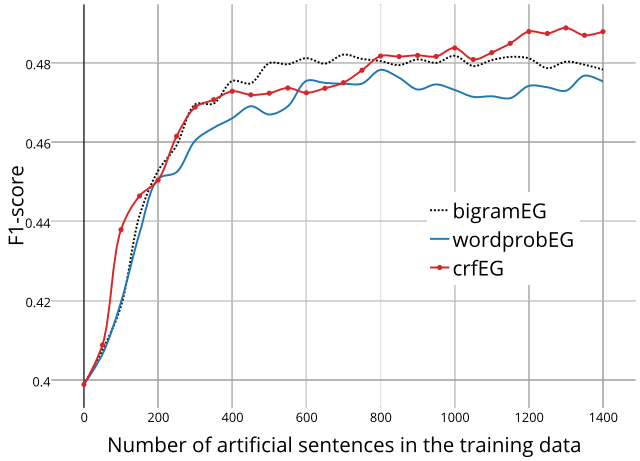

Figure 10: Ternary classification: performance of error generators

The addition of new "bad" sentences leads to an improvement in quality, regardless of the sentence generation method used. Models trained on datasets generated by different strategies display the same trend: adding up to 400 sentences results in a considerable increase in quality, while further addition of data

only slightly improves quality. Figure 10 shows the results of the experiments – here for clarity we included only the results for datasets generated with the **unigramWI**, although the **paraphraseWI** demonstrates a similar behaviour with slightly lower quality. The best F1-score of 0.49 is achieved by a model trained on the data generated with the **crf** error generator, which is an absolute improvement of 1.9% over the baseline.

Word-level QE task Here we tested the impact of the artificial data on the task of classifying individual words as "good" or "bad". The baseline set contains 47335 words, 35% of which have the tag "bad". The test set has 9613 words with the same label distribution.

All the datasets led to similar results. Overall, the addition of artificial data harms prediction performance: the F1-score goes down until 1500 sentences are added, and then levels off. The performance for all datasets is similar. However, analogously to the previous tasks, there are differences between **crfEG** and the other two error generation techniques: the former leads to faster deterioration of F1-score. No differences were observed among the word insertion techniques tested.

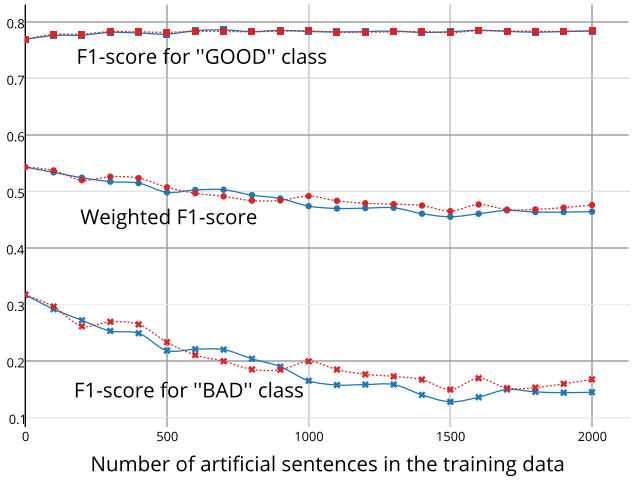

Figure 11: Word-level QE. Blue solid lines – results for **crfEG**, red dotted lines – **bigramEG**

Figure 11 shows the average weighted F1-score and F1-scores for both classes. Since all datasets behave similarly, we show the results for two of them that demonstrate slightly different performance: **crfEG+unigramWI** is shown with solid blue lines, while **bigramEG+unigramWI** is shown with dotted red lines. The use of data generated with CRF-based methods results in slightly faster decline in performance than the use of data generated with **bigramEG** or **wordprobEG**. One possible reason is that the CRF-generated datasets have

fewer errors, hence they change the original tags distribution in the training data. Therefore, test instances are tagged as "bad" less often. That explains why the F1-score of the "bad" class decreases, whereas the F1-score of the "good" class stays at the same.

To summarise our findings for word-level QE, the strategies of data generation proposed and tested thus far do not lead to improvements. The word-level predictions are more sensitive to individual words in training sentences, so the replacement of tokens with random words may confuse the model. Therefore, the word-level task needs more elaborate methods for substituting words.

3.3 Conclusions

We discovered that the direct use of human feedback in MT systems is not beneficial in many cases. Therefore, our goal is to incorporate it indirectly: we first use human feedback to train a quality estimation system which will propagate human judgements on unseen data, then use the information.

We developed a QE system Marmot which is able to extract features relevant for quality estimation and train a QE model using a number of classification or sequence labelling algorithms. Moreover, a user can easily enhance it with new feature extractors and machine learning methods.

We ran into the lack of training data for a QE system and tested a number of artificial data generation strategies. Unfortunately, the artificial data generated so far did not improve the accuracy of QE at the word level, but the sentence-level score increased for systems trained on the artificial data. It means, that we can .

The future work will include the incorporation of the sentence-level and word-level quality scores into MT at different stages: use of word-level QE at translation time to prevent generating unlikely words, use of word- or sentence-level scores after the translation to choose the best translation option from the N best MT system outputs.

4 Conclusions

The growing amount of human feedback on automatic translations suggests that it could be used for the improvement of MT systems.

There are many techniques of the incorporation of human feedback into MT systems, but they have numerous shortcomings. Firstly, they mainly focus on incremental updates of MT system components in online retraining scenario. While this direction is promising, it is not the only possible use of post-editions and feedback in general. MT systems with online retraining have quite specific properties and the way they incorporate feedback is often inapplicable in other settings.

Another related point is that the majority of research does not apply human feedback to general-purpose MT systems. That is reasonable, because the

translation agencies using highly specialised systems are more likely to produce feedback than users that use online MT systems for gisting.

Therefore, the research should concentrate on using the small amount of feedback and/or using the types of feedback that are easier to acquire than others. We have tried to address this challenge in our research:

- We exploited small corpus of post-editions to select data for MT training — i.e. to use small amount of labelled data for the selection of unlabelled data with the same properties.
- Similarly, we used this data to generate new sentences with machine-like errors in order to use them for QE system training.

Moreover, the QE systems we used in our experiments do not have to be trained on post-editions which are difficult to acquire. Since we need only the information on whether a sentence or word is correct or wrong, we could replace post-editions with quality judgements at the word or sentence level. This type of feedback is much easier to get as it does not require user to be a professional translator, and marking words/sentences as correct or wrong is faster than editing the sentence.

Our future work will incorporate the predicted quality scores into MT systems, i.e. the human feedback will be indirectly used to improve the quality of machine translation.

References

[Ananthakrishnan et al., 2010] Ananthakrishnan, S., Prasad, R., Stallard, D., and Natarajan, P. (2010). Discriminative Sample Selection for Statistical Machine Translation. In *EMNLP-2010: Conference on Empirical Methods in Natural Language Processing*, pages 626–635, Massachusetts, USA.

[Bloodgood and Callison-burch, 2010] Bloodgood, M. and Callison-burch, C. (2010). Bucking the Trend: Large-Scale Cost-Focused Active Learning for Statistical Machine Translation. In *ACL-2010: 48th Annual Meeting of the Association for Computational Linguistics*, pages 854–864, Uppsala, Sweden.

[Bojar et al., 2013] Bojar, O., Buck, C., Callison-burch, C., Federmann, C., Haddow, B., Koehn, P., Monz, C., Post, M., Soricut, R., and Specia, L. (2013). Findings of the 2013 Workshop on Statistical Machine Translation. In *WMT-2013: 8th Workshop on Statistical Machine Translation*, pages 1–44, Sofia, Bulgaria.

[Eck et al., 2005] Eck, M., Vogel, S., and Waibel, A. (2005). Low Cost Portability for Statistical Machine Translation based on N-gram Frequency and TF-IDF. In *IWSLT 2005: Proceedings of the International Workshop on Spoken Language Translation. October 24-25, 2005, Pittsburgh, PA.*

[Groves and Schmidtke, 2009] Groves, D. and Schmidtke, D. (2009). Identification and Analysis of Post-Editing Patterns for MT. In *MT Summit XII: 12th Machine Translation Summit*, Ottawa, Ontario, Canada.

[Lagoudaki, 2006] Lagoudaki, E. (2006). Translation Memories Survey 2006: Users' perceptions around TM use. *Translating and the Computer*, 28(November):1–29.

[Lommel et al., 2014] Lommel, A., Burchardt, A., Popović, M., Harris, K., Avramidis, E., and Uszkoreit, H. (2014). Using a New Analytic Measure for the Annotation and Analysis of MT Errors on Real Data. In *EAMT-2014: Proceedings of the 18th Annual Conference of the European Association for Machine Translation*, pages 165–172.

[Luong et al., 2014] Luong, N. Q., Besacier, L., and Lecouteux, B. (2014). Lig system for word level qe task at wmt14. In *Proceedings of WMT-14*, pages 335–341, Baltimore, Maryland, USA. Association for Computational Linguistics.

[Mohit and Hwa, 2007] Mohit, B. and Hwa, R. (2007). Localization of Difficult-to-Translate Phrases. In *WMT-2007: 2nd Workshop on Statistical Machine Translation*, pages 248–255, Prague, Czech Republic.

[Papineni et al., 2002] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-j. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation. In *ACL-2002*, pages 311–318.

[Potet et al., 2010] Potet, M., Besacier, L., and Blanchon, H. (2010). The LIG machine translation system for WMT 2010. In *Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR, ACL-2010*, pages 161–166, Uppsala, Sweden.

[Potet et al., 2012] Potet, M., Esperança-Rodier, E., Besacier, L., and Blanchon, H. (2012). Collection of a Large Database of French-English SMT Output Corrections. In *LREC 2012: Eighth international conference on Language Resources and Evaluation, 21-27 May 2012, Istanbul, Turkey*, pages 4043–4048.

[Raybaud et al., 2011] Raybaud, S., Langlois, D., and Smaïli, K. (2011). This sentence is wrong. Detecting errors in machine-translated sentences. *Machine Translation*, 25(1):1–34.

[Snover et al., 2006] Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A Study of Translation Edit Rate with Targeted Human Annotation. In *AMTA-2006*, pages 223–231.

[Specia et al., 2013] Specia, L., Shah, K., de Souza, J. G. C., and Cohn, T. (2013). QuEst - A translation quality estimation framework. In *ACL-2013: 51st Annual Meeting of the Association for Computational Linguistics, Demo session*, Sofia, Bulgaria.

[Wisniewski, 2013] Wisniewski, G. (2013). Design and Analysis of a Large Corpus of Post-Edited Translations: Quality Estimation, Failure Analysis and the Variability of Post-Edition. In *MT Summit XIV: 14th Machine Translation Summit*, pages 117–124, Nice, France.