The role of artificially generated negative data for quality estimation of machine translation

Varvara Logacheva
University of Sheffield
Sheffield, United Kingdom
v.logacheva@sheffield.ac.uk

Lucia Specia
University of Sheffield
Sheffield, United Kingdom
l.specia@sheffield.ac.uk

Abstract

The modelling of natural language tasks using data-driven methods is often hindered by the problem of insufficient naturally occurring examples of certain linguistic constructs. The task we address in this paper – quality estimation (QE) of machine translation – suffers from lack of negative examples at training time, i.e., examples of low quality translation. We propose various ways to artificially generate examples of translations containing errors and evaluate the influence of these examples on the performance of QE models both at sentence and word levels.

1 Introduction

The task of classifying texts as "correct" or "incorrect" often faces the problem of unbalanced training sets: examples of the "incorrect" class can be very limited or even absent. In many cases, naturally occurring instances of these examples are rare (e.g. incoherent sentences, errors in human texts). In others, the labelling of data is a non-trivial task which requires expert knowledge.

Consider the task of quality estimation (QE) of machine translation (MT) systems output. When performing binary classification of automatically translated sentences one should provide examples of both bad and good quality sentences. Good quality sentences can be taken from any parallel corpus of human translations, whereas there are very few corpora of sentences annotated as having low quality. These corpora need to be created by human translators, who post-edit automatic translations, mark errors in translations, or rate translations for quality. This process is slow and expensive. It is therefore desirable to devise automatic procedures to generate negative training data for QE model learning.

Previous work has followed the hypothesis that machine translations can be assumed to have low quality (Gamon et al., 2005). However, this is not the case nowadays: many translations can be considered flawless. Particularly for word-level QE, it is unrealistic to presume that every single word in the MT output is incorrect. Another possibility is to use automatic quality evaluation metrics based on reference translations to provide a quality score for MT data. Metrics such as BLEU (Papineni et al., 2002), TER (Snover et al., 2006) and METEOR (Banerjee and Lavie, 2005) can be used to compare the automatic and reference translations. However, these scores can be very unreliable, especially for word-level QE, as every word that differs in form or position would be annotated as bad.

Previous efforts have been made for negative data generation, including random generation of sentences from word distributions and the use of translations in low-ranked positions in n-best lists produced by statistical MT (SMT) systems. These methods are however unsuitable for QE at the word level, as they provide no information about the quality of individual words in a sentence.

In this paper we adopt a different strategy: we insert errors in otherwise correct sentences. This provides control over the proportion of errors in the negative data, as well as knowledge about the quality of individual words in the generated sentences. The goals of the research presented here are to understand the influence of artificially generated data (by various methods and in various quan-

tities) on the performance of QE models at both sentence and word levels, and ultimately improve upon baseline models by extending the training data with suitable artificially created examples. In Section 2 we further review existing strategies for artificial data generation. We explain our generation strategies in Section 3. In Section 4 we describe our experiment and their results.

2 Previous work

2.1 Discriminative language modelling

One example of task that requires low quality examples is discriminative language modelling (DLM), i.e., the classification of sentences as "good" or "bad". It was first introduced in a monolingual context within automatic speech recognition (Collins et al., 2005), and later applied to MT. While in speech recognition negative examples can be created from system outputs that differ from the reference (Bhanuprasad and Svenson, 2008), in MT there are multiple correct outputs, so negative examples need to be defined more carefully.

In Okanohara (2007) bad sentences used as negative training instances are drawn from the distribution $P(w_i|w_{i-N+1}, ..., w_{i-1})$: first the start symbol $< s >$ is generated, then the next words are taken based on the word probability given the already generated words.

Other approaches to discriminative LMs use the n-best list of the MT system as training data (Li and Khudanpur, 2008). The translation variant which is closest to the oracle (e.g. has the highest BLEU score) is used as a positive example, while the variant with high system score and low BLEU score is used as a negative example. Such dataset allows the classifier to reduce the differences between the model score and the actual quality score of a sentence.

Li et al. (2010) simulate the generation of an n-best list using translation tables from SMT systems. By taking entries from the translation table with the same source side they create a set of alternative translations for a given target phrase. For each sentence, these are combined, generating a confusion set for this sentence.

2.2 Quality estimation for MT

QE can be modelled as a classification task where the goal is to distinguish good from bad translations, or to provide a quality score to each translation. Therefore, examples of bad sentences or words produced by the MT system are needed. To the best of our knowledge, the only previous work on adding errors to well-formed sentences is that by Raybaud et al. (2011).

In (Raybaud et al., 2011), the training data for the negative data generation process consists of a set of MT hypotheses manually post-edited by a translator. Hypotheses are aligned with the corresponding post-editions using the TERp tool (Snover et al., 2008). The alignment identifies the edit operations performed on the hypothesis in order to convert it to the post-edited version: leave word as is (no error), delete word, insert new word, substitute word with another word. Two models of generation of error strings from a well-formed sentence are proposed. Both are based on the observed frequency of errors in the post-edited corpus and do not account for any relationships between the errors and the actual words. The *bigram error model* draws errors from the bigram probabilities $P(C_i|C_{i-1})$ where C_i is an error class. The *cluster error model* generates clusters of errors based on the distribution of lengths of erroneous word sequences in the training data. Substituting words are chosen from a probability distribution defined as the product of these words' probabilities in the IBM-1 model and a 5-gram LM. A model trained only on artificial data performs slightly better than one trained on a small manually annotated corpus.

2.3 Human error correction

Another task that can benefit from artificially generated examples is language learner error correction. The input for this task is text that potentially contains errors. The goal is to find these errors, similarly to QE at the word level, and additionally correct them. While the text is written by humans, it is assumed that these are non-native speakers, who possibly translate the text from their native language. The difference is that in this task the source text is a hidden variable, whereas in MT it is observed.

The strategy of adding errors to correct sentences has also been used for this task. Human errors are more intuitive to simulate as language learners explicitly attempt to use natural language grammars. Therefore, rule-based systems can be used to model some grammar errors, particularly those affecting closed class words, e.g. determiner errors (Izumi et al., 2003) or countability errors (Brockett et al., 2006).

More recent statistical methods use the distributions of errors in corpora and small seed sets of errors. They often also concentrate on a single error type, usually with closed class words such as articles and prepositions (Rozovskaya and Roth, 2010). Felice and Yuan (2014) go beyond closed class words to evaluate how errors of different types are influenced by various linguistic parameters: text domain, learner's first language, POS tags and semantic classes of erroneous words. The approach led to the generation of high-quality artificial data for human error correction. However, it could not be used for MT error identification, as MT errors are different from human errors and usually cannot be assigned to a single type.

3 Generation of artificial data

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 (as it was done for discriminative LMs). 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.

3.1 A two-stage error generation method

As previously discussed, existing methods that artificially generate entire sentences have drawbacks that make them difficult or impossible to use for QE. Therefore, following Raybaud et al. (2011) and previous work on human error correction, 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 1 gives an example of the complete generation process.

3.1.1 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* → *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

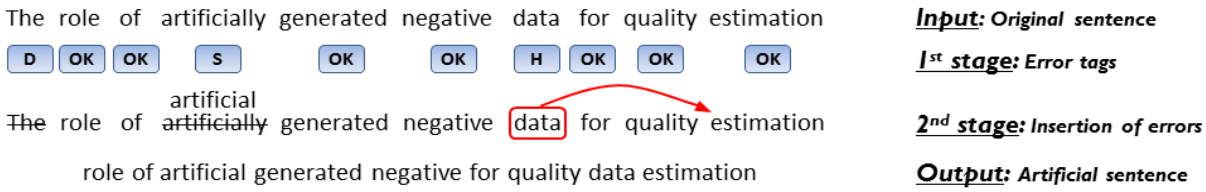

Figure 1: Example of the two-stage artificial data generation process

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 training set with a default string or with the word class, e.g. a POS tag.

3.1.2 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 a TERp-aligned 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.

4 Experiments

We conducted a set of experiments to evaluate the performance of artificially generated data on different tasks of QE at the sentence and word levels.

4.1 Tools and datasets

The tools and resources required for our experiments are: a QE toolkit to build QE models, the training data for them, the data to extract statistics for the generation of additional examples.

The for sentence-level QE we used the QUEST toolkit (Specia et al., 2013). It trains QE models using `sklearn`[1] versions of Support Vector Machine (SVM) classifier (for ternary classification task, Section 4.4) and SVM regression (for HTER prediction, Section 4.5). The word-level version of QUEST[2] was used for word-level feature extraction. Word-level classifiers were trained with **CRFSuite**[3]. The CRF error models were trained with **CRF++**[4]. POS tagging was performed with **TreeTagger** (Schmid, 1994). Sentence-level QuEst uses 17 baseline features[5] for all tasks. Word-level QuEst reimplements the set of 30 baseline features described in (Luong et al., 2014). The QE models were built and tested based on the data provided for the WMT14 English–Spanish QE shared task (Section 4.3).

The statistics on error distributions were computed using the English–Spanish part of training data for WMT13 shared task on QE[6]. The statistics on the distributions of words, alignments and lexical probabilities were extracted from the Europarl corpus (Koehn, 2005). We trained the alignment model with **FastAlign** (Dyer et al., 2013) and extracted the lexical probabilities tables for words using scripts for phrase table building in **Moses** (Koehn et al., 2007). For all the methods, errors were injected into the News Commentary corpus[7].

[1] `http://scikit-learn.org/`
[2] `http://github.com/ghpaetzold/quest`
[3] `http://www.chokkan.org/software/crfsuite/`
[4] `https://code.google.com/p/crfpp/`
[5] `http://www.quest.dcs.shef.ac.uk/quest_files/features_blackbox_baseline_17`
[6] `http://www.quest.dcs.shef.ac.uk/wmt13_qe.html`
[7] `http://statmt.org/wmt14/training-parallel-nc-v9.tgz`

4.2 Generated data

Combining three methods of errors generation and two methods of errors insertion into sentences resulted in a total of six artificial datasets. Here we perform some analysis on the generated data.

The datasets differ in the percentage of errors injected into the sentences. **BigramEG** datasets have 23% of edits which matches the distribution of errors on the real data. **WordprobEG** datasets contain fewer errors — 17%.

The **crfEG** models contain the lowest number of errors — 5% of the total number of words. As it was expected, data sparsity makes the CRF model tag the majority of the words with the most frequent tag ("Good"). Replacing rare words with a default word token or with a POS tag did not improve these statistics.

Word inserters	Unigram	Paraphrase
Error generators		
Bigram	699.9	888.64
Wordprob	538.84	673.61
CRF + default word	165.36	172.97
CRF + POS tag	161.59	167.23

Table 1: Perplexities of the artificial datasets

We computed the perplexity of all datasets with respect to an LM trained on the Spanish part of the Europarl corpus (see Table 1). The figures match the error percentages in the data — the lower the number of errors, the more is kept from the original sentence, and thus the more natural it looks (lower perplexity). Note that sentences where errors were inserted from a general distribution (**unigramWI**) have lower perplexity than those generated using using paraphrases. This can be because the **unigramWI** model tends to choose high-frequency words with lower perplexity, while the constructed paraphrases contain more noise and rare words.

4.3 Experimental setup

We evaluated the performance of the artificially generated data in three tasks: the ternary classification of sentences as "good", "almost good" or "bad", the prediction of HTER (Snover et al., 2009) score for a sentence, and the classification of words in a sentence as "good" or "bad" (tasks 1.1, 1.2 and 2 of WMT14 QE shared task[8], respectively).

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

The goal of the experiments was to check whether it is possible to improve upon the baseline results by adding artificially generated examples to the training sets. The baseline models for all tasks were trained on the data provided for the corresponding shared tasks for the English–Spanish language pair. All models were tested on the official test sets provided for the corresponding shared tasks.

Since we know how many errors were injected into the sentences, we know the TER scores for our artificial data. The discrete labels for the ternary classification task are defined as follows: "bad" sentences have four or more non-adjacent errors (two adjacent erroneous words are considered one error), "almost good" sentences contain one erroneous phrase (possibly of several words), and "good" sentences are error-free.

The new training examples were added to the baseline datasets. We ran a number of experiments gradually increasing the number of artificially generated sentences used. At every run, the new data was chosen randomly in order to reduce the influence of outliers. In order to make the results more stable, we ran each experiment 10 times and averaged the evaluation scores.

4.4 Sentence-level ternary QE task

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.

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 2 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.

However, adding only negative data makes the distribution of classes in the training data less

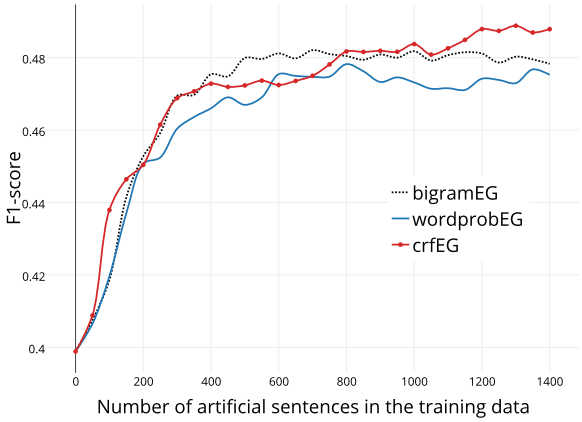

Figure 2: Ternary classification: performance of error generators

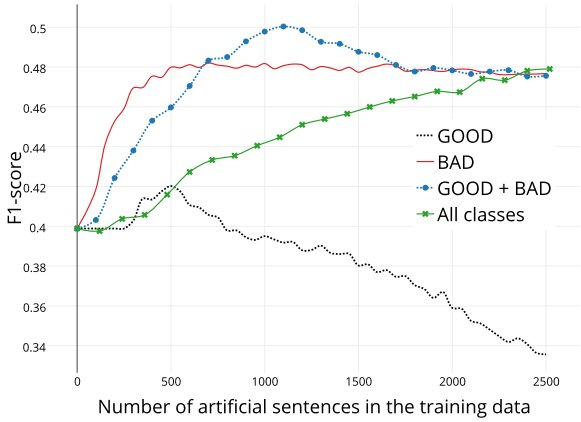

Figure 3: Ternary classification: artificial examples of different classes

similar to that of the test set, which might affect performance negatively. Therefore, we conducted other three sets of experiments: we added (i) equal amount of artificial data for the "good" and "bad" classes (ii) batches of artificial data for all classes that keep the original proportion of classes in the data (iii) artificial data for only the "good" class. The latter setting is tested in order to check whether the classifier benefits from negative instances, or just from having new data added to the training sets.

The results are shown in Figure 3. We plot only the results for the **bigramEG + unigramWI** setting as it achieved the best result in absolute values, but the trends are the same for all data generation techniques. The best strategy was to add both "good" and "bad" sentences: it beats the models which uses only negative examples, but after 1000 artificial sentences its performance degrades. Keeping the original distribution of classes is not beneficial for this task: it performs worse than any other tested scenario since it decreases the F1-score for the "good" class dramatically.

Overall, the additional negative training data improves the ternary sentence classification. The addition of both positive and negative examples can further improve the results, while providing additional instances of the "almost good" class did not seem to be as helpful.

4.5 Sentence-level HTER QE task

Figure 4 shows that the addition of any type of artificial data leads to substantial improvements in quality for this task. The results were evaluated in terms of Mean Absolute Error (MAE). The initial training dataset was very small – 896 sentences (200 sentences for test), which may explain the substantial improvements in prediction quality as new data is added. We also noticed that the performance of the generated datasets was primarily defined by the method of errors generation, whereas different word choice strategies did not impact the results as much. Figure 4 depicts the results for the **unigramWI** words selection method only with all error generation methods.

The addition of data from datasets generated with **crfEG** gives the largest drop in MAE (from 0.161 to 0.14). This result is achieved by a model that uses 1200 artificial sentences. Further addition of new data harms performance. The data generated by other error generators does not cause such a large improvement in quality, although it also helps reduce the error rate.

As it was described earlier, the **crfEG** model generates sentences with a small number of errors. Since the use of this dataset leads to the largest improvements, we can suggest that in the HTER prediction task, using the baseline dataset only, the majority of errors is found in sentences whose HTER score is low. However, the reason might also be that the distributions of scores in the baseline training and test sets are different: the test set has lower average score (0.26 compared to 0.31 in the training set) and lower variance (0.03 versus 0.05 in the training set). The use of artificial data with a small number of errors changes this distribution.

We also experimented with training a model using only artificial data. The results of models trained on only 100 artificial sentences for each

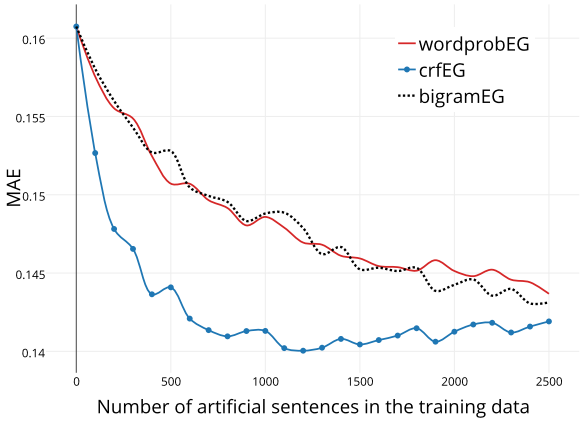

Figure 4: HTER regression results

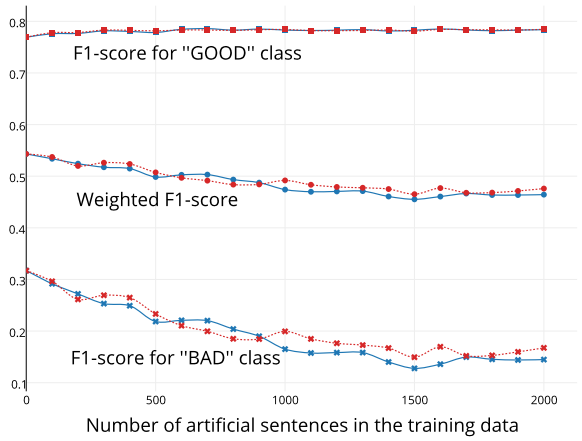

Figure 5: Word-level QE. Blue solid lines – results for **crfEG**, red dotted lines – **bigramEG**

generation method were surprisingly good: their MAE ranged from 0.149 to 0.158 (compared to the baseline result of 0.161 on the original data). However, the further addition of new artificial sentences did not lead to improvements. Thus, despite the positive impact of the artificial data on the results, the models cannot be further improved without real training examples.

4.6 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 5 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.

5 Conclusions and future work

We presented and experimented with a set of new methods of simulation of errors made by MT systems. Sentences with artificially added errors were used as training data in models that predict the quality of sentences or words.

The addition of artificial data can help improve the output of sentence-level QE models, with substantial improvements in HTER score prediction and some improvements in sentences classification into "good", "almost good" and "bad". However, the largest improvements are related to the fact that the additional data changes the overall distribution of scores in the training set, making it more similar to the test set. On the other hand, the fact that the artificial sentences did not decrease the quality in such cases proves that it can be used to counter-balance the large number of positive examples. Unlike sentence-level QE, the task of

word-level QE did not benefit from the artificial data. That may relate to our choice of method to replace words in artificial sentences.

While thus far we analysed the usefulness of artificial data for the QE task only, it would be interesting to check if this data can also improve the performance of discriminative LMs.

Acknowledgements

This work was supported by the EXPERT (EU Marie Curie ITN No. 317471) project.

References

Banerjee, Satanjeev and Alon Lavie. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *ACL-2005, MTSumm workshop*, pages 65–72.

Bhanuprasad, Kamadev and Mats Svenson. 2008. Errgrams A Way to Improving ASR for Highly Inflected Dravidian Languages. In *IJCNLP-2008*, pages 805–810.

Brockett, Chris, William B. Dolan, and Michael Gamon. 2006. Correcting esl errors using phrasal smt techniques. In *Coling-ACL-2006*.

Collins, Michael, Brian Roark, and Murat Saraclar. 2005. Discriminative Syntactic Language Modeling for Speech Recognition. In *ACL-2005*.

Dyer, Chris, Victor Chahuneau, and A. Noah Smith. 2013. A simple, fast, and effective reparameterization of ibm model 2. In *NAACL-HLT-2013*, pages 644–648.

Felice, Mariano and Zheng Yuan. 2014. Generating artificial errors for grammatical error correction. In *EACL-2014*, pages 116–126.

Gamon, Michael, Anthony Aue, and Martine Smets. 2005. Sentence-level MT evaluation without reference translations: beyond language modeling. In *EAMT-2005*.

Izumi, Emi, Kiyotaka Uchimoto, Toyomi Saiga, Thepchai Supnithi, and Hitoshi Isahara. 2003. Automatic error detection in the japanese learners' english spoken data. In *ACL-2003*, pages 145–148.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *ACL-2007, Demo session*, pages 177–180.

Koehn, Philipp. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *MT-Summit 2005*, pages 79–86.

Li, Zhifei and Sanjeev Khudanpur. 2008. Large-scale Discriminative n -gram Language Models for Statistical Machine Translation. In *AMTA-2008*, pages 21–25.

Li, Zhifei, Ziyuan Wang, Sanjeev Khudanpur, and Jason Eisner. 2010. Unsupervised Discriminative Language Model Training for Machine Translation using Simulated Confusion Sets. In *Coling-2010*.

Luong, Ngoc Quang, Laurent Besacier, and Benjamin Lecouteux. 2014. Lig system for word level qe task at wmt14. In *WMT-2014*, pages 335–341.

Okanohara, Daisuke. 2007. A Discriminative Language Model with Pseudo-Negative Samples. In *ACL-2007*, pages 73–80.

Papineni, Kishore, Salim Roukos, Todd Ward, and Weijing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *ACL-2002*, pages 311–318.

Raybaud, Sylvain, David Langlois, and Kamel Smaïli. 2011. This sentence is wrong. Detecting errors in machine-translated sentences. *Machine Translation*, 25(1):1–34.

Rozovskaya, Alla and Dan Roth. 2010. Generating confusion sets for context-sensitive error correction. In *EMNLP-2010*, pages 961–970.

Schmid, Helmut. 1994. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, pages 44–49.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *AMTA-2006*, pages 223–231.

Snover, Matthew, Nitin Madnani, Bonnie Dorr, and Richard Schwartz. 2008. TERp System Description. In *AMTA-2008, MetricsMATR workshop*.

Snover, Matthew, Nitin Madnani, Bonnie J. Dorr, and Richard Schwartz. 2009. Fluency, adequacy, or hter?: Exploring different human judgments with a tunable mt metric. In *WMT-2009*, pages 259–268.

Specia, Lucia, Kashif Shah, Jose G C de Souza, and Trevor Cohn. 2013. QuEst - A translation quality estimation framework. In *ACL-2013, Demo session*.