Machine Translation Detection from Monolingual Web-Text

Yuki Arase
Microsoft Research Asia
No. 5 Danling St., Haidian Dist.
Beijing, P.R. China
yukiar@microsoft.com

Ming Zhou
Microsoft Research Asia
No. 5 Danling St., Haidian Dist.
Beijing, P.R. China
mingzhou@microsoft.com

Abstract

We propose a method for automatically detecting low-quality Web-text translated by statistical machine translation (SMT) systems. We focus on the *phrase salad* phenomenon that is observed in existing SMT results and propose a set of computationally inexpensive features to effectively detect such machine-translated sentences from a large-scale Web-mined text. Unlike previous approaches that require bilingual data, our method uses only *monolingual* text as input; therefore it is applicable for refining data produced by a variety of Web-mining activities. Evaluation results show that the proposed method achieves an accuracy of 95.8% for sentences and 80.6% for text in noisy Web pages.

1 Introduction

The Web provides an extremely large volume of textual content on diverse topics and areas. Such data is beneficial for constructing a large scale monolingual (Microsoft Web N-gram Services, 2010; Google N-gram Corpus, 2006) and bilingual (Nie et al., 1999; Shi et al., 2006; Ishisaka et al., 2009; Jiang et al., 2009) corpus that can be used for training statistical models for NLP tools, as well as for building a large-scale knowledge-base (Suchanek et al., 2007; Zhu et al., 2009; Fader et al., 2011; Nakashole et al., 2012). With recent advances in statistical machine translation (SMT) systems and their wide adoption in Web services through APIs (Microsoft Translator, 2009; Google Translate, 2006), a large amount of text in Web pages is translated by SMT systems. According to Rarrick et al. (2011), their Web crawler finds that more than 15% of English-Japanese parallel documents are machine translation. Machine-translated sentences are useful if they are of sufficient quality and indistinguishable from human-generated sentences; however, the quality of these machine-translated sentences is generally much lower than sentences generated by native speakers and professional translators. Therefore, a method to detect and filter such SMT results is desired to best make use of Web-mined data.

To solve this problem, we propose a method for automatically detecting Web-text translated by SMT systems[1]. We especially target machine-translated text produced through the Web APIs that is rapidly increasing. We focus on the *phrase salad* phenomenon (Lopez, 2008), which characterizes translations by existing SMT systems, *i.e.*, each phrase in a sentence is semantically and syntactically correct but becomes incorrect when combined with other phrases in the sentence. Based on this trait, we propose features for evaluating the likelihood of machine-translated sentences and use a classifier to determine whether the sentence is generated by the SMT systems.

The primary contributions of the proposed method are threefold. First, unlike previous studies that use parallel text and bilingual features, such as (Rarrick et al., 2011), our method only requires monolingual text as input. Therefore, our method can be used in monolingual Web data mining where bilingual information is unavailable. Second, the proposed features are designed to be computationally light so that the method is suitable for handling a large-scale Web-mined data. Our method determines if an input sentence contains phrase salads using a simple yet effective features, *i.e.*, language models (LMs) and automatically obtained non-contiguous phrases that are frequently used by people but difficult for SMT systems to generate. Third, our method computes features using both human-generated text and SMT

[1] In this paper, the term *machine-translated* is used for indicating translation by SMT systems.

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 1597–1607,
Sofia, Bulgaria, August 4-9 2013. ©2013 Association for Computational Linguistics

results to capture a phrase salad by contrasting these features, which significantly improves detection accuracy.

We evaluate our method using Japanese and English datasets, including a human evaluation to assess its performance. The results show that our method achieves an accuracy of 95.8% for sentences and 80.6% for noisy Web-text.

2 Related Work

Previous methods for detecting machine-translated text are mostly designed for bilingual corpus construction. Antonova and Misyurev (2011) design a phrase-based decoder for detecting machine-translated documents in Russian-English Web data. By evaluating the BLEU score (Papineni et al., 2002) of translated documents (by their decoder) against the target-side documents, machine translation (MT) results are detected. Rarrick et al. (2011) extract a variety of features, such as the number of tokens and character types, from sentences in both the source and target languages to capture words that are mis-translated by MT systems. With these features, the likelihood of a bilingual sentence pair being machine-translated can be determined.

Confidence estimation of MT results is also a related area. These studies aim to precisely replicate human judgment in terms of the quality of machine-translated sentences based on features extracted using a syntactic parser (Corston-Oliver et al., 2001; Gamon et al., 2005; Avramidis et al., 2011) or essay scoring system (Parton et al., 2011), assuming that their input is always machine-translated. In contrast, our method aims at making a binary judgment to distinguish machine-translated sentences from a mixture of machine-translated and human-generated sentences. In addition, although methods for confidence estimation can assume sentences of a known source language and reference translations as inputs, these are unavailable in our problem setting.

Another related area is automatic grammatical error detection for English as a second language (ESL) learners (Leacock et al., 2010). We use common features that are also used in this area. They target specific error types commonly made by ESL learners, such as errors in prepositions and subject-verb agreement. In contrast, our method does not specify error types and aims to detect machine-translated sentences focusing on the phrase salad phenomenon produced by SMT systems. In addition, errors generated by ESL learners and SMT systems are different. ESL learners make spelling and grammar mistakes at the word level but their sentence are generally structured while SMT results are unstructured due to phrase salads. Works on *translationese* detection (Baroni and Bernardini, 2005; Kurokawa et al., 2009; Ilisei et al., 2010) aim to automatically identify *human-translated* text by professionals using text generated by native speakers. These are related, but our work focuses on machine-translated text.

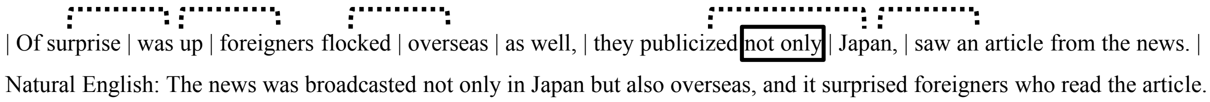
The closest to our approach is the method proposed by Moore and Lewis (2010). It automatically selects data for creating a domain-specific LM. Specifically, the method constructs LMs using corpora of target and non-target domains and computes a cross-entropy score of an input sentence for estimating the likelihood that the input sentence belongs to the target or non-target domains. While the context is different, our work uses a similar idea of data selection for the purpose of detecting low-quality sentences translated by SMT systems.

3 Proposed Method

When APIs of SMT services are used for machine-translating an Web page, they typically insert specific tags into the HTML source. Utilizing such tags makes MT detection trivial. However, the actual situation is more complicated in real Web data. When people manually copy and paste machine-translated sentences, such tags are lost. In addition, human-generated and machine-translated sentences are often mixed together even in a single paragraph. To observe the distribution of machine-translated sentences in such difficult cases, we examine $3K$ sentences collected by our in-house Web crawler. Among them, excluding the pages with the tags of MT APIs, 6.7% of them are found to be clearly machine translation. Our goal is to automatically identify these sentences that cannot be simply detected by the tags, except when the sentences are of sufficient quality to be indistinguishable from human-generated sentences.

3.1 Phrase Salad Phenomenon

Fig. 1 illustrates the phrase salad phenomenon that characterizes a sentence translated by an existing



| Of surprise | was up | foreigners flocked | overseas | as well, | they publicized not only | Japan, | saw an article from the news. |

Natural English: The news was broadcasted not only in Japan but also overseas, and it surprised foreigners who read the article.

Figure 1: The phrase salad phenomenon in a sentence translated by an SMT system; each (segmented) phrase is correct and fluent, but dotted arcs show unnatural sequences of phrases and the boxed phrase shows an incomplete non-contiguous phrase.

SMT system. Each phrase, a sequence of consecutive words, is fluent and grammatically correct; however, the fluency and grammar correctness are both poor in inter-phrases. In addition, a phrase salad becomes obvious by observing distant phrases. For example, the boxed phrase in Fig. 1 is a part of the non-contiguous phrase "not only $\star$ but also[2];" however, it lacks the latter part of the phrase ("but also") that is also necessary for composing a meaning. Such non-contiguous phrases are difficult for most SMT systems to generate, since these phrases require insertion of sub-phrases in distant parts of the sentence.

Based on the observation of these characteristics, we define features to capture a phrase salad by examining local and distant phrases. These features evaluate (1) fluency (Sec. 3.2), (2) grammaticality (Sec. 3.3), and (3) completeness of non-contiguous phrases in a sentence (Sec. 3.4). Furthermore, humans can distinguish machine-translated text because they have prior knowledge of how a human-generated sentence would look like, which has been accumulated by observing a lot of examples through their life. This knowledge makes phrase-salads, e.g., missing objects and influent sequence of words, obvious for humans since they rarely appear on human-generated sentences. Based on this assumption, we extract these features using both human-generated and machine-translated text. Features extracted from human-generated text represent the similarity to human-generated text. Likewise, features extracted from machine-translated text depict the similarity to machine-translated text. By contrasting these feature weights, we can effectively capture phrase salads in the sentence.

3.2 Fluency Feature

In a machine-translated sentence, fluency becomes poor among phrases where a phrase salad occurs. We capture this influency using two independent LM scores; $f_{w,H}$ and $f_{w,MT}$. The former LM is trained with human-generated sentences and the latter one is trained with machine-translated sentences. We input a sentence into both of the LMs and use the scores as the fluency features.

3.3 Grammaticality Feature

In a sentence with phrase salads, its grammaticality is poor because tense and voice become inconsistent among phrases. We capture this using LMs trained with part-of-speech (POS) sequences of human-generated and machine-translated sentences, and the features of $f_{pos,H}$ and $f_{pos,MT}$ are respectively computed. In a similar manner with a word-based LM, such grammatical inconsistency among phrases is detectable when computing a POS LM score, since the score becomes worse when an N-gram covers inter-phrases where a phrase salad occurs. This approach achieves computational efficiency since it only requires a POS tagger.

Since a phrase salad may occur among distant phrases of a sentence, it is also effective to evaluate combinations of phrases that cannot be covered by the span of N-gram. For this purpose, we make use of function words that sparsely appear in a sentence where their combinations are syntactically constrained. For example, the same preposition rarely appears many times in a human-generated sentence, while it does in a machine-translated sentence due to the phrase salad. Similar to the POS LM, we first analyze sentences generated by human or SMT by a POS tagger, extract sequences of function words, and finally train LMs with the sequences. We use these LMs to obtain scores that are used as features $f_{fw,H}$ and $f_{fw,MT}$.

3.4 Gappy-Phrase Feature

There are a lot of common non-contiguous phrases that consist of sub-phrases (contiguous word string) and gaps, which we refer to as *gappy-phrases* (Bansal et al., 2011). We specifically use gappy-phrases of 2-tuple, *i.e.*, phrases consisting of two sub-phrases and one gap in the middle. Let us take an English example "not only $\star$ but

[2] We use the symbol $\star$ to represent a gap in which any word or phrase can be placed.

Sequences
World population not only grows , but grows old .
A press release not only informs but also teases .
Hazelnuts are not only for food , but also fuel .
The coalition must not only listen but also act .

Table 1: Example of a sequence database

also." When a sentence contains the phrase "not only," the phrase "but also" is likely to appear in human-generated setences. Such a gappy-phrase is difficult for SMT systems to correctly generate and causes a phrase salad. Therefore, we define a feature to evaluate how likely a sentence contains gappy-phrases in a complete form without missing sub-phrases. This feature is effective to complement LMs that capture characteristics in N-grams.

Sequential Pattern Mining It is costly to manually collect a lot of such gappy-phrases. Therefore, we regard the task as sequential pattern mining and apply PrefixSpan proposed by Pei et al. (2001), which is a widely used sequential pattern mining method[3].

Given a set of sequences and a user-specified *min_support* $\in \mathbb{N}$ threshold, the sequential pattern mining finds all frequent subsequences whose occurrence frequency is no less than *min_support*. For example, given a sequence database like Table 1, the sequential pattern mining finds all frequent subsequences, *e.g.*, "not only," "not only $\star$ but also," "not $\star$ but $\star$," and *etc*.

To capture a phrase salad by contrasting appearance of gappy-phrases in human-generated and machine-translated text, we independently extract gappy-phrases from each of them using PrefixSpan. We then compute features $f_{g,H}$ and $f_{g,MT}$ using the obtained phrases.

Observation of Extracted Gappy-Phrases Based on a preliminary experiment, we set the parameter *min_support* of PrefixSpan to 100 for computational efficiency. We extract gappy-phrases (of 2-tuple) from our development dataset described in Sec. 4.1 that includes $254K$ human-generated and $134K$ machine-translated sentences in Japanese, and $210K$ human-generated and $159K$ machine-translated sentences in English.

Regarding the Japanese dataset, we obtain about $104K$ and $64K$ gappy-phrases from human-generated and machine-translated sentences, respectively. According to our observation of the extracted phrases, $21K$ phrases commonly appear in human-generated and machine-translated sentences. Many of these common phrases are incomplete forms of gappy-phrases that lack semantic meaning to humans, such as "not only $\star$ the" and "not only $\star$ and." On the other hand, complete forms of gappy-phrases that preserve semantic meaning exclusively appear in phrases extracted from human-generated sentences. We also obtain about $74K$ and $42K$ phrases from human-generated and machine-translated sentences in the English dataset ($21K$ of them are common).

Phrase Selection As a result of sequential pattern mining, we can gather a huge number of gappy-phrases from human-generated and machine-translated text, but as we described above, many of them are common. In addition, it is computationally expensive to use all of them. Therefore, our method selects useful phrases for detecting machine-translated sentences.

Although there are several approaches for feature selection, *e.g.*, (Sebastiani, 2002), we use a method that is suitable for handling a large number of feature candidates. Specifically, we evaluate gappy-phrases based on the information gain that measures the amount of information in bits obtained for class prediction when knowing the presence or absence of a phrase and the corresponding class distribution. This corresponds to measuring an expected reduction in entropy, *i.e.*, uncertainty associated with a random factor. The information gain $G \in \mathbb{R}$ for a gappy-phrase g is defined as

$$\begin{aligned} G(g) \quad &\doteq \quad H(C) - P(X_g^1)H(C|X_g^1) \\ &\quad -P(X_g^0)H(C|X_g^0), \end{aligned}$$

where $H(C)$ represents the entropy of the classification, C is a stochastic variable taking a class, X_g is a stochastic variable representing the presence (X_g^1) or absence (X_g^0) of the phrase g, $P(X_g)$ represents the probability of presence or absence of the phrase g, and $H(C|X_g)$ is the conditional entropy due to the phrase g. We use top-k phrases based on the information gain G. Specifically, we use the top 40% of phrases to compute the feature values. Table 2 shows examples of gappy-phrases extracted from human-generated and machine-translated text in our development dataset and remain after feature selection.

[3] Due to the severe space limitation, readers are referred to that paper.

Human	in the early ⋆ period	MT	after ⋆ after the
	known as ⋆ to		and also ⋆ and
	more ⋆ than		and ⋆ but the
	not only ⋆ but also		no ⋆ not
	with ⋆ as well as		not ⋆ not

Table 2: Example of gappy-phrases extracted from human-generated and machine-translated text; phrases preserving semantic meaning are extracted only from human-generated text.

The gappy-phrases depend on each other, and the more phrases extracted from human-generated (machine-translated) text are found in a sentence, the more likely the sentence is human-generated (machine-translated). Therefore, we compute the feature as

$$f_c(s) = \sum_{i \in k} w_i \delta(i, s),$$

where w_i is a weight of the i-th phrase, and $\delta(i, s)$ is a Kronecker's delta function that takes 1 if the sentence s includes the i-th phrase and takes 0 otherwise. We may set the weight w_i according to the importance of the phrase, such as the information gain. In this work, we set w_i to 1 for simplicity.

3.5 Classification

Table 3 summarizes the features employed in our method. In addition to the discussed features, we use the length of a sentence as a feature f_{len} to avoid the bias of LM-based features that favor shorter sentences. The proposed method takes a monolingual sentence from Web data as input and computes a feature vector of $f = (f_{w,H}, \ldots, f_{len}) \in \mathbb{R}^9$. Each feature is finally normalized to have a zero-mean and unit variance distribution. In the feature space, a support vector machine (SVM) classifier (Vapnik, 1995) is used to determine the likelihoods of machine-translated and human-generated sentences.

4 Experiments

We evaluate our method using both Japanese and English datasets from various aspects and investigate its characteristics. In this section, we describe our experiment settings.

4.1 Data Preparation

For the purpose of evaluation, we use human-generated and machine-translated sentences for constructing LMs, extracting gappy-phrases, and training a classifier. These sentences should be ensured to be human-generated or machine-translated, and the human-generated and machine-translated sentences express the same content for fairness of evaluation to avoid effects due to vocabulary difference.

Feature	Notation
Fluency	$f_{w,H}$, $f_{w,MT}$
Grammaticality	$f_{pos,H}$, $f_{pos,MT}$
	$f_{fw,H}$, $f_{fw,MT}$
Gappy-phrase	$f_{g,H}$, $f_{g,MT}$
Length	f_{len}

Table 3: List of proposed features and their notations

As a dataset that meets these requirements, we use parallel text in public websites (this is for fair evaluation and our method can be trained using *nonparallel* text on an actual deployment). Eight popular sites having Japanese and English parallel pages are crawled, whose text is manually verified to be human-generated. The main textual content of these $131K$ parallel pages are extracted, and the sentences are aligned using (Ma, 2006). As illustrated in Fig. 2, the text in one language is fed to the Bing translator, Google Translate, and an in-house SMT system[4] implemented based on (Chiang, 2005) by ourselves for obtaining sentences translated by SMT systems. Due to a severe limitation on the number of requests to the APIs, we randomly subsample sentences before sending them to these SMT systems. We use text in the other language as human-generated sentences[5].

In this manner, we prepare $508K$ human-generated and $268K$ machine-translated sentences as a Japanese dataset, and $420K$ human-generated and $318K$ machine-translated sentences as an English dataset. We split each of them into two even datasets and use one for development and the other for evaluation.

4.2 Experiment Setting

For the fluency and grammaticality features, we train 4-gram LMs using the development dataset with the SRI toolkit (Stolcke, 2002). To obtain the POS information, we use Mecab (Kudo et al., 2004) for Japanese and a POS tagger developed by Toutanova et al. (2003) for English. We evaluate

[4] A preliminary evaluation of the in-house SMT system shows that it has comparable quality with Bing translator.

[5] These are a mixture of sentences generated by native speakers and professional translators/editors.

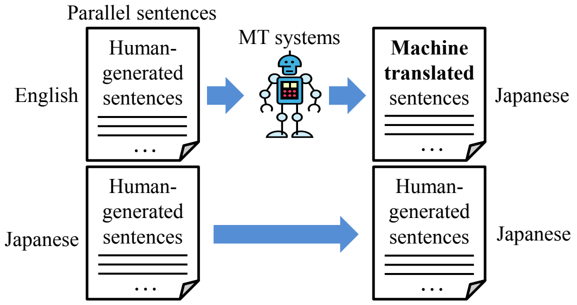

Parallel sentences
English
Human-generated sentences
...
MT systems
Machine translated sentences
...
Japanese
Japanese
Human-generated sentences
...
Human-generated sentences
...
Japanese

Figure 2: Experimental data preparation; text in one language is fed to SMT systems and the other is used as human-generated sentences.

Method		Accuracy
Cross-Entropy		90.7
Lexical Feature		87.8
Proposed feature	**Word LMs**	**94.1**
	POS LMs	91.3
	FW LMs	82.7
	GPs	85.7

Table 4: Accuracy (%) of individual features and comparison methods

the effect of the sizes of N-grams and development dataset in the experiments.

Using the proposed features, we train an SVM classifier for detecting machine-translated sentences. We use an implementation of LIBSVM (Chang and Lin, 2011) with a radial basis function kernel due to the relatively small number of features in the proposed method. We set appropriate parameters by grid search in a preliminary experiment.

We evaluate the performance of MT detection based on accuracy[6] that is a broadly used evaluation metric for classification problems:

$$\text{accuracy} = \frac{n_{TP} + n_{TN}}{n},$$

where n_{TP} and n_{TN} are the numbers of true-positives and true-negatives, respectively, and n is the total number of exemplars. The accuracy scores that we report in Sec. 5 are all based on 10-fold cross validation.

4.3 Comparison Method

We compare our method with the method of (Moore and Lewis, 2010) (*Cross-Entropy*). Although the Cross-Entropy method is designed for the task of domain adaptation of an LM, our problem is a variant of their original problem and thus their method is directly relevant. In our context, the method computes the cross-entropy scores $I_{MT}(s)$ and $I_H(s)$ of an input sentence s against LMs trained on machine-translated and human-generated sentences. Cross-entropy and perplexity are monotonically related, as perplexity of s according to an LM M is simply obtained by $b^{I_M(s)}$ where $I_M(s)$ is cross-entropy score and b is a base with regard to which the cross-entropy is measured. The method scores the sentence according to the cross-entropy difference, *i.e.*, $I_{MT}(s) - I_H(s)$, and decides that the sentence is machine-translated when the score is lower than a predefined threshold. The classification is performed by 10-fold cross validation. We find the best performing threshold on a training set and evaluate the accuracy with a test set using the determined threshold.

Additionally, we compare our method to a method that uses a feature indicating presence or absence of unigrams, which we call *Lexical Feature*. This feature is commonly used for translationese detection and shows the best performance as a single feature in (Baroni and Bernardini, 2005). It is also used by Rarrick et al. (2011) and shows the best performance by itself in detecting machine-translated sentences in English-Japanese translation in the setting of bilingual input. We implement the feature and use it against a monolingual input to fit our problem setting.

5 Results and Discussions

In this section, we analyze and discuss the experiment results in detail.

5.1 Accuracy on Japanese Dataset

We evaluate the sentence-level and document-level accuracy of our method using the Japanese dataset. Specifically, we evaluate effects of individual features and their combinations, compare with human annotations, and assess performance variations across different sentence lengths and various settings on LM training.

Effect of Individual Feature Table 4 shows the accuracy scores of individual features and comparison methods. We refer to features for fluency ($f_{w,H}$, $f_{w,MT}$) as *Word LMs*, grammaticality using POS LMs ($f_{pos,H}$, $f_{pos,MT}$) as *POS LMs*

[6] Although we also examine precision and recall of classification results, they are similar to accuracy reported in this paper.

Method	Accuracy
Word LMs + GPs	94.7
Word LMs + POS LMs	95.1
Word LMs + POS LMs + GPs	95.4
Word LMs + POS LMs + FW LMs	95.5
All	**95.8**

Table 5: Accuracy (%) of feature combinations; there are significant differences ($p \ll .01$) against the accuracy score of Word LMs.

Error	Ratio (%)	Accuracy	
		Word LMs	All
Has wrong content words	37.8	93.1	95.0
Misses content words	12.2	91.8	96.5
Has wrong function words	19.7	92.7	97.1
Misses function words	13.0	93.3	95.6
Has wrong inflections	10.8	97.3	98.7

Table 6: Distribution (%) of machine translation errors and accuracy (%) of proposed method on the different errors

and function word LMs ($f_{fw,H}$, $f_{fw,MT}$) as *FW LMs*, respectively, and for completeness of gappy-phrases ($f_{g,H}$, $f_{g,MT}$) as *GPs*. The Word LMs show the best accuracy that outperforms Cross-Entropy by 3.4% and Lexical Feature by 6.3%. This high accuracy is achieved by contrasting fluency in human-generated and machine-translated text to capture the phrase salad phenomenon. The accuracy of Word LM trained only on human-generated sentences is limited to 65.5%. On the other hand, the accuracy of Word LM trained on machine-translated sentences shows a better performance (84.4%). By combining these into a single feature vector $f = (f_{w,H}, f_{w,MT}, f_{len})$, the accuracy is largely improved.

It is interesting that Lexical Feature achieves a high accuracy of 87.8% despite its simplicity. Since Lexical Feature is a bag-of-words model, it can consider distant words in a sentence. This is effective for capturing a phrase salad that occurs among distant phrases, which N-gram cannot cover. As for Cross-Entropy, a simple subtraction of cross-entropy scores cannot well contrast the fluency in human-generated and machine-translated text and results in poorer accuracy than Word LMs.

The accuracy of POS LMs (91.3%) is slightly lower than that of Word LMs due to the limited vocabulary, *i.e.*, the number of POSs. The accuracy of FW LMs and GPs are even lower. This is convincing since these features cannot have reasonable values when a sentence does not include a function word and gappy-phrase. However, these features are complementary to Word LMs as we will see in the next paragraph.

Effect of Feature Combination Table 5 shows the accuracy when combining features. Sign tests show that the accuracy scores of these feature combinations are significantly different ($p \ll .01$) against the accuracy of Word LMs. The results show that the features complement each other. The combination of all features reaches an accuracy of 95.8%, which improves the accuracy of Word LMs by 1.7%. This result supports that FW LMs and GPs are effective to capture a phrase salad occurring in distant phrases and complement the evidence in N-grams that is captured by LMs. This effect becomes more obvious in the human evaluation.

We also evaluate the accuracy of the proposed method at a document level. Due to the high accuracy at the sentence-level, we use a voting method to judge a document, *i.e.*, deciding if the document is machine-translated when γ% of its sentences are judged as machine-translated. We use all features and find that our method achieves 99% precision and recall with $\gamma = 50$.

Human Evaluation To further investigate the characteristics of our method, we conduct a human evaluation. We sample Japanese sentences and ask three native speakers to 1) judge whether a sentence is human-generated or machine-translated and 2) list errors that the sentence contains. Regarding the task 1), we allow the annotators to assign "hard to determine" for difficult cases. We allocate about 230 sentences for each annotator (in total 700 sentences) without overlapping annotation sets.

The accuracy of annotations is found to be 88.2%, which shows that our method is even superior to native speakers. Agreement between the annotators and our method (with all features) is 85.1%. As we interview the annotators, we find that human annotations are strongly affected by the annotators' domain knowledge. For example, technical sentences are more often misclassified by the annotators.

Table 6 shows the distribution of errors on machine-translated sentences found by the annotators (on sentences that they correctly classified) with the accuracy of Word LMs and all features on

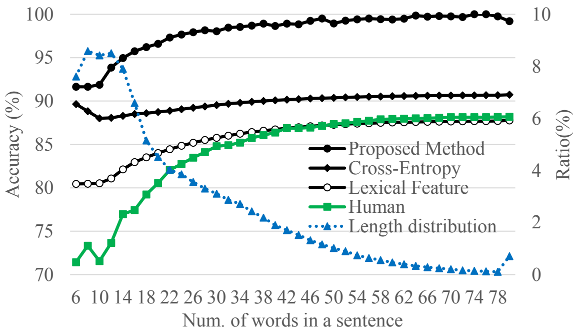


Figure 3: Accuracy (%) across different sentence lengths (the primary axis) and distribution (%) of sentence lengths in the evaluation dataset (the secondly axis)

these sentences (a sentence may contain multiple errors). It indicates that the accuracy of Word LMs is improved by feature combination; from 1.4% on sentences of "Has wrong inflections" to 4.7% on sentences of "Misses content words".

Effect of Sentence Length The accuracy of the proposed method is significantly affected by sentence length (the number of words in a sentence). Fig. 3 shows the accuracy of the proposed method (with all features) and comparison methods w.r.t. sentence lengths (with the primary axis), as well as the distribution of sentence lengths in the evaluation dataset (with the secondly axis). We aggregate the classification results on each cross-validation (test results). It also shows the accuracy of human annotations w.r.t. sentence lengths, which we obtain for the 700 sentences in the human evaluation. The accuracy drops on all methods when sentences are short; the accuracy of our method is 91.6% when a sentence contains less than or equal to 10 words. The proposed method shows the similar trend with the human annotations, and even the accuracy of human annotations significantly drops on such short sentences. This result indicates that SMT results on short sentences tend to be of sufficient quality and indistinguishable from human-generated sentences. Since such high-quality machine-translations do not harm the quality of Web-mined data, we do not need to detect them.

Effect of Setting on LM Training We evaluate the performance variation w.r.t. the sizes of N-grams and development dataset. Fig. 4 shows the accuracy of the LM based features and feature combination when changing sizes of N-grams. The performance of Word LMs is stabilized after 3-gram while that of POS LMs is still improved at 4-gram. This is because POS LMs need more evidence to compensate for their limited vocabulary. FW LMs become stable at 3-gram because the possible number of function words in a sentence should be small.

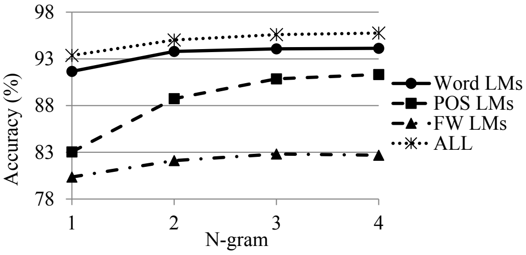


Figure 4: Effect of the sizes of N-grams on MT detection accuracy (%)

When we change the size of the development dataset with 10% increments, the accuracy curve is stabilized when the size is 40% of all set. Considering the fact that the overall development dataset is small, it shows that our method is deployable with a small dataset.

5.2 Accuracy on English Dataset

To investigate the applicability of our method to other languages, we apply the same method to the English dataset. Because English is a configurational language, function words are less flexible than case markers in Japanese. Therefore, SMT systems may better handle English function words, which potentially decreases the effect of FW LMs in our method. In addition, because English is a morphologically poor language, the effect of POS LMs may be reduced.

Nevertheless, in our experiment, all features are shown to be effective even with the English dataset. The combination of all features achieves the best performance, with an accuracy of 93.1%, which outperforms Cross-Entropy by 1.9%, and Lexical Feature by 8.5%. Even though improvements by POS LMs and FW LMs are smaller than Japanese case, their effects are still positive. We also find that GPs stably contribute to the accuracy. These results show the applicability of our method to other languages.

5.3 Accuracy on Raw Web Pages

To avoid unmodeled factors affecting the evaluation, we have carefully removed noise from our experiment datasets. However, real Web pages are more complex; there are often instances of sentence fragments, such as captions and navigational link text. To evaluate the accuracy of our method on real Web pages, we conduct experiments using the dataset generated by Rarrick et al. (2011) that contains randomly crawled Web pages annotated by two annotators to judge if a page is human-generated or machine-translated. We use Japanese sentences extracted from 69 pages (43 human-generated and 26 machine-translated pages) where the annotators' judgments agree; $3,312$ sentences consisting of $1,399$ machine-translated and $1,913$ human-generated sentences. To replicate the situation in real Web pages, we conduct a minimal preprocessing, *i.e.*, simply removing HTML tags, and then feed all the remaining text to our method.

An SVM classifier is trained with features obtained by the LMs and gappy-phrases computed from the data described in Sec. 4.1. Our method shows 80.6% accuracy at a sentence level and 82.4% accuracy at a document level using the voting method. One factor for this performance difference is again sentence lengths, as SMT results of short phrases in Web pages can be of high-quality. Another factor is the noise in Web pages. We find that experimental pages contain lots of non-sentences, such as fragments of scripts and product codes. The results show that we need a preprocessing to remove typical noise in Web text before SMT detection to handle noisy Web pages.

5.4 Quality of Cleaned Data

Finally, we briefly demonstrate the effect of machine-translation filtering in an end-to-end scenario, taking LM construction as an example. We construct LMs reusing the Japanese evaluation dataset described in Sec. 4.1 where machine-translated sentences are removed by the proposed method (LM-Proposed), Lexical Feature (LM-LF), and Cross-Entropy (LM-CE), as well as an LM with all sentences, *i.e.*, *with* machine-translated sentences (LM-All). As a result of 5-fold cross-validation, LM-Proposed has 17.8%, 17.1%, and 16.3% lower perplexities on average compared to LM-All, LM-LF, and LM-CE, respectively. These results show that our method is useful for improving the quality of Web-mined data.

6 Conclusion

We propose a method for detecting machine-translated sentences from monolingual Web-text focusing on the phrase salad phenomenon produced by existing SMT systems. The experimental results show that our method achieves an accuracy of 95.8% for sentences and 80.6% for noisy Web text.

We plan to extend our method to detect machine-translated sentences produced by different MT systems, *e.g.*, a rule-based system, and develop a unified framework for cleaning various types of noise in Web-mined data. In addition, we will investigate the effect of source and target languages on translation in terms of MT detection. As Lopez (2008) describes, a phrase-salad is a common phenomenon that characterizes current SMT results. Therefore, we expect that our method is basically effective on different language pairs. We will conduct experiments to evaluate performance difference using various language pairs.

Acknowledgments

We sincerely appreciate Spencer Rarrick and Will Lewis for active discussion and sharing the experimental data with us. We thank Junichi Tsujii for his valuable feedback to improve our work.

References

Alexandra Antonova and Alexey Misyurev. 2011. Building a web-based parallel corpus and filtering out machine translated text. In *Proceedings of the Workshop on Building and Using Comparable Corpora*, pages 136–144.

Eleftherios Avramidis, Maja Popovic, David Vilar Torres, and Aljoscha Burchardt. 2011. Evaluate with confidence estimation: Machine ranking of translation outputs using grammatical features. In *Proceedings of the Workshop on Statistical Machine Translation (WMT 2011)*, pages 65–70.

Mohit Bansal, Chris Quirk, and Robert C. Moore. 2011. Gappy phrasal alignment by agreement. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, pages 1308–1317.

Marco Baroni and Silvia Bernardini. 2005. A new approach to the study of translationese: Machine-learning the difference between original and translated text. *Literary and Linguistic Computing*, 21(3):259–274.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 263–270.

Simon Corston-Oliver, Michael Gamon, and Chris Brockett. 2001. A machine learning approach to the automatic evaluation of machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2001)*, pages 148–155.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 1535–1545.

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

Google N-gram Corpus. 2006. `http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2006T13`.

Google Translate. 2006. `http://code.google.com/apis/language/`.

Iustina Ilisei, Diana Inkpen, Gloria Corpas Pastor, and Ruslan Mitkov. 2010. Identification of translationese: A machine learning approach. In *Proceedings of the International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2010)*, pages 503–511.

Tatsuya Ishisaka, Masao Utiyama, Eiichiro Sumita, and Kazuhide Yamamoto. 2009. Development of a Japanese-English software manual parallel corpus. In *Proceedings of the Machine Translation Summit (MT Summit XII)*.

Long Jiang, Shiquan Yang, Ming Zhou, Xiaohua Liu, and Qingsheng Zhu. 2009. Mining bilingual data from the web with adaptively learnt patterns. In *Proceedings of the Joint Conference of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP 2009)*, pages 870–878.

Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. Applying conditional random fields to Japanese morphological analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, pages 230–237.

David Kurokawa, Cyril Goutte, and Pierre Isabelle. 2009. Automatic detection of translated text and its impact on machine translation. In *Proceedings of the Machine Translation Summit (MT-Summit XII)*.

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2010. *Automated Grammatical Error Detection for Language Learners*. Morgan and Claypool Publishers.

Adam Lopez. 2008. Statistical machine translation. *ACM Computing Surveys*, 40(3):1–49.

Xiaoyi Ma. 2006. Champollion: a robust parallel text sentence aligner. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2006)*, pages 489–492.

Microsoft Translator. 2009. `http://www.microsofttranslator.com/dev/`.

Microsoft Web N-gram Services. 2010. `http://research.microsoft.com/web-ngram`.

Robert Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 220–224.

Ndapandula Nakashole, Gerhard Weikum, and Fabian M. Suchanek. 2012. PATTY: A taxonomy of relational patterns with semantic types. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 1135–1145.

Jian-Yun Nie, Michel Simard, Pierre Isabelle, and Richard Durand. 1999. Cross-language information retrieval based on parallel texts and automatic mining of parallel texts from the web. In *Proceedings of the Annual International ACM SIGIR Conference (SIGIR 1999)*, pages 74–81.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 311–318.

Kristen Parton, Joel Tetreault, Nitin Madnani, and Martin Chodorow. 2011. E-rating machine translation. In *Proceedings of the Workshop on Statistical Machine Translation (WMT 2011)*, pages 108–115.

Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Mei-Chun Hsu. 2001. PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *Proceedings of the International Conference on Data Engineering (ICDE 2001)*, pages 215–224.

Spencer Rarrick, Chris Quirk, and Will Lewis. 2011. MT detection in web-scraped parallel corpora. In *Proceedings of the Machine Translation Summit (MT Summit XIII)*.

Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47.

Lei Shi, Cheng Niu, Ming Zhou, and Jianfeng Gao. 2006. A DOM tree alignment model for mining parallel data from the web. In *Proceedings of the International Conference on Computational Linguistics and the Annual Meeting of the Association for Computational Linguistics (COLING-ACL 2006)*, pages 489–496.

Andreas Stolcke. 2002. SRILM-an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP 2002)*, pages 901–904.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of International Conference on World Wide Web (WWW 2007)*, pages 697–706.

Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (HLT-NAACL 2003)*, pages 252–259.

Vladimir N. Vapnik. 1995. The nature of statistical learning theory. *Springer*.

Jun Zhu, Zaiqing Nie, Xiaojiang Liu, Bo Zhang, and Ji-Rong Wen. 2009. StatSnowball: a statistical approach to extracting entity relationships. In *Proceedings of International Conference on World Wide Web (WWW 2009)*, pages 101–110.