Regression with Phrase Indicators for Estimating MT Quality[*]

Chunyang Wu **Hai Zhao**[†]
Center for Brain-Like Computing and Machine Intelligence,
Department of Computer Science and Engineering, Shanghai Jiao Tong University
chunyang506@sjtu.edu.cn, zhaohai@cs.sjtu.edu.cn

Abstract

We in this paper describe the regression system for our participation in the quality estimation task of WMT12. This paper focuses on exploiting special phrases, or word sequences, to estimate translation quality. Several feature templates on this topic are put forward subsequently. We train a SVM regression model for predicting the scores and numerical results show the effectiveness of our phrase indicators and method in both ranking and scoring tasks.

1 Introduction

The performance of machine translation (MT) systems has been considerable promoted in the past two decades. However, since the quality of the sentence given by MT decoder is not guaranteed, an important issue is to automatically predict or identify its characteristics. Recent studies on quality estimation or confidence estimation have focused on measuring the translating quality at run-time, instead of involving reference corpus. Researches on this topic contribute to offering advices or warnings for users even without knowledge about either side of languages and illuminating some other potential MT applications.

This paper describes the regression system for our participation in the WMT12 quality estimation task. In this shared task, we analyzed the pattern of translating errors and studied on capturing such patterns among the corpus. The basic objective in this paper is to recognize those phrases, or special word sequence combinations which can indicate the quality of a translation instance. By introducing no external NLP toolkits, we exploited several feasible techniques to extract such patterns directly on the corpus. One contribution of this paper is those feature templates on the basis of this topic. Numerical results show their positive effects on both ranking and scoring subtasks.

The rest of this paper is organized as follows: In Section 2, we show the related work. In Section 3, we specify the details of our system architecture. The experimental results are reported in Section 4. Finally, the conclusion is given in Section 5.

2 Related Work

Compared with traditional MT metrics such as BLEU (Papineni et al., 2002), the fundamental goal of quality estimation (QE) is predicting the quality of output sentences without involving reference sentences.

Early works (Quirk, 2004; Gamon et al., 2005) have demonstrated the consistency of the automatic score and human evaluation. Several further works aimed at predicting automatic scores in order to better select MT n-best candidates (Specia and Farzindar, 2010), measure post-editing effort (Specia et

[*]This work was partially supported by the National Natural Science Foundation of China (Grant No. 60903119 and Grant No. 61170114), the National Research Foundation for the Doctoral Program of Higher Education of China under Grant No. 20110073120022, the National Basic Research Program of China (Grant No. 2009CB320901), the Science and Technology Commission of Shanghai Municipality (Grant No. 09511502400), and the European Union Seventh Framework Program (Grant No. 247619).

[†]corresponding author

al., 2011) or combine SMT and TM systems (He et al., 2010). Instead of estimating on word or sentence levels, Soricut and Echihabi (2010) proposed a document-level ranking system which grants the user to set quality threshold. Besides, recent studies on the QE topic introduced syntactic and linguistic information for better estimating the quality such as dependency preservation checking (Bach et al., 2011).

3 System Description

We specify the details of our system in this section. Following previous approaches for quality estimation, it is first trained on the corpus with labeled quality scores and then it is able to predict the score for unlabeled instances.

A major challenge for this estimating task is to exploit effective indicators, or features, to identify the quality of the translating results. In this paper, all the features are extracted from the official corpora, involving no external tools such as pre-trained parsers or POS taggers. Most of the feature templates focus on special phrases or word sequences. Some of the phrases could introduce translation errors and others might declare the merit of the MT output. Their weights are automatically given by the regressor.

3.1 Regression Model

For obtaining MT quality predictor, we utilize SVM^{light} (Joachims, 1999)[1] to train this regression model. The radial basis function kernel is chosen as the kernel of this model. The label for each instance is the score annotated manually and the input vector consists of a large amount of indicators described in Section 3.2.

3.2 Features

For training the regression model, we utilize the 17 baseline features: number of source/target tokens, average source token length, source/target LM probability, target-side average of target word occurrences, original/inverse frequency average of translations per source word, source/target percentage of uni-/bi-/tri-grams in quartile 1 or 4, source percentage of unigrams in the training corpus and source/target number of punctuation. Besides, several features and templates are proposed as follows:

- **Inverted Automatic Scores:** For each Spanish system output sentence, we translate it to English and get its scores of BLEU and METEOR (Denkowski and Lavie, 2011). These scores are treated as features named *inverted automatic scores*. In order to obtain these numerals, we train a Spanish-to-English phrase-based Moses[2] (Koehn et al., 2007) decoder with default parameters on the official parallel corpus. The original training corpus is split into a developing set containing the last 3000 sentence pairs at the end of the corpus and a training set with the remained pairs. The word alignment information is generated by GIZA++ (Och and Ney, 2003) and the feature weights are tuned on the developing set by Z-MERT (Zaidan, 2009).

- **Minimal/Maximal link likelihood of general language model:** In the word graph of each decoding instance, denote the minimal and maximal general language model likelihood of links as l_{min} and l_{max}. We treat $\exp(l_{min})$ and $\exp(l_{max})$ as features respectively.

- **Trace Density:** Define the trace density ρ_T as the quotient of decoding trace length and sentence length:

$$\rho_T = \text{TraceLength / SentenceLength.} \qquad (1)$$

- **Average of Phrase Length:** This feature is also obtained from the decoding trace information.

- **Number of Name Entity:** This feature cannot be obtained exactly due to the resource constrains. We in this task count the number of the word whose first letter is capitalized, and that is not the first word in the sentence.

We also extract several special phrases or sequences. The total of each phrase/sequence type and each pattern are respectively defined as features. When an instance matches a pattern, the entry representing this pattern in its vector is set to $|1/Z|$. In

[1]http://svmlight.joachims.org/

[2]http://www.statmt.org/moses/

this paper the regressor term Z is the size of the template which the pattern belongs to. The detail description of such templates is presented as follows:

- **Reference 2$\sim$5-grams:** All the 2$\sim$5-grams of the reference sentences provided in the official data are generated as features.

- **Bi-gram Source Splitting:** This template comes from the GIZA alignment document. We scan the parallel corpus: for each bi-gram in the source sentence, if its words' counterparts in the target side are separated, we add it to the *bi-gram source splitting* feature template.

The part-of-speech tags of the words seem to be effective to this task. Since it is not provided, we utilize a trick design for obtaining similar information:

- **Target Functional Word Patterns:** On the target corpus, we scan those words whose length is smaller than or equal to three. Such a word w is denoted as *functional word*. Any bi-gram in the corpus starting or ending with w is added to a dictionary $\mathbb{D}$. For each system-output translation instance, we compare the analogous bi-grams in it with this dictionary, all bi-grams not in $\mathbb{D}$ are extracted as features.

Denote the collection of 2$\sim$5-grams of the system-output sentences scored lower than 1.5 as $\mathbb{B}$; that with scores higher than 4.5 as $\mathbb{G}$. Here the "s-core" is the manual score provided in the official resource.

- **Target Bad 2$\sim$5-grams:** $\mathbb{B} - \mathbb{G}$

- **Target Good 2$\sim$5-grams:** $\mathbb{G} - \mathbb{B}$

- **Source Bad/Good 2$\sim$5-grams:** Analogous phrases on the source side are also extracted by the same methods as *Target Bad/Good n-grams*.

For each output-postedit sentence pair, we construct a bipartite graph by aligning the same words between these two sentences. By giving a maximal matching, the output sentence can be split to several segments by the unmatched words.

- **Output-Postedit Different 2$\sim$5-grams:** For each unaligned segments, we attach the previous word to the left side and the next word to the right. 2$\sim$5-grams in this refined segment are extracted as features.

- **Output-Postedit Different Anchor:** Denote the refined unaligned segment as

  $$s_r = (\text{prevWord}, s_1, s_2, \ldots, s_n, \text{nextWord}).$$

  A special sequence with two word segments

  $$\underline{\text{prevWord } s_1} \ \ldots \ \underline{s_n \text{ nextWord}}$$

  is given as a feature.

In the source-side scenario with the inverted translations, similar feature templates are extracted as well:

- **Source-Invert Different 2$\sim$5-grams/Anchor**

A significant issue to be considered in this shared task is that the training data set is not a huge one, containing about two thousand instances. Although carefully designed, the feature templates however cannot involve enough cases. In order to overcome this drawback, we adopt the following strategy:

For any template $\mathbb{T}$, we compare its patterns with the items in the phrase table. If the phrase item p is similar enough with the pattern g, p is added to the template $\mathbb{T}$. Two similarity metrics are utilized: Denote the longest common sequence as $LCSQ(p, g)$ and the longest common segment as $LCSG(p, g)$ [3],

$$\frac{LCSQ(p,g)^2}{|p||g|} > 0.6, \tag{2}$$

$$LCSG(p, g) \geq 3. \tag{3}$$

Besides, when training the regression model or testing, the entry representing the similar items in the feature vector are also set to $1/|\text{template size}|$.

4 Experiments

4.1 Data

In order to conduct this experiment, we randomly divide the official training data into two parts: one

[3]To simplify, the sequence allows separation while the segment should be contiguous. For example, $LCSQ(p, g)$ and $LCSG(p, g)$ for "*I am happy*" and "*I am very happy*" are "*I, am, happy*" and "*I, am*", respectively.

Data Set	Score Distribution			
	[1-2)	[2-3)	[3-4)	[4-5)
Original	3.2%	24.1%	38.7%	34.0%
Train	3.2%	24.2%	38.3%	34.4%
Dev	3.3%	24.0%	40.3%	32.4%

Table 1: The comparison of the score distributions among three data sets: Original, Training (Train) and Development (Dev).

	Ranking		Scoring	
	DA	SC	MAE	RMSE
Baseline	0.47	0.49	0.61	0.79
This paper	**0.49**	**0.52**	**0.60**	**0.77**

Table 2: The experiment results on the ranking and scoring tasks. In this table, DA, SC, MAE and RMSE are DeltaAvg, Spearman Correlation, Mean-Average-Error and Root-Mean-Squared-Error respectively.

training set with about 3/4 items and one development set with the other 1/4 items. The comparison of the score distribution among these data sets is listed in Table 1.

4.2 Results

The baseline of this experiment is the regression model trained on the 17 baseline features. The parameters of the classifier are firstly tuned on the baseline features. Then the settings for both the baseline and our model remain unchanged. The numerical results for the ranking and scoring tasks are listed in Table 2. The ranking task is evaluated on the DeltaAvg metric (primary) and Spearman correlation (secondary) and the scoring task is evaluated on Mean-Average-Error and Root-Mean-Squared-Error. For the ranking task, our system outperforms 0.02 on DeltaAvg and 0.03 on Spearman correlation; for the scoring task, 0.01 lower on MAE and 0.02 lower on RMSE.

The official evaluation results are listed in Table 3. The official LibSVM[4] model is a bit better than our submission. Our system was further improved after the official submission. Different combinations of the rates defined in Equation 2$\sim$3 and regressor parameter settings are tested. As a result, the "Refined" model in Table 3 is the results of the refined version. Compared with the official model, it gives similar ranking results and performs better on the scoring task.

[4]http://www.csie.ntu.edu.tw/ cjlin/libsvm/

	Ranking		Scoring	
	DA	SC	MAE	RMSE
SJTU	0.53	0.53	0.69	0.83
Official SVM	0.55	0.58	0.69	0.82
Refined	0.55	0.57	**0.68**	**0.81**
Best Workshop	0.63	0.64	0.61	0.75

Table 3: The official evaluation results.

5 Conclusion

We presented the SJTU regression system for the quality estimation task in WMT 2012. It utilized a support vector machine approach with several features or feature templates extracted from the decoding and corpus documents. Numerical results show the effectiveness of those features as indicators for training the regression model. This work could be extended by involving syntax information for extracting more effective indicators based on phrases in the future.

References

Nguyen Bach, Fei Huang, and Yaser Al-Onaizan. 2011. Goodness: a method for measuring machine translation confidence. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 211–219, Stroudsburg, PA, USA. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the EMNLP 2011 Workshop on Statistical Machine Translation.*

Michael Gamon, Anthony Aue, and Martine Smets. 2005. Sentence-level mt evaluation without reference translations: Beyond language modeling. In *In European Association for Machine Translation (EAMT).*

Yifan He, Yanjun Ma, Josef van Genabith, and Andy Way. 2010. Bridging smt and tm with translation recommendation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 622–630, Uppsala, Sweden, July. Association for Computational Linguistics.

Thorsten Joachims. 1999. Advances in kernel methods. chapter Making large-scale support vector machine learning practical, pages 169–184. MIT Press, Cambridge, MA, USA.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the Association for Computational Linguistics Companion Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29(1):19–51, March.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Christopher B. Quirk. 2004. Training a sentence-level machine translation confidence metric. *LREC*, 4:2004.

Radu Soricut and Abdessamad Echihabi. 2010. Trustrank: Inducing trust in automatic translations via ranking. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 612–621, Uppsala, Sweden, July. Association for Computational Linguistics.

Lucia Specia and Atefeh Farzindar. 2010. Estimating machine translation post-editing effort with hter. In *AMTA 2010- workshop, Bringing MT to the User: MT Research and the Translation Industry*. The Ninth Conference of the Association for Machine Translation in the Americas, nov.

Lucia Specia, Hajlaoui N., Hallett C., and Aziz W. 2011. Predicting machine translation adequacy. In *Machine Translation Summit XIII*.

Omar F. Zaidan. 2009. Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems. *The Prague Bulletin of Mathematical Linguistics*, 91:79–88.