机器翻译评测中的模糊匹配

刘洋[1,2]　刘群[1,3]

[1](中国科学院计算技术研究所，北京 100080)

[2](中国科学院研究生院，北京 100039)

[3](北京大学计算语言学研究所，北京 100871)

E-mail: {yliu, liuqun}@ict.ac.cn

摘要:

目前的大多数机器翻译自动评测方法都没有考虑在未匹配的词语中可能包含被忽略的信息。本文提出一种在参考译文和待评测译文之间自动搜索模糊匹配词对的方法，并给出了相似度的计算方法。模糊匹配和计算相似度的过程将通过一个例子进行说明。实验表明，我们的方法能够较好地找到被忽略的、有意义的词对。更重要的是，通过引入模糊匹配，BLEU的性能得到显著的提高。模糊匹配可以用来提高其他自动评测方法的性能。

关键词:

机器翻译评测；模糊匹配

1 Introduction

In recent years, many automatic metrics have been proposed for evaluating MT quality, as human evaluation is much expensive and time-consuming. The most important goal of automatic methods is to yield scores that correlate highly with human judgments of translation quality.

The dominant approach is to compute the closeness of a machine-translated sentence to several reference translations [1] [2]. Papineni's BLEU (Bilingual Evaluation Understudy) and Doddington's related NIST metric are two in common use today.

However, a serious problem for BLEU and NIST (may be include other metrics) is that they allow only **full matching** (e.g., two words are either matched or not). They treat matched words as relative to the source text and unmatched words as irrelative and not meaningful. Typically, there are many "perfect" translations of a given source text. These translations may vary in word choice. Thus, any deviations within MT output can be only partially attributed to errors. Although multiple reference translations are used to alleviate the problem, there are still some meaningful words that may be treated as unmatched words. In other words, some of the unmatched words are indeed irrelative to the source text while some not. Neglecting these meaningful words may limit the performance of N-gram translation evaluation metrics [3].

This paper proposes a fuzzy matching strategy for machine translation. The central idea is that we should allow the similarity of a word pair between zero and one. In section 2, we discuss the rationale of fuzzy matching and then demonstrate how to search fuzzy-matched word pairs and compute the similarity in detail. Experimental results and analysis are presented in section 3. The final section is the conclusion and our future work.

2 Fuzzy Matching

2.1 Similarity

Matching is a fundamental operation for automatic machine translation evaluation. When a word in a candidate translation is compared with another word in a reference translation to find out whether they are identical or not, this is called **matching**. We use **similarity** to denote the matching degree. Traditionally, if two words are identical, the similarity is one. If not, the similarity is zero. For instance, the similarity between *army* and *army* is 1, while the similarity between *army* and *military* is 0. As we can see, *army* and *military* are synonymous. If the candidate translation uses *military* instead of *army*, we should think that *military* is relative to source text and meaningful.

Thus, the point is that **the similarity of a word pair between candidate translation and reference translations should between 0 and 1**. In following sections, we will discuss how to find fuzzy-matched word pairs and how to figure out the similarity.

2.2 Graphical Representation

(Turian et al, 2003) proposed a new automatic method that uses unigram-based F-measure to measure MT quality [4]. We are especially interested in their idea that represents the matching between candidate translation and reference translations in bitext grid, which is an intuitionistic way to demonstrate matching. Here, we present an example to show how the bitext grid works. Note that our demonstration is somewhat different from (Turian et al, 2003). The sentences in the example have been used in (Papineni et al, 2002).

Example:
Ref: *It is a guide to action that ensures that military will forever heed party commands*.
Can: *It is to insure the troops forever hearing the activity guidebook that party direct*.

As shown in Figure 1, candidate text is on X-axis and reference text is on Y-axis. Each word gets its coordinate by its position in the text. For instance, *insure* is the fourth word in the candidate translation, so it is on x=4. Whenever a cell in the grid co-ordinates two words that are identical, we place a '■' in it, and call it a **point**. For example, point (3, 5) denotes the word pair {*to*, *to*}.

Contiguous sequences of matching words appear in bitext grid as diagonally adjacent points, running parallel to the main diagonal. We refer to such sequences as **runs**. As shown in Figure 1, "*it is*" is a run.

Now we introduce the notion of **collision**. If some points in the grid are in the same row or same column, we call that they are collided. Only one of the collided points will be reserved, the others will be deleted. In Figure 1, (5, 10) is collided with (9, 10). Usually, the corresponding word pairs of collided points are function words.

2.3 LCCSR

As we have discussed, the similarity of a word pair should between 0 and 1. But how do we compute the similarity of {*army*, *military*}? Obviously, linguistic resources such as WordNet are

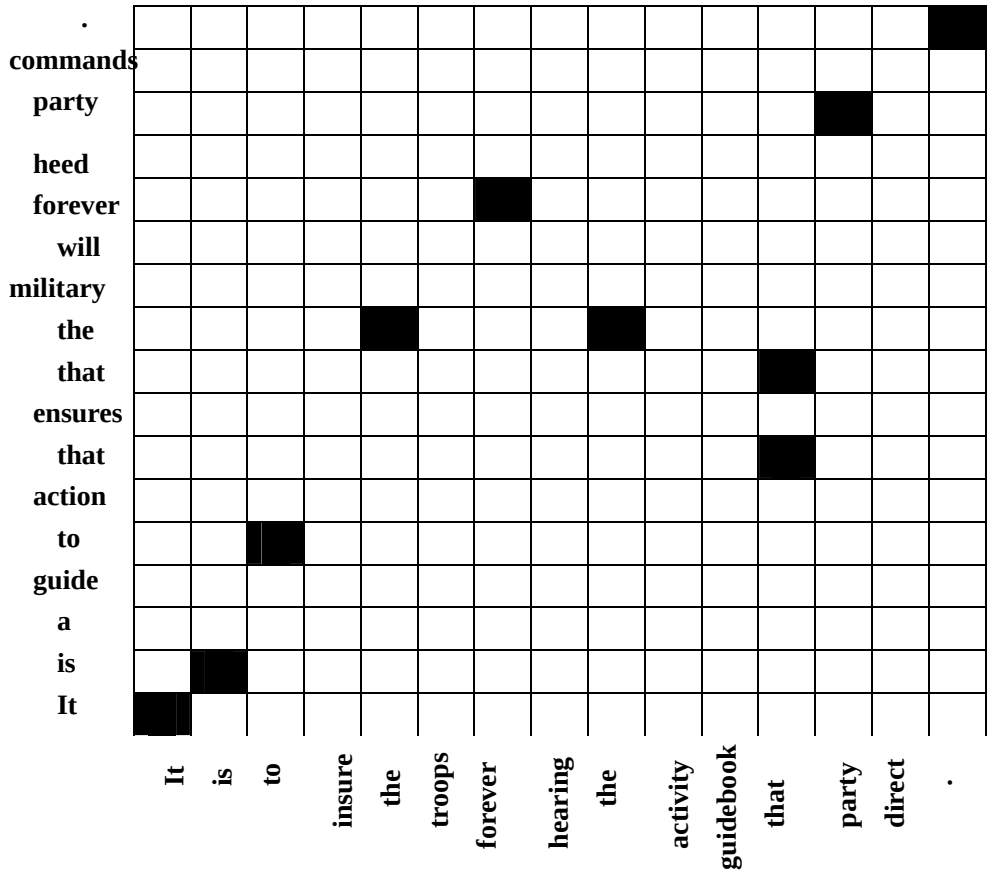

Figure 1: Full matching of Example

very useful and it is interesting to study how to use WordNet to find out the similarity of a word pair. But in the paper, we would like to seek a way to search fuzzy-matched word pairs and compute similarity independent of linguistic resources as we consider it more general. In the future, we will use linguistic resources to improve our approach.

We find that two kinds of word pairs should result in high similarities: synonyms and cognate words. For instance, {*army*, *military*} are synonyms and {*absent*, *absence*} are cognate words. For cognate words, we find that the more **continuous** characters two words share the more similar they are. We propose **LCCSR** (Longest Continuous Common String Ratio) to compute the similarity. LCCSR is similar to LCSR (Longest Common String Ratio), which is proposed by (Melamed 1995) to measure the cognate-ness for a pair of words in two languages [5]. We use LCCSR instead of LCSR because we think that continuous common strings are important to compute the similarity. LCCSR is computed as follows:

$$LCCSR(w_1, w_2) = \frac{|LCCS|}{\max\{|w_1|, |w_2|\}}$$

w_1 and w_2 is the word pair. $LCCSR$ is the longest continuous common string.

LCCSR is useful for cognate-similar word pairs, but cannot measure synonyms very well. For example, the LCCSR of {*army*, *military*} is 0.25. LCCSR is also probable to assign high similarity

to literal-similar but not cognate word pairs: the LCCSR of {*be*, *bee*} is 0.67. This problem will be solved by introducing linguistic resources. Currently, we alleviate the problem by allow matching only between content words (e.g. *be* is a function word and *bee* is a content word) and using structural information.

2.4 The Rationale of Fuzzy Matching Strategy

We refer to related but not identical word pairs as **fuzzy-matched word pairs** and the corresponding points in the bitext grid as **fuzzy-matched points**. For instance, {*army*, *military*} is a fuzzy-matched word pairs.

How do we discover these fuzzy-matched word pairs? Our strategy is to first make candidates and then filter them. We divide word in two categories: content words and function words. It is important that function words are countable. By listing all possible function words, we can easily distinguish content word from function word when a new word comes. We do comparison between content words, making assumption that every content word pair is possible to be fuzzy-matched. Thus, we get many word pairs. We call them **candidates**. We add the candidate points to the grid. Then, a deleting-point strategy is adopted to filter the candidate points. The remainder is considered relative to the source text and meaningful. At last, we compute similarity for every point.

The progress is described in detail as follows:

Step 1: Add full-matched points.

We only draw full-matched points. The result is shown in Figure 1. There are 10 full-matched points.

Step 2: Delete Collided full-matched points for the first time.

We present a property to a point: **runLen**, which is the length of the run that the point belongs to. If a point is isolated, its runLen is 1. If several points are collided and the runLen of some points are higher then the others, then reserve the points with the highest runLen and delete the others. If the runLens of collided points are equal, then no points will be deleted. So it is possible that there are collided points remain.

Step 3: Add candidate points.

Collecting all unmatched words in both reference and candidate translations; we add all possible candidate points between content words into the grid.

Step 4: Delete collided candidates points.

After study a lot of word pairs, we find that the higher LCCSR is, the more similar the two words would be. But if LCCSR is much lower, it cannot reflect similarity well. So we reserve points with LCCSR not lower than 0.5 and delete points collided with them. By observation, we find that the candidate point which is diagonally adjacent to the matched points (both full-matched and fuzzy-matched) may be a fuzzy-matched point. And the higher the runLens of its diagonally adjacent matched points are, the more possible the candidate point is a fuzzy-matched point. We introduce **connectiveness** to denote this property. First find all diagonally adjacent matched points, at most two. The connectiveness is the sum of the runLens of found matched points. If there are no such matched points, the connectiveness is 0. Thus, we reserve points with higher connectiveness and delete collided points. All candidate points with zero connectiveness will be deleted. This is because we have no enough context information to determine whether these candidate points are meaningful or not, although we may leave some

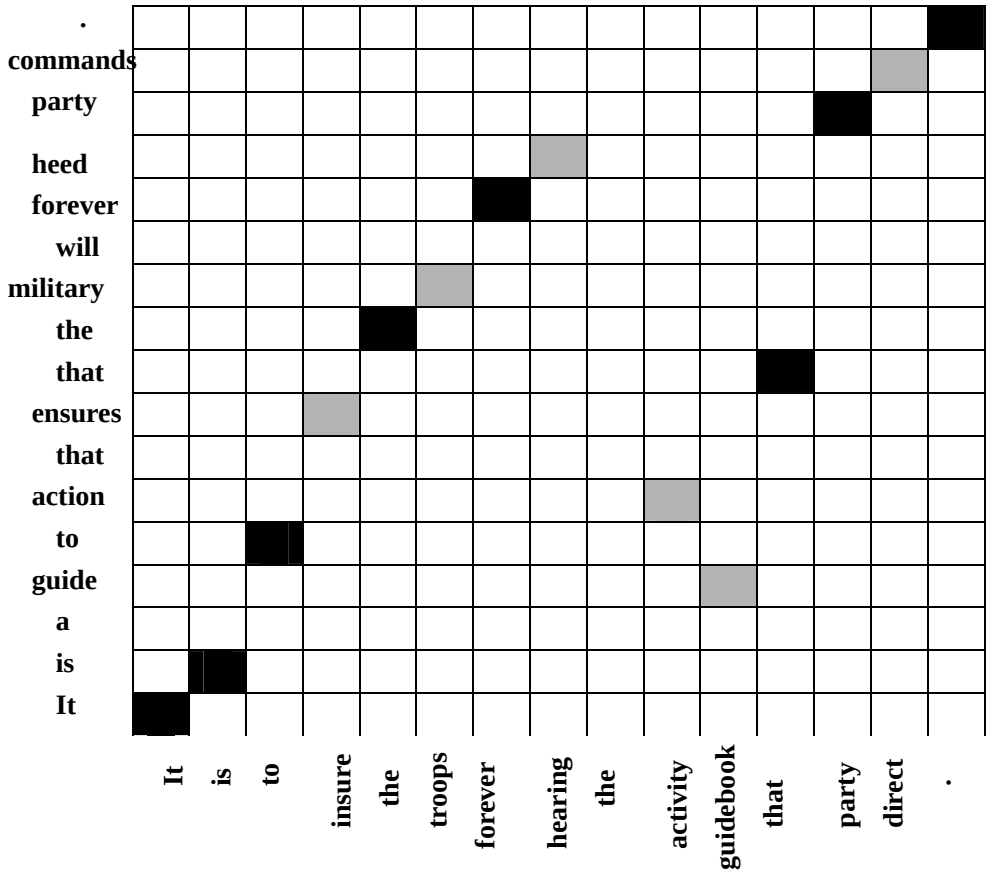

Figure 2: Fuzzy Matching of Example

meaningful candidate points uncovered.

Step 5: Delete collided full-matched points for the second time.

When comparing collided points with the same runLen, we prefer to reserve the one with lower dist to the main diagonal.

Step 6: Compute similarity.

After step 1- 5, the result is shown in Figure 2. Dark cells denote full-matched points and gray cells denote fuzzy-matched points.

We think that the similarity of a fuzzy-matched point is determined by two factors: LS (Literal Similarity) and SS (Structural Similarity). The similarity is computed as follows:

$$LS = \begin{cases} LCCSR & LCCSR \geq 0.5 \\ 0 & LCCSR < 0.5 \end{cases} \qquad SS = confidence * ratio_{run}$$

$$confidence = \frac{2 * count_{full-matched}}{count_{ref} + count_{can}} \qquad ratio_{run} = \frac{runLen}{runLen_{\max}}$$

(Tiedemann, 2003) introduced an approach for combing all clues, which uses disjunction of all indications [6]. As we think that LS and SS are independent, so the similarity is computed as follows:

$$similarity = LS + SS - LS * SS$$

In Table 1 we listed the similarity of all fuzzy-matched points in the grid.

Table 1: The similarity of all fuzzy-matched points.

Point	Word pair	Similarity
(4, 8)	{insure, ensures}	0.7619
(6, 11)	{troops, military}	0.3333
(8,14)	{hearing, heed}	0.3333
(10, 6)	{activity, action}	0.5833
(11, 4)	{guidebook, guide}	0.6296
(14, 16)	{direct, commands}	0.5

3 Experiment

We designed two experiments to investigate the applicability of fuzzy matching strategy.

The first experiment is set to find out how well our fuzzy matching strategy to catch meaningful unmatched word pairs. We use 100 Chinese sentences as source texts, and then prepare one reference translation produced by a professional translator and one candidate translation by a MT system. The text is mixed with both short and long sentences. There are more short sentences than long sentences, resulting in 11.795 words per sentence in reference translation and 11.925 words per sentence in candidate translation. We use precision, recall and F-measure to judge the predicting ability of fuzzy matching strategy. If a professional translator thinks that an unmatched word pair (one from reference translation and one from candidate translation) is relative and FMS also does, we consider FMS makes the right decision. If not, it made a wrong decision.

The result is show in Table 2.

Table 2: Precision, recall and F-measure

Precision	Recall	F-measure
79.33%	78.81%	79.07%

After careful study, we find that most predicting errors occurs when structural similarity is being used to determine the whether the point should be reserved or not. As we have mentioned, structural similarity are not so reliable; especially when there are few full-matched points. This problem can be alleviated by assigning lower similarity to SS-determined points.

The second experiment is to look into how fuzzy matching can improve current metrics. We compare BLEU baseline and extended BLEU using fuzzy matching. The source texts are Chinese dialogs[1], broken into 200 segments. We prepared four reference translations in English written by professional translators and used five MT systems to produce the candidate translations. Candidate translations were then evaluated by four other professional translators,

[1] Because dialogs usually contain only one sentence, and fuzzy matching will fully show its predicting ability within one sentence. Although the text is dialogs, there are still many pretty long and complex sentences.

ranked from 0 to 6. After collecting all rankings for every segment, we normalize the human judgment to a value between 0 and 1.

Fuzzy matching is integrated into BLEU by modifying the way of counting, that is, by counting similarity. For instance, when comparing two N-grams: $w_1 w_2 ... w_n$ and $w'_1\ w'_2 ... w'_n$, the similarity is $\min\{similarity(w_i, w'_i)\}$.

The result is shown in Table 2. S1, S2, S3, S4, S5 stand for the five MT systems. 'P' denotes Pearson coefficient and 'S' denotes Spearman coefficient.

Table 3: Comparison between BLEU baseline and BLUE using fuzzy matching.

	Human	Full-matched	Fuzzy-matched
S1	0.6158	0.1853	0.3410
S2	0.4384	0.1689	0.3190
S3	0.4463	0.1205	0.2875
S4	0.7316	0.3894	0.4998
S5	0.5039	0.1683	0.3218
S		0.7000	0.9000
P		0.8880	0.9036

In Table 3, we can find that after using fuzzy matching, Pearson coefficient is improved. More important is that Spearman coefficient is also significantly improved, as fuzzy-matched BLEU made the right ranking between S2 and S5. The encouraging results indicate that fuzzy matching is subtler and is possible to improve other automatic metrics.

4 Conclusion

Most current automatic metrics of machine translation evaluation do not consider that among unmatched words there may be neglected information. In the paper, we describe a strategy to find fuzzy-matched word pairs between reference and candidate translation automatically and propose an approach to compute the similarity.

Our experiments show that FMS can find neglected meaningful word pairs pretty well. More importantly, the performance of BLEU is improved by integrating fuzzy matching. Fuzzy matching is possible to be utilized to improve other automatic methods.

Although fuzzy matching is promising, there are still a lot of issues to be solved. We think that finding potential meaningful word pairs more precisely and perfect the way of computing similarity is the two key issues in fuzzy matching, which should be more deeply studied. We must keep cautious about the predicting ability of FMS, as FMS may find many non-meaningful word pairs when the sentence is much long and complex. New clues can be proposed to improve FMS. Now, we simply weaken the side effect by lowering the similarity. In addition, we have only studied fuzzy matching in English; it will be extended to other languages. In the future, we will utilize linguistic resources such as WordNet and HowNet to improve our approach.

参考文献：

[1] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation Evaluation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. 2002.

[2] George Doddington. Automatic Evaluation of Machine Translation Quality Using N-gram Co-Occurrence Statistics. In Human Language Technology: Notebook Proceedings: 128-132. 2002.

[3] Christopher Culy and Susanne Z. Riehemann. The Limits of N-Gram Translation Evaluation Metrics. In Proceedings of Machine Translation Summit IX. 2003.

[4] Joseph Turian, Luke Shen, and I. Dan Melamed. Evaluation of Machine Translation and its Evaluation. In Proceedings of Machine Translation Summit IX. 2003.

[5] I. Dan Melamed. Automatic Evaluation and Uniform Filter Cascades for Inducing N-best Translation Lexicons. Third Workshop on Very Large Corpora. 1995.

[6] Jörg Tiedemann. Combing Clues for Word Alignment. In Proceedings of 10th Conference of European Chapter of ACL (EACL03). 2003.

作者简介：

刘洋，男，1979 年生，中国科学院计算技术研究所博士生，研究方向为统计机器翻译。

刘群，男，1966 年生，中国科学院计算技术研究所副研究员，研究工作主要集中在与自然语言处理（特别是中文处理）相关的理论、技术、方法和应用系统。

Fuzzy Matching in Machine Translation Evaluation

LIU Yang[1, 2] LIU Qun[1, 3]

[1](*Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080, China*)

[2](*Graduate School of Chinese Academy of Sciences, Beijing 100039, China*)

[3](*Institute of Computational Linguistics, Peking University, Beijing 100871, China*)

E-mail: {yliu, liuqun}@ict.ac.cn

Abstract:

Most current automatic metrics of machine translation evaluation do not consider that among unmatched words there may be neglected information. In this paper, we describe a strategy to find fuzzy-matched word pairs between reference and candidate translations automatically and propose an approach to compute the similarity. The whole process of finding fuzzy-matched word pairs and computing their similarity is demonstrated in detail. Experiments show that our method is capable of finding neglected meaningful word pairs fairly well. More importantly, the performance of BLEU is significantly improved by integrating fuzzy matching. Fuzzy matching is possible to be utilized to improve other automatic methods.

Key words:

machine translation evaluation; fuzzy matching