Discriminative Latent Variable Based Classifier for Translation Error Detection

Jinhua Du[1], Junbo Guo[2], and Fei Zhao[1]

[1] Faculty of Automation and Information Engineering
jhdu@xaut.edu.cn
[2] Faculty of High Vocational Education,
Xi'an University of Technology, Xi'an, 710048 China

Abstract. This paper presents a discriminative latent variable model (DPLVM) based classifier for improving the translation error detection performance for statistical machine translation (SMT). It uses latent variables to carry additional information which may not be expressed by those original labels and capture more complicated dependencies between translation errors and their corresponding features to improve the classification performance. Specifically, we firstly detail the mathematical representation of the proposed DPLVM method, and then introduce features, namely word posterior probabilities (WPP), linguistic features, syntactic features. Finally, we compare the proposed method with MaxEnt and SVM classifiers to verify its effectiveness. Experimental results show that the proposed DPLVM-based classifier reduce classification error rate (CER) by relative 1.75%, 1.69%, 2.61% compared to the MaxEnt classifier, and relative 0.17%, 0.91%, 2.12% compared to the SVM classifier over three different feature combinations.

Keywords: Translation Error Detection, Binary Classification, MaxEnt Classifier, SVM Classifier, DPLVM Classifier.

1 Introduction

In recent years, a number of different types of SMT methods have been proposed, such as the phrase-based, hierarchical phrased-based, and syntax-based models etc., which significantly improve the translation quality, and a lot of effort has been put to apply SMT systems to practical use. However, the translation quality cannot fully satisfy the actual demand of industry yet. For example, the ungrammatical errors and disordered words in the translation often increase human cost. Therefore, high-quality automatic translation error detection or word-level confidence estimation is necessary to further improve the working efficiency of the post-editors or translators.

Typically, most translation error detection methods utilize system-based features (e.g. WPP) combining with extra knowledge such as linguistic features to decrease the classification error rate (CER) [1–9]. As to the system-based features, a number of different algorithms to calculate the WPP were proposed based on the N-best list or word lattice, and had been applied to SMT translation quality estimation. Afterwards, some researchers try to introduce more useful knowledge sources such as syntactic and semantic features to further improve the error detection capability. However, these features are not that easy to extract due to their complexity, low generalization capability, and dependency on specific languages etc. Hence, currently the system-based features such as WPP and lexicalized features (e.g. word and part-of-speech (POS)) still play the main role in the error detection task or the confidence estimation task.

Generally, translation error detection can be regarded as a binary classification task. Thus, the accuracy of the classifier also plays an important role in terms of improving the prediction capability besides adding new features and extra knowledge. This paper presents a more effective classifier – discriminative probabilistic latent variable model based classifier that uses latent variables to carry additional information which may not be expressed by those original labels and capture more complicated dependencies between errors and their corresponding features to improve the classification performance [10–12].

The rest of the paper is organized as follows: Section 2 briefs the related work. Section 3 describes the DPLVM-based classifier as well as the feature representation. In Section 4, three typical WPP and three linguistic features are described. Experimental settings, implementation and analysis are reported in Section 5. Some observations from the results are also given in this section. The final section concludes and gives avenues for future work.

2 Related Work

The question of translation confidence estimation has attracted a number of researcher due to its importance in promoting SMT application. In 2004, Blatz et al. improved the basic confidence estimation method by combining the neural network and a naive Bayes classifier to predict the word-level and the sentence-level translation errors [2]. The features they used include WPP calculated from the N-best list, translation model-based features, semantic feature extracted from the WordNet, as well as simple syntactic features. Experimental results show that all among these features, WPP is more effective with strong generalization capability than linguistic features.

Ueffing and Ney exhaustively explore various kinds of WPP features to perform confidence measures, and proposed different WPP algorithms to verify the effectiveness in confidence estimation task [1, 3]. In their task, the words in the generated target sentence can be tagged as *correct* or $false$ to facilitate post-editing or work in an interactive translation environment. Their experiments conducted on different data sets show that different WPP algorithms perform differently, but basically each can reduce the CER. Furthermore, the combination of different features can perform better than any individual features.

Specia et al. have done a lot of work with regard to the confidence estimation in the computer-aided translation field [13, 14]. They categorize translations into “bad” or “good” classes based on sentence-level binary scores of the post-edition MT fragments. The features used are called “black-box” features, which can be extracted from any MT systems only if the information from the input (source) and translation (target) sentences are given, such as source and target sentence lengths and their ratios, the edit distance between the source sentence and sentences in the corpus used to train the SMT system. Recently, Specia et al. (2011) have started exploiting linguistic information for sentence-level quality estimation, for instance, used POS tagging, chunking, dependency relations and named entities for English-Arabic quality estimation [6, 7]. Hardmeier explored the use of constituency and dependency trees for English-Swedish/Spanish quality estimation [8].

Xiong et al. proposed an MaxEnt classifier based error detection method to predict translation errors (each word is tagged as *correct* or *incorrect*) by integrating a WPP feature, a syntactic feature extracted from LG parser and some lexical features [4]. The experimental results show that linguistic features can reduce CER when used alone, and it outperforms WPP. Moreover, linguistic features can further provide complementary information when combined with WPP, which collectively reduce the classification error rate.

On the basis of Xiong's work, Du and Wang carried out a systematic comparison between the MaxEnt and SVM classifiers in order to show the influence of different classifiers on the error detection capability. Under the conditions of same data sets and same feature sets, their experiments indicated that the SVM-based classifier performed better than the MaxEnt-based classifier in terms of the CER [9].

On the basis of previous work, this paper mainly focuses on introducing a new classifier to significantly improve the classification performance. Specifically, this paper

– verifies the performance of various classifiers, namely the MaxEnt classifier and the SVM classifier on the translation error detection task;
– presents a new classifier – DPLVM-based classifier – to obtain better results.

3 Discriminative Probabilistic Latent Variable Model Based Classifier

In this section, we come up with a new classifier – DPLVM-based classifier – to perform our translation error detection task.

In natural language processing (NLP) such as sequential labeling [11], DPLVM demonstrated excellent capability of learning latent dependencies of the specific problems, and have outperformed several commonly-used conventional models, such as support vector machines, conditional random fields and hidden Markov models. In this section, we theoretically introduce the definition and mathematical description of the DPLVM algorithm in our task and compare the classification performance with two other classifiers in later sections.

Given a sequence of observations $\mathbf{x} = \{x_1, x_2, \ldots, x_m\}$ and a sequence of labels $\mathbf{y} = \{y_1, y_2, \ldots, y_m\}$, the task is to learn a mapping between $\mathbf{x}$ and $\mathbf{y}$. y_i is a class label and is a member of a set $\mathbf{Y}$ of possible class labels. DPLVM also assumes a sequence of latent variables $\mathbf{h} = \{h_1, h_2, \ldots, h_m\}$, which is hidden in the training examples.

The DPLVM is defined as in (1)(Morency et al., [10]; Sun and Tsujii, [11]):

$$P(\mathbf{y}|\mathbf{x}, \Theta) = \sum_{\mathbf{h}} P(\mathbf{y}|\mathbf{h}, \mathbf{x}, \Theta) P(\mathbf{h}|\mathbf{x}, \Theta) \tag{1}$$

where Θ are the parameters of the model. It can be seen that the DPLVM equates to a CRF model if it has only one latent variable for each label.

For the sake of efficiency, the model is restricted to have disjoint sets of latent variables associated with each class label. Each h_j is a member in a set $\mathbf{H}_{y_j}$ of possible latent variables for the class label y_j. We define $\mathbf{H}$ as the union of all $\mathbf{H}_{y_j}$ sets, so sequences which have any $h_j \notin \mathbf{H}_{y_j}$ will by definition have $P(\mathbf{y}|\mathbf{x}, \Theta) = 0$, so that the model can be rewritten as in (2):

$$P(\mathbf{y}|\mathbf{x}, \Theta) = \sum_{\mathbf{h} \in \mathbf{H}_{y_1} \times \ldots \mathbf{H}_{y_m}} P(\mathbf{h}|\mathbf{x}, \Theta) \tag{2}$$

where $P(\mathbf{h}|\mathbf{x}, \Theta)$ is defined by the usual conditional random field formulation, as in (3):

$$P(\mathbf{h}|\mathbf{x}, \Theta) = \frac{\exp \Theta \cdot \mathbf{f}(\mathbf{h}, \mathbf{x})}{\sum_{\forall \mathbf{h}} \exp \Theta \cdot \mathbf{f}(\mathbf{h}, \mathbf{x})} \tag{3}$$

in which $\mathbf{f}(\mathbf{h}, \mathbf{x})$ is a feature vector. Given a training set consisting of n labeled sequences (x_i, y_i), for $i = 1 \ldots n$, parameter estimation is performed by optimizing the objective function in (4):

$$L(\Theta) = \sum_{i=1}^{n} \log P(y_i|x_i, \Theta) - R(\Theta) \tag{4}$$

The first term of this equation is the conditional log-likelihood of the training data. The second term is a regularizer that is used for reducing over-fitting in parameter estimation.

For decoding in the test stage, given a test sequence $\mathbf{x}$, we want to find the most probable label sequence y^*, as in (5):

$$\mathbf{y}^* = \arg\max_{y} P(\mathbf{y}|\mathbf{x}, \Theta^*) \tag{5}$$

Sun and Tsujii (2009) argued that for latent conditional models like DPLVMs, the best label path $\mathbf{y}^*$ cannot directly be generated by the Viterbi algorithm because of the incorporation of hidden states. They proposed a latent-dynamic inference (LDI) method based on A^* search and dynamic programming to efficiently decode the optimal label sequence $\mathbf{y}^*$. For more details of the LDI algorithm, refer to [11].

Our translation error detection is a binary classification task that annotates a word e of the translation hypothesis e_1^I as "*correct*" if it is translated correctly, or "*incorrect*" if it is a wrong translation. Therefore, the label set for the classification task can be denoted as $\boldsymbol{y} = \{c, i\}$, where $\boldsymbol{y}$ indicates the label set, c stands for class "*correct*" and i represents class "*incorrect*".

4 Features and Vector Representation

4.1 WPP Feature

WPP is served as a major and effective confidence estimation feature both in speech recognition and SMT post-processing. As to SMT, WPP refers to the probability of a word occurring in the hypothesis given a source input. Generally speaking, the underlying idea is that if the posterior probability of a word occurring in a hypothesis is high, then the chance that it is believed to be correct is big correspondingly. Thus, it is reasonable that the more useful information considered in the WPP algorithm, the better the performance would achieve.

The general mathematical description of WPP is as:

For an SMT system S, given the input sentence f_1^J, and the exported N-best list $e_{n,1}^{n,I_n}$, where $n = 1, \ldots, N$, e_n refers to the n^{th} hypothesis with the probability $p(f_1^J, e_{n,1}^{n,I_n})$, then the WPP in the error detection task can be represented as calculating the probability $p_i(e|f_1^J, e_1^I)$ of the word e at position i in the 1-best hypothesis of the N-best list as in (6),

$$p_i(e|f_1^J, e_1^I) = \frac{\sum_{n=1}^{N} f(a, e_{n,i}, e) \cdot p(f_1^J, e_{n,1}^{n,I_n})}{\sum_{n=1}^{N} p(f_1^J, e_{n,1}^{n,I_n})} \tag{6}$$

where a is a hidden variable which indicates an alignment measure; $f(a, e_{n,i}, e)$ is a binary sign function as in (7),

$$f(a, e_{n,i}, e) = \begin{cases} 1 & e_{n,i} = e \\ 0 & otherwise \end{cases} \tag{7}$$

It can be seen from the description of N-best based WPP algorithm that the posterior probability of a word in a hypothesis can be worked out according to the sentence-level posterior probabilities of hypotheses in the N-best list. The vital information to be considered is the position of the word e which is determined by the alignment measure between the 1-best hypothesis and the rest of the N-best list.

Here we introduces three typical WPP methods to illustrate their different influence on the error detection performance over different kinds of classifiers.

4.1.1 Fixed Position Based WPP

The fixed position based WPP is also called "direct WPP". The basic idea is that given an input f_1^J, the posterior probability of a word e at position i in the hypothesis e_1^I can be calculated by summing the posterior probabilities of all sentences in the N-best list containing target word e at target position i, which is as in (8),

$$p_i(e|f_1^J, e_1^I) = \frac{\sum_{n=1}^{N} \delta(e_{n,i}, e) \cdot p(f_1^J, e_{n,1}^{n,I_n})}{\sum_{e'} \sum_{n=1}^{N} \delta(e_{n,i}, e') \cdot p(f_1^J, e_{n,1}^{n,I_n})} \tag{8}$$

where $\delta(x, y)$ is the Kronecker function as in (9),

$$\delta(x, y) = \begin{cases} 1 & x = y \\ 0 & otherwise \end{cases} \tag{9}$$

This method only uses the original position information of each word without any extra alignment measure between the 1-best and any other hypotheses.

4.1.2 Flexible Position Based WPP

The potential problem of fixed position based WPP is that generally the hypotheses in the N-best list have different length that will make the same word occur at different positions so that the WPP would have a large error compared to the real probability distribution. Naturally the intuition to improve this method is to make the position flexible, e.g. using a sliding window.

The basic idea of sliding window is to consider the words around the position i, i.e., the context. Let the window size be t, then the sliding window at position i can be denoted as $i \pm t$. If the target word e appears inside the window, then we regard it occurring at position i and sum up the probability of the current hypothesis, which is formulated as in (10),

$$p_{i,t}(e|f_1^J, e_1^I) = \sum_{k=i-t}^{i+t} p_k(e|f_1^J, e_1^I) \tag{10}$$

where $p_k(e|f_1^J, e_1^I)$ is as illustrated in Eq. (8).

4.1.3 Word Alignment Based WPP

The sliding window based method needs to choose a proper window size which can only be determined by experiments. Thus, another straightforward way to improve the fixed position method is to perform the word alignment between the 1-best hypothesis and the rest of hypotheses in the N-best list, i.e., align the rest of hypotheses against the 1-best hypothesis.

Specifically, let $L(e_1^I, e_{n,1}^{n,I_n})$ be the Levenshtein alignment between e_1^n and other hypotheses, then the WPP of the word e at position i is as in (11):

$$p_{lev}(e|f_1^J, e_1^I) = \frac{p_{lev}(e, f_1^J, e_1^I)}{\sum_{e'} p_{lev}(e', f_1^J, e_1^I)} \tag{11}$$

where

$$p_{lev}(e, f_1^J, e_1^I) = \sum_{n=1}^{N} \delta(e, L_i(e_1^I, e_{n,1}^{n,I_n})) \cdot p(f_1^J, e_{n,1}^{n,I_n}) \tag{12}$$

$p(f_1^J, e_{n,1}^{n,I_n})$ is the posterior probability of each hypothesis in the N-best list, which is given by the SMT system. $\delta(x, y)$ is the Kronecker function as in Eq. (9).

4.2 Linguistic Features

4.2.1 Syntactic Features

Xiong et al. extracted syntactical feature by checking whether a word is connected with other words from the output of the LG parser. When the parser fails to parse the entire sentence, it ignores one word each time until it finds linkages for remaining words. After parsing, those ignored words which are not connected to any other words to be called *null*-linked words. These *null*-linked words are prone to be syntactically incorrect and the linked words are prone to be syntactically correct, then a binary syntactic feature for a word according to its links can be defined as in (13),

$$link(e) = \begin{cases} yes & \texttt{e has links with other words} \\ no & \texttt{otherwise} \end{cases} \tag{13}$$

Refer to detailed description in [4].

4.3 Lexical Features

Lexical features such as the word itself and the POS are common features used in NLP tasks. In this paper, we also utilize the word/pos with its context (e.g. the previous two words/pos and next two words/pos) to form a feature vector as follows,

- *word*: $(w_{-2}, w_{-1}, w, w_1, w_2)$
- *pos*: $(pos_{-2}, pos_{-1}, pos, pos_1, pos_2)$

4.4 Feature Vector Representation

Generally in the NLP classification task, context information is usually to be considered in the process of feature extraction. In our task, we have four kinds of features: *wpp*, *pos*, *word* and *link* (c.f. Section 4). To build a feature vector for a word e, we look at 2 words before and 2 words after the current word position as well. Thus, the feature vector $\mathbf{x}$ that includes four kinds of features can be denoted as,

$$\begin{aligned}\mathbf{x} = <\, & wpp_{-2}, wpp_{-1}, wpp, wpp_1, wpp_2, pos_{-2}, pos_{-1},\\ & pos, pos_1, pos_2, word_{-2}, word_{-1}, word, word_1,\\ & word_2, link_{-2}, link_{-1}, link, link_1, link_2 >\end{aligned}$$

As to the individual classifiers, we use the MaxEnt toolkit[1] as our MaxEnt classifier, use LibSVM[2] as our SVM classifier, and use Sun's open source toolkit [11] as the proposed DPLVM-based classifier respectively. Refer to [9] for more details about Maxent and SVM classifiers.

[1] `http://homepages.inf.ed.ac.uk/s0450736/maxenttoolkit.html.`
[2] Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`

5 Experiments and Analysis

5.1 Chinese-English SMT

We utilize Moses [15] to provide $10,000$-best list with translation direction from Chinese to English. The training data consists of 3,397,538 pairs of sentences (including Hong Kong news, FBIS, ISI Chinese-English Network Data and Xin-Hua news etc.). The language model is five-gram built on the English part of the bilingual corpus and Xinhua part of the English Gigaword.

The development set for SMT training is the current set of NIST MT 2006 (1,664 source sentences) and the test sets are NIST MT-05 (1,082 sentences) and NIST MT-08 (1,357 sentences). Each source sentence has four references. During the decoding process, the SMT system exports $10,000$-best hypotheses for each source sentence, i.e., $N = 10,000$.

Performance of SMT systems on two test sets is shown in Table 1 in terms of BLEU4, TER scoresand ratio of correct words (RCW) scores.

Table 1. SMT performance and the ratio of correct words (RCW)

dataset	BLEU4(%)	WER(%)	TER(%)	RCW(%)
NIST MT 2008	25.97	69.79	63.56	37.99
NIST MT 2005	33.17	69.50	61.40	41.59

5.2 Experimental Settings for Translation Error Detection Task

Development and test sets: In the error detection task, we use NIST MT-08 as the development set to tune the classifiers, and NIST MT-05 as the test set to evaluate the classification performance.
Data annotation: We use the WER metric in TER toolkit [16] to determine the true labels for words in the development and the test sets. Firstly, we perform the minimum edit distance alignment between the hypothesis and the four references, and then select the one with minimum WER score as the final reference to tag the hypothesis. That is, a word e in the hypothesis is tagged as c if it is the same as that in the reference, otherwise tag it as i.

There are 14,658 correct words and 23,929 incorrect words in the 1-best hypothesis of MT-08 set (37.99% ratio of correct words, RCW), 15,179 correct words and 21,318 incorrect words in the 1-best hypothesis of MT-05 set (41.59% RCW). See RCW in Table 1.
Evaluation Metrics: The commonly-used evaluation metrics for the classification task includes CER (classification error rate), precision, recall and F measure. In our translation error detection task, we use CER as the main evaluation metric to evaluate the system performance that is defined as in (14),

$$\texttt{CER} = \frac{\#\texttt{of wrongly tagged words}}{\#\texttt{of total words}} \tag{14}$$

Since the RCW is less than 50% (41.59%), i.e., the number of incorrect words is more than correct words, it is reasonable to use the RCW as the baseline of CER to examine the classification performance of classifiers.

We also use F measure as the auxiliary evaluation metrics to evaluate some performance of features and classifiers. See definitions in [4].

5.3 Error Detection Experiments

5.3.1 Classification Experiments Based on Individual Features

Results of three typical WPP features and three linguistic features on MaxEnt, SVM and the proposed DPLVM classifiers are shown in Table 2.

Table 2. Results of individual features over three classifiers

	MaxEnt		SVM		DPLVM	
Feature	CER(%)	F(%)	CER(%)	F(%)	CER(%)	F(%)
Baseline	*41.59*	–	*41.59*	–	*41.59*	–
WPP_Dir	40.48	67.65	*37.64*	75.11	***37.16***	74.13
WPP_Win	39.70	68.51	*37.47*	75.18	***36.87***	74.90
WPP_Lev	40.12	72.83	*37.37*	75.25	***36.99***	73.84
word	39.11	69.04	*37.68*	71.48	***36.93***	73.11
pos	39.50	71.89	*39.12*	73.68	***37.39***	72.68
link	40.89	72.77	*37.70*	74.78	***37.38***	74.61

WPP_Dir represents the fixed position-based WPP, *WPP_Win* represents the flexible position-based WPP with the window size 2, and *WPP_Lev* represents word alignment-based WPP.

We can see that 1) three WPP features over three classifiers significantly reduce the CER compared to the baseline; 2) the *WPP_Win* and *WPP_Lev* perform better than *WPP_Dir* which shows that position information is helpful; 3) Regarding the linguistic features, they are helpful to significantly reduce the error rate compared to the baseline over three classifiers; 4) the proposed DPLVM performs best compared to the SVM and MaxEnt classifiers in terms of the CER, which verifies that the proposed classifier is effective.

5.3.2 Classification Experiment on Combined Features

The results of the feature combination experiment which combines three typical WPP and three linguistic features over three individual classifiers respectively are shown in Table 3.

In Table 3, *com*1 represents the feature combination of *WPP_Dir + Word + Pos + Link*, *com*2 stands for the feature combination of *WPP_Win + Word + Pos + Link*, and *com*3 indicates the feature combination of *WPP_Lev + Word + Pos + Link*. All these feature combinations are linearly combined without any weights in the model.

We can see from the results that compared to the MaxEnt and SVM classifiers over three feature combinations, namely *com*1, *com*2 and *com*3, the proposed

Table 3. Results of combined features over three classifiers

	MaxEnt		SVM		DPLVM	
Feature	CER(%)	F(%)	CER(%)	F(%)	CER(%)	F(%)
Baseline	*41.59*	–	*41.59*	–	*41.59*	–
com1	35.93	74.17	*35.36*	74.95	***35.30***	74.61
com2	35.55	73.83	*35.27*	74.86	***34.95***	74.69
com3	35.62	73.15	*35.44*	74.75	***34.69***	74.04

DPLVM classifier method achieved significant improvement respectively by relative 1.75%, 1.69%, 2.61%, and 0.17%, 0.91%, 2.12% in terms of CER.

From the systematic comparison of the results, we can conclude:

– generally speaking, *WPP_Win* performs the best and robust both in the three individual WPP features and the three combined features. The reason we consider is that the sliding window makes the alignment more flexible and considers more context information.
– linguistic features are helpful to the error detection.
– SVM classifier outperforms the MaxEnt classifier in all sets of experiments in terms of CER and F measures, and the proposed DPLVM-based classifier performs best in terms of the CER that shows its effectiveness in translation error detection task. It is analyzed that the latent variables can carry additional information and capture more relations between translation errors and features so that the classification performance can be significantly improved.

5.3.3 Observations

We carried out a deep analysis on the results classified, and based on the observations, we found that,

– the name entities (person name, location name, organization name etc.) are prone to be wrongly classified;
– the prepositions, conjunctions, auxiliary verbs and articles are easier to be wrongly classified due to the factors that they often have an impact on the word orders or lead to empty alignment links;
– the proportion of the notional words that are wrongly classified is relatively small.

6 Conclusions and Future Work

This paper presents a new classifier – DPLVM-based classifier – for translation error detection. Firstly a discriminative probabilistic latent variable model based classifier is proposed which takes advantage of hidden information to predict the label for each word in a hypothesis. Then three different kinds of WPP features, three linguistic features are introduced, and finally a systematic comparison among the MaxEnt classifier, SVM classifier and our DPLVM classifier using different individual and combined features is carried out. Experimental results on Chinese-to-English NIST MT data sets show that the proposed classifier performs best compared to two other individual classifiers in terms of CER.

In future work, we intend to carry out further study on the error detection task in the respects of 1) introducing paraphrases to annotate the hypotheses so that it can truly reflect the *correct* or *incorrect* at the semantic level; 2) introducing new useful features to further improve the detection capability; 3) performing experiments on more language pairs to verify our proposed method.

Acknowledgments. This work is supported by NSF project (61100085), SRF for ROCS, State Education Ministry, and Research Foundation of Education Department of Shaanxi Provincial Government (11JK1029). Thanks the reviewers for their insightful comments and suggestions.

References

1. Ueffing, N., Klaus, M., Hermann, N.: Confidence Measures for Statistical Machine Translation. In: Proceedings of the MT Summit IX, pp. 169–176 (2003)
2. Blatz, J., Fitzgerald, E., Foster, G., Gandrabur, S., Goutte, C., Kuesza, A., Sanchis, A., Ueffing, N.: Confidence Estimation for Machine Translation. In: Proceedings of the 20th International Conference on Computational Linguistics, pp. 315–321 (2004)
3. Ueffing, N., Ney, H.: Word-Level Confidence Estimation for Machine Translation. Computational Linguistics 33(1), 9–40 (2007)
4. Xiong, D., Zhang, M., Li, H.: Error detection for statistical machine translation using linguistic features. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 604–611 (2010)
5. Nguyen, B., Huang, F., AI-Onaizan, Y.: Goodness: A Method for Measuring Machine Translation Confidence. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, pp. 211–219 (2011)
6. Specia, L., Hajlaoui, N., Hallett, C., Aziz, W.: Predicting machine translation adequacy. In: MT Summit XIII: Proceedings of the Thirteenth Machine Translation Summit, pp. 513–520 (2011)
7. Mariano, F., Specia, L.: Linguistic features for quality estimation. In: WMT 2012: Proceedings of the 7th Workshop on Statistical Machine Translation, pp. 96–103 (2012)
8. Hardmeier, C., Nivre, J., Tiedemann, J.: Tree kernels for machine translation quality estimation. In: Proceedings of the 7th Workshop on Statistical Machine Translation, pp. 109–113 (2012)
9. Du, J., Wang, S.: A Systematic Comparison of SVM and Maximum Entropy Classifiers for Translation Error Detection. In: Proceedings of the International Conference on Asian Language Processing, IALP (2012)
10. Morency, L.P., Quattoni, A., Darrell, T.: Latent-dynamic Discriminative Models for Continuous Gesture Recognition. In: Proceedings of the CVPR 2007, pp. 1–8 (2007)
11. Sun, X., Tsujii, J.: Sequential Labeling with Latent Variables: An Exact Inference Algorithm and An Ecient Approximation. In: Proceedings of the European Chapter of the Association for Computational Linguistics (EACL 2009), pp. 772–780 (2009)

12. Du, J., Way, A.: A discriminative latent variable-based classifier for Chinese-English SMT. In: Proceedings of the 23rd International Conference on Computational Linguistics, pp. 286–294 (2010)
13. Specia, L., Cancedda, N., Dymetman, M., Turchi, M., Cristianini, N.: Estimating the sentence-level quality of machine translation systems. In: Proceedings of the 13th Annual Conference of the European Association for Machine Translation, pp. 28–35 (2009)
14. Specia, L., Saunders, C., Turchi, M., Wang, Z., Shawe-Taylor, J.: Improving the confidence of machine translation quality estimates. In: Proceedings of the Twelfth Machine Translation Summit, pp. 136–143 (2009)
15. Koehn, P., Hoang, H., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., Herbst, E.: Moses: Open Source Toolkit for Statistical Machine Translation. In: Proceedings of the Demo and Poster Sessions, ACL 2007, pp. 177–180 (2007)
16. Snover, M., Dorr, B., Schwartz, R., Micciulla, L., Makhoul, J.: A study of translation edit rate with targeted human annotation. In: Proceedings of the 7th Conference of the Association for Machine Translation in the Americas, pp. 223–231 (2006)