A Metric for Paraphrase Detection

João Cordeiro
Centre of Human Language
Technology and Bioinformatics
University of Beira Interior
Covilhã, Portugal
Email: jpaulo@di.ubi.pt

Gaël Dias
Centre of Human Language
Technology and Bioinformatics
University of Beira Interior
Covilhã, Portugal
Email: ddg@di.ubi.pt

Pavel Brazdil
Artificial Intelligence and
Computer Science Laboratory
University of Porto
Porto, Portugal
Email: pbrazdil@liacc.up.pt

***Abstract*— Monolingual text-to-text generation is an emerging research area in Natural Language Processing. One reason for the interest in such generation systems is the possibility to automatically learn text-to-text generation strategies from aligned monolingual corpora. In this context, paraphrase detection can be seen as the task of aligning sentences that convey the same information but yet are written in different forms, thereby building a training set of rewriting examples. In this paper, we propose a new metric for unsupervised detection of paraphrases and test it over a set of standard paraphrase corpora. The results are promising as they outperform state-of-the-art measures developed for similar tasks.**

I. Introduction

Monolingual text-to-text generation is an emerging research area in Natural Language Processing. Unlike in traditional concept-to-text generation, text-to-text generation applications take a text as input and transform it into a new text satisfying specific constraints, such as length in summarization [3], [8], [9], [10], [18] or style in text simplification [14].

One reason for the interest in such generation systems is the possibility to automatically learn text-to-text generation strategies from aligned monolingual corpora. Such text collections are usually called paraphrase corpora. In fact, text-to-text generation is a particularly promising research direction given that there are naturally occurring examples of comparable texts that convey the same information yet are written in different styles. Web news are an obvious example of these non-parallel corpora. So, presented with such texts, one can pair sentences that convey the same information, thereby building a training set of rewriting examples i.e. a paraphrase corpus. These pairs of sentences are called paraphrases and share almost the same meaning, but contain different lexical elements and possibly different syntactical structure.

However, the unsupervised methodologies proposed so far [3], [6] are not well tailored for the reality and special needs of paraphrase detection, showing a major drawback, by extracting quasi-exact or even exact match pairs of sentences as they rely on classical string similarity measures such as the Edit Distance in the case of [6] and word overlap for [3]. Such pairs are obviously useless.

As a consequence, we propose a new metric - named the *Sumo-Metric* - that solves these limitations and outperforms all state-of-the-art metrics both in the general case where exact and quasi-exact pairs do not occur and in the real-world case where exact and quasi-exact pairs occur (like in web news stories).

In fact, the *Sumo-Metric* extracts a great deal of pairs of non-symmetric entailed sentences. For example, it can identify as paraphrase a pair $\langle S_a, S_b \rangle$ where the sentence S_a entails sentence S_b ($S_a \trianglerighteq S_b$), but S_b does not entail S_a ($S_a \ntrianglelefteq S_b$), or vice-versa:

> S_a**:** *The control panel looks the same but responds more quickly to commands and menu choices.*
>
> S_b**:** *The control panel responds more quickly.*

This particular case is much more challenging for classical string similarity measures that do not have been tailored for paraphrase detection but instead for exact match of string pairs.

II. Related Work

The issue of finding paraphrases in monolingual comparable corpora is recently becoming more and more relevant as researchers realize the importance of such resources for Information Retrieval, Information Extraction, Automatic Text Summarization and Automatic Text Generation [3], [8], [9], [10], [14], [18].

In particular, three different approaches have been proposed for paraphrase detection: unsupervised methodologies based on lexical similarity [3], [6], supervised methodologies based on context similarity measures [4] and methodologies based on linguistic analysis of comparable corpora [7].

[6] endeavored a work to find and extract monolingual paraphrases from massive comparable news stories. They use the Edit Distance (also known as *Levenshtein Distance* [11]) and compare it with an heuristic derived from Press writing rules. The evaluation shows that the data produced by the Edit Distance is cleaner and more easily aligned than by using the heuristic. However, using word error alignment rate results show that both techniques perform similarly.

[3] used the simple word n-gram ($n = 1, 2, 3, 4$) overlap measure in the context of paraphrase lattices learning. In particular, this string similarity measure is used to produce clusters of paraphrases using hierarchical complete-link clustering.

More deepening techniques rely on context similarity measures such as [4]. They find sentence alignments in comparable corpora by considering sentence contexts (local alignment) after semantically aligning equivalent paragraphs. Although this methodology shows interesting results, it relies on supervised learning techniques, which needs huge quantities of training data that may be scarce and difficult to obtain.

Others [7] go further by exploring heavy linguistic features combined with machine learning techniques to propose a new text similarity metric. Once again it is a supervised approach and also heavily dependent on valuable linguistic resources which is not available for the vast majority of languages.

III. Metrics Overview

In the literature [3], [11], [15], we can find the *Levenshtein Distance* [11] and what we call the *Word N-Gram Overlap Family* [3], [15]. Indeed in the latter case, some variations of word n-gram overlap measures are proposed but not clearly explained. In this section, we will review all the existing metrics and propose an enhanced n-gram overlap metric based on LCP (Longest Common Prefix) [21].

A. The Levenshtein Distance

The Levenshtein Distance, also known as the Edit Distance, is a well-known metric [11] that may be adapted for calculating *Sentence Edit Distance* upon words instead of characters [6]. Considering two strings, it computes the number of character/words insertions, deletions and substitutions that would be needed to transform one string into the opposite.

A problem, when using the Edit Distance for the detection of paraphrases, is the possibility that there exist sentence pairs that are true paraphrases but are not identified as such. In fact, if the sentences show high lexical alternations or different syntactical structures they are unlikely defined as similar.

B. The Word N-Gram Family

In fact, we found not only one, but a set of text similarity measures based on word n-gram overlap in the literature. Sometimes it is not clear or unspecified which word n-gram version is used. In fact, two metrics are usually found in the literature (the Word Simple N-gram Overlap and the BLEU Metric). But, in order to be complete, we propose a third metric based on the LCP paradigm.

1) Word Simple N-gram Overlap: This is the simplest metric that uses word n-gram overlap between sentences. For a given sentence pair, the metric counts how many 1-grams, 2-grams, 3-grams, ..., N-grams overlap. Usually N is chosen equal to 4 or less [3]. Let's name this counting function $Count_{match}(\text{n-gram})$. For a given $N \geqslant 1$, a normalized metric that equally weights any matching n-gram and evaluates similarity between sentences S_a and S_b, is given in Equation 1:

$$sim_o(S_a, S_b) = \frac{1}{N} * \sum_{n=1}^{N} \frac{Count_{match}(\text{n-gram})}{Count(\text{n-gram})} \qquad (1)$$

where the function $Count(\text{n-gram})$ counts the maximum number of n-grams that exist in the shorter sentence as it rules the max number of overlapping n-grams.

2) Exclusive LCP N-gram Overlap: In most work in Natural Language Processing, the longest a string is, the more meaningful it should be [5]. Based on this idea, we propose an extension of the word simple n-gram overlap metric. The difference between simple and exclusive n-gram overlap lays on the fact that the exclusive form counts prefix overlapping 1-grams, 2-grams, 3-grams, ..., N-grams, regarding the Longest Common Prefix (LCP) paradigm proposed by [21]. For example, if some maximum overlapping 4-gram is found then its 3-grams, 2-grams and 1-grams prefixes will not be counted. Only the 4-gram and its suffixes will be taken into account. This is based on the idea that the longer the match the more significant the match will be. Therefore smaller matches are discarded. As an example, consider the two sentences:

(3) *The President ordered the final strike over terrorists camp.*

(4) *President ordered the assault.*

Between these sentences we have the LCP n-gram overlap given by: "President ordered the" which is a 3-gram. So the complete set of overlapping n-grams, besides the 3-gram, is: "ordered the" (2-gram) and "the" (1-gram), i.e all its suffixes.

If one wants to normalize the n-gram overlap then a particular difficulty rises due to the LCP n-gram considerations i.e. the maximum number of overlapping n-grams depends on the number of (n+1)-gram overlaps that exist. For example, in the previous case and for 1-grams, we only have one overlapping 1-gram ("the") between the two sentences and not 3 as it could be computed with the word simple n-gram overlap metric i.e. "the", "President" and "ordered". Thus, with this process of considering exclusive n-grams, it is unlikely to compute similarity based on a weighted sum like in formula 1. Another method, more suitable, is used and it is expressed by Equation 2:

$$sim_{exo}(S_a, S_b) = \max_{\text{n}} \left\{ \frac{Count_{match}(\text{n-gram})}{Count(\text{n-gram})} \right\} \qquad (2)$$

where S_a and S_b are two sentences and the following functions $Count_{match}(\text{n-gram})$ and $Count(\text{n-gram})$ are the same as above with this new matching strategy i.e. we first calculate $sim_{exo}(S_a, S_b)$ for 4-grams and then for the remaining 3-grams and so on and so forth, and then choose the maximum ratio.

3) The BLEU Metric: The BLEU metric was introduced by [15] for automatic evaluation of machine translation, and after used to automatically evaluate summaries [12]. It is clear that this metric can easily be adapted to calculate similarity between two sentences as it is based on the calculation of string overlaps between texts. The adapted formula is given below in Equation 3:

$$BLEU_{adapted} = \frac{1}{N} * exp[\sum_{n=1}^{N} log \sum_{\text{n-gram}} \frac{Count_{match}(\text{n-gram })}{Count(\text{n-gram })}] \qquad (3)$$

The $Count_{match}$(n-gram) function counts the number of exclusive or no-exclusive n-grams co-occurring between the two sentences, and the function $Count$(n-gram) the maximum number of n-grams that exists in the shorter sentence.

IV. The Proposed Sumo-Metric

Our main research area lays in the field of Automatic Sentence Compression where the paraphrase issue is a relevant one. We see paraphrase clusters as nice raw material to discover Sentence Compression patterns as in [3], [9], [10], [18].

A. Motivation

Based on the previous statement, we propose the automatic construction of a huge paraphrase corpus valuable for our main research task. It is not the first work in automatic paraphrase corpus construction [3], [6] but it is the only one that clearly addresses the problems of existing string similarity metrics. Indeed, when applying existing metrics for paraphrase detection, most of the results are exact or quasi-exact match pairs of sentences. Such results are obviously useless.

For that purpose, we designed a new metric to detect paraphrases that avoids the extraction of exact and quasi-exact matches and outperforms state-of-the-art metrics in all evaluation situations presented in VII. In fact, four main premises guided our research: (1) Achieve maximum automation in corpus construction - minimum or even no human intervention, with high reliability, (2) Penalize equal and almost equal sentences - they are not useful for our research needs, but frequent in real-world situations, (3) Consider pairs having a high degree of lexical reordering, and different syntactic structure, (4) Define a computationally fast and well founded metric.

The basic idea of the *Sumo-Metric* lays on the notion of exclusive lexical links between a sentence pair, as shown in figure 1.

It is another form to think about 1-gram exclusive overlap. If a link is established between sentence S_a and sentence S_b, for the word **w**, then other occurrences of word **w** in sentence S_a will engage a new link to sentence S_b if there exists at least one more occurrence of **w** in S_b, besides the one which is already connected.

B. Definition

First, we introduce some notions to understand the definition of our metric. The number of links between the two sentences are defined as λ and the number of words in the longest and shortest sentence as x and y, respectively. In the previous example, we have $x = 16$, $y = 13$, and $\lambda = 9$. As we can see, the fractions $\frac{\lambda}{x}$ and $\frac{\lambda}{y}$ are values in the interval $[0, 1]$, indicating some normalized lexical connectivity among sentences. In particular, in our example, we have $\frac{\lambda}{x} = \frac{9}{16}$ and $\frac{\lambda}{y} = \frac{9}{13}$.

To calculate the *Sumo-Metric* $S(.,.)$, we first evaluate the function $S(x, y, \lambda)$ as in Equation 4

$$S(x, y, \lambda) = \alpha \log_2(\frac{x}{\lambda}) + \beta \log_2(\frac{y}{\lambda}) \tag{4}$$

where $\alpha, \beta \in [0, 1]$ and $\alpha + \beta = 1$. After that, we compute the *Sumo-Metric* $S(.,.)$ as in Equation 5.

$$S(S_a, S_b) = \begin{cases} S(x, y, \lambda) & if \ \ S(x, y, \lambda) < 1.0 \\ e^{-k*S(x,y,\lambda)} & otherwise \end{cases} \tag{5}$$

With the α and β parameters, one may weight the value of the two main components involved in the calculation as in any linear interpolation. For example, to give more relevance to the component that depends on $\frac{\lambda}{y}$ (the shortest sentence), let β be superior to 0.5. In our experiments we equally weighted both components, i.e. $\alpha = \beta = 0.5$[1]. The effect of using the $log_2(.)$ function is to gradually penalize pairs that are very similar - remark that for equal pairs the result is exactly zero.

The second branch of function 5 guarantees that the metric never returns values greater than 1.0. Theoretical work shows that this is the case when $x^\alpha y^\beta > 2\lambda$. For $\alpha = \beta = 0.5$, this occurs when $\sqrt{xy} > 2\lambda$ or $xy > 4\lambda^2$. As an example, let us consider the following two situations:

1) $\langle x, y, \lambda \rangle = \langle 15, 6, 5 \rangle \;\Rightarrow\; S(x, y, \lambda) = 0.924$
2) $\langle x, y, \lambda \rangle = \langle 30, 6, 5 \rangle \;\Rightarrow\; S(x, y, \lambda) = 1.424$

The first example is clearly a relevant situation. However, the second example is over-evaluated in terms of similarity. As a consequence, $e^{-k*S(x,y,\lambda)}$ is a penalizing factor, where the constant k is a tuning parameter[2] that may scale this factor more or less. In particular, we can see its effect as follows.

2. $\langle x, y, \lambda \rangle = \langle 30, 6, 5 \rangle \;\Rightarrow\; e^{-k*S(x,y,\lambda)} = 0.014$

In fact, when sentences tend to be very asymmetric, in number of words, the computation of $S(x, y, \lambda)$ gives values greater than 1.0, despite the number of links that exist. So, the higher $S(x, y, \lambda)$ is beyond 1.0, the more unlikely the pair will be classified as positive with respect to $S(.,.)$.

C. Complexity

The *Sumo-Metric* is computed in $\Theta(x * y)$ time, in the worst case - when the sentences are completely different, i.e. there is no link among them. In that case, we compute $x * y$ comparisons i.e. each word in the longest sentence is compared with each word from the shortest one. In the best situation the computation will take only $\Theta(y)$ time. This is the case when the shortest sentence is a prefix of the longest one. In terms of comparison, all metrics show time complexity $\Theta(x*y)$ except the *exclusive LCP n-gram overlap* metric that evidences time complexity $\Theta((x + y)log(x + y))$, which is better for large values of x and y [3].

The application of any metric in paraphrase detection on a large collection of text, has the same complexity of $\Theta(n^2/2)$, where n is the number of sentences present in the collection,

[1] Best results were obtained in this case for the used corpora set.

[2] $k = 3$ was used in our experiments.

[3] However, x and y are relatively small, on average (less than 50 words).

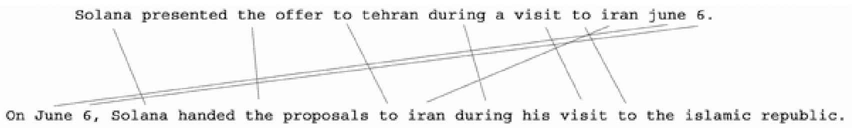
Solana presented the offer to tehran during a visit to iran june 6.
On June 6, Solana handed the proposals to iran during his visit to the islamic republic.

Fig. 1. Links between a sentence pair.

because sentence i has to be compared with the remaining $n - i$ sentences.

V. The Corpora Set

Two standard corpora were used for comparative tests between metrics: The Microsoft Research Paraphrase Corpus [6] and a corpus supplied by Daniel Marcu that has been used for research in the field of Sentence Compression [9], [10]. By adapting these corpora we created three new corpora to serve as a benchmark for our specific purpose.

A. The Microsoft Paraphrase Corpus

In 2005, Microsoft researchers Dolan, Brocket, and Quirck [6] published the first paraphrase corpus containing 5801 pairs of sentences with 3900 tagged as "semantically equivalent" or true paraphrases. Sentences were obtained from massive parallel news sources and tagged by 3 human raters according to guidelines described in [6]. We will refer to this corpus as the label $\{MSRPC\}$.

B. The Knight and Marcu Corpus.

The corpus used by [9] in their Sentence Compression research work, contains 1087 sentence pairs, where one sentence is a compressed or summarized version of the other one. This corpus was produced completely manually from pairs of texts and respective summaries. We labeled this corpus as $\{KMC\}$.

C. The Corpora Used

One major limitation with the $\{KMC\}$ corpus is that it only contains positive pairs. Therefore it should not be taken as such to perform any evaluation. Indeed, we need an equal number of negative pairs of sentences to produce a fair evaluation for any paraphrase detection metric. Although the $\{MSRPC\}$ corpus already contains negative pairs, they are only 1901 against 3900 positive examples. To perform an equitable evaluation, we first expanded both corpora by adding negative sentence pairs selected from *Web News Corpora* so that they have the same number of positive and negative examples and also created a new corpus based on the combination of the $\{MSRPC\}$ and $\{KMC\}$.

1) The $\{MSRPC \cup X^-_{1999}\}$ Corpus: This new derived corpus contains the original $\{MSRPC\}$ collection of 5801 pairs (3900 positives and 1901 negatives) plus 1999 extra negative sentences (symbolized by X^-_{1999}), selected from web news stories. So we end with 3900 positive pairs and 3900 negative ones.

2) The $\{KMC \cup X^-_{1087}\}$ Corpus: From the $\{KMC\}$, we derived a new corpus that contains its 1087 positive pairs plus a set of negative pairs, in equal number, selected from web news stories. We named this new corpus $\{KMC \cup X^-_{1087}\}$, where the X^-_{1087} stands for extra negative paraphrase pairs (1087 in this case).

3) The $\{MSRPC^+ \cup KMC \cup X^-_{4987}\}$ Corpus: Finally we decided to build a bigger corpus that gathers the positive $\{MSRPC\}$ part i.e. 3900 positive examples, and the 1087 positive pairs of sentences from the $\{KMC\}$ corpus, giving a total of 4987 positive pairs. To balance these positive pairs we added an equal number of negative pairs, selected in a same fashion as described previously. We labeled this wider corpus as the $\{MSRPC^+ \cup KMC \cup X^-_{4987}\}$ corpus. In this corpus we intentionally ignored the $\{MSRPC\}$ negative pairs as many pairs that are labeled negative, following the guidelines expressed in [6], are in fact useful paraphrases.

VI. Results

This work does not only propose a new metric for finding paraphrases, but also gives a comparative study between already existing metrics and new adapted ones, and proposes a new benchmark of paraphrase test corpora. In particular, we tested the performance of 5 metrics over 3 corpora and will show the results achieved by each metric over each corpus.

A. How to Classify a Paraphrase?

Before presenting the results, it is is necessary to talk about a classical problem in classification - *thresholds*. Usually, for a classification problem, a system takes decisions upon some parameters, which are called thresholds. In our case, we present metrics that calculate some similarity[4] between sentences. However, after this computation it is necessary to

[4] Or dissimilarity, in the *Levenshtein Distance* case.

TABLE I

THRESHOLDS MEAN AND STANDARD DEVIATION

thresholds	A	B	C
edit	17.222 ± 0.1109	20.167 ± 1.3751	17.312 ± 0.00
sim_o	0.2030 ± 0.0068	0.2604 ± 0.0026	0.2537 ± 0.00
sim_{exo}	0.5006 ± 0.0000	0.7250 ± 0.0130	0.5005 ± 0.00
bleu	0.5024 ± 0.0002	0.5005 ± 0.0000	0.5005 ± 0.00
sumo	0.0765 ± 0.0035	0.0053 ± 0.0006	0.0069 ± 0.00

decide upon some value, the threshold, what is a paraphrase and what is not.

Thresholds are parameters that unease the process of evaluation. Indeed, the best parameter should be determined for each metric. However, this is not always the case and wrong evaluations are proposed. In our evaluation, we do not pre-define any threshold for any metric. Instead, for each metric, we automatically compute the best threshold. This computation is a classical problem of function maximization or optimization [16]. In particular, we use the bisection strategy as it computes fast, and well approximates the global maximum of the smooth curve of our functions. As a result, we are optimizing the value of the threshold for each metric in the same way and do not introduce any subjectivity in the choice of the parameters.

In Table I, we present the obtained thresholds for the five compared metrics using a *10-fold cross validation* scheme. The results show that the bisection strategy performs well for our task as the standard deviation for each measure and corpus is almost negligible. In the remainder of this paper, we renamed $\{MSRPC \cup X^-_{1999}\}$ as **A**, $\{KMC \cup X^-_{1087}\}$ as **B** and $\{MSRPC^+ \cup KMC \cup X^-_{4987}\}$ as **C** in order to ease the reading.

B. Experiments and Results

In order to evaluate the results of each metric over each corpus, we computed both the F-Measure and the Accuracy measures. In particular, the results were calculated by averaging the 10 F-Measure and Accuracy values obtained from the *10-fold cross validation* test executed over the data. For every fold, the best threshold was found on the $\frac{9}{10}$ training data and then used on the $\frac{1}{10}$ test block to measure the correspondent F-Measure and Accuracy. The overall results are presented in Table II and III. The F-measure and the Accuracy are respectively defined in Equation 6 and 8. In particular, the experiments with the F-Measure were made with $\beta = 1$.

TABLE II

$F - Measure$ OBTAINED.

	A	B	C
edit	74.41%	70.65%	80.98%
sim_o	78.06%	94.66%	91.92%
sim_{exo}	77.27%	90.87%	87.19%
bleu	70.77%	82.39%	76.79%
sumo	**80.92%**	**98.45%**	**98.53%**

TABLE III

Accuracy OBTAINED.

	A	B	C
edit	67.67%	68.02%	79.02%
sim_o	73.15%	94.47%	91.79%
sim_{exo}	72.37%	90.23%	86.00%
bleu	66.17%	78.89%	74.13%
sumo	**78.19%**	**98.43%**	**98.53%**

$$F_\beta = \frac{(1+\beta^2) * precision * recall}{\beta^2 * precision + recall} \tag{6}$$

with

$$precision = \frac{TP}{TP+FP}, recall = \frac{TP}{TP+FN} \tag{7}$$

where TP are True Positives, TN True Negatives, FP False Positives and FN False Negatives.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{8}$$

The results evidenced in Table II show that the *Sumo-Metric* outperforms all state-of-the-art metrics over all corpora. For instance, on the biggest corpus (C), the *Sumo-Metric* correctly classified, on average, 98.53% of all the 9974 sentence pairs, either positives or negatives. It shows systematically better F-Measure and Accuracy measures over all other metrics showing an improvement of (1) at least 2.86% in terms of F-Measure and 3.96% in terms of Accuracy and (2) at most 6.61% in terms of F-Measure and 6.74% in terms of Accuracy compared to the second best metric which is also systematically the sim_o similarity measure.

Another interesting result is the fact that all metrics behave the same way over all corpora. The simple word n-gram overlap (sim_o) and the exclusive lcp n-gram overlap (sim_{exo}) metrics always get second and third places, respectively and the BLEU metric and the Edit Distance obtain the worst results over all corpora being respectively fourth and fifth. The only exception is for the BLEU measure compared to the Edit Distance for the **A** corpus. These results are not surprising as we already told that the **A** corpus contains lots of negative pairs that should be taken as positive examples. Indeed, almost all positive examples are near string matches[5].

C. The Influence of Random Negative Pairs

One may criticize that the superior performance obtained by the *Sumo-Metric* depends exclusively on the set of equal or quasi-equal pairs which are present in the corpora. However, this is not the case. Indeed, to acknowledge this situation, we performed another experiment with a corpus similar to the **C** corpus (the biggest one) but without any quasi-equal or equal pair. Let's call it the **C'** corpus. The performance obtained over the **C'** is illustrated in Table IV and clearly shows that the *Sumo-Metric* outperforms all other state-of-the-art metrics in all evaluation situations, even when equal or quasi-equal pairs are not present in the corpora.

[5] It is important to remind here that the **A** corpus was previously computed with the Edit Distance.

TABLE IV

CORPUS WITHOUT QUASI-EQUAL OR EQUAL PAIRS

Accuracy %	edit	sim_o	sim_{exo}	bleu	sumo
C'	84.31	96.36	90.19	77.98	99.58

In this case, we only show the Accuracy measure as the F-measure evidences similar results. This give us at least 99% statistical confidence[6] (1% significance) that $Accuracy_{sumo} > Accuracy_{simx}$, where $simx \in \{edit, sim_o, sim_{exo}, bleu\}$ (any other tested metric).

VII. CONCLUSION AND FUTURE WORK

In this paper, we proposed a new metric, the *Sumo-Metric*, for finding paraphrases. But, we also performed a comparative study between already existing metrics and new adapted ones and proposed a new benchmark of paraphrase test corpora. In particular, we tested the performance of 5 metrics over 4 corpora. One main and general conclusion is that the *Sumo-Metric* performed better than any other measure over all corpora either in terms of F-Measure and Accuracy. Moreover, the Word Simple N-gram Overlap and the Exclusive LCP N-gram Overlap are systematically second and third in the ranking over all corpora, thus negating [5]'s assumption for the task of paraphrase detection. Finally, the *Levenshtein Distance* [11] performs poorly over corpora with high lexical and syntactical diversity unlike the BLEU measure. However, when paraphrases are almost string matches, the Edit Distance outperforms the BLEU measure. Nevertheless, in all cases, we must point at that the Edit Distance and the BLEU measure are always classified fourth or fifth in the ranking.

In the future we will try to insert tf.idf [17] information in our metric, as we believe that word links between sentences should have distinct weights. Indeed, it is different to have a match between determinants (with low tf.idf) or between verbs or nouns/names (with high tf.idf). Verbs and nouns/names obviously convey relevant information about the sentence while it is not the case for determinants. We may also integrate the notion of content character n-grams that can be extracted from monolingual corpora as in [5]. Finally, [3] propose a clustering methodology to group similar sentences (i.e. paraphrases) into clusters. We already made some preliminary experiments in this direction that show promising results with an adaptation of the Q clustering algorithm.

ACKNOWLEDGMENT

We are grateful to Daniel Marcu for providing us with his corpus and we would also thank the Portuguese *Fundação para a Ciência e a Tecnologia* agency for funding this research (Reference: POSC/PLP/57438/2004).

REFERENCES

[1] R. Barzilay and K. McKeown *Extracting Paraphrases from a Parallel Corpus*. In Proceedings of ACL/EACL, Toulouse, France, pages:50-57, 2001.

[2] R. Barzilay and L. Lee *Lee Bootstrapping Lexical Choice via Multi-Sequence Alignment* . In Proceedings of EMNLP. pages:164-171, 2002.

[3] R. Barzilay and L. Lee: Learning to Paraphrase: An Unsupervised Approach Using Multiple-Sequence Alignment. In Proceedings of Human Language Technology and North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003), Edmonton, Canada, 2003.

[4] R. Barzilay and N. Elhadad: Sentence Alignment for Monolingual Comparable Corpora. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2003), pages:25-33, Sapporo, Japan, 2003.

[5] G. Dias, S. Guilloré and J.G.P. Lopes: Extraction Automatique d'Associations Textuelles à Partir de Corpora Non Traités, In Proceedings of 5th International Conference on the Statistical Analysis of Textual Data, pages:213-221, 2000.

[6] W.B Dolan, C. Quirck and C. Brockett: Unsupervised Construction of Large Paraphrase Corpora: Exploiting Massively Parallel News Sources, In Proceedings of 20th International Conference on Computational Linguistics (COLING 2004), Geneva, Switzerland, 2004.

[7] V. Hatzivassiloglou, J.L. Klavans and E. Eskin: Detecting Text Similarity over Short Passages: Exploring Linguistic Feature Combinations via Machine Learning, In Proceedings of Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP 1999), University of Maryland, USA, 1999.

[8] H. Jing and K. McKeown: Cut and Paste based Text Summarization, In Proceedings of 1st Meeting of the North American Chapter of the Association for Computational Linguistics, pages:178-185, Seattle, USA, 2000.

[9] K. Knight and D. Marcu: Summarization Beyond Sentence Extraction: A Probabilistic Approach to Sentence Compression. Artificial Intelligence, 139(1):91-107, 2002.

[10] M. Le Nguyen, S. Horiguchi, A. Shimazu and B. Tu Ho: Example-based Sentence Reduction using the Hidden Markov Model, ACM Transactions on Asian Language Information Processing (TALIP), 3(2):146-158, 2004.

[11] V. Levenshtein: Binary Codes Capable of Correcting Deletions, Insertions, and Reversals., Soviet Physice-Doklady, 10:707-710, 1966.

[12] C.Y. Lin and E.H. Hovy: Automatic Evaluation of Summaries Using N-gram Co-Occurrence Statistics, In Proceedings of Human Language Technology and North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003), Edmonton, Canada, 2003.

[13] D. Lin and P. Pantel, DIRT - Discovery of Inference Rules from Text. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining* pp. 323-328, 2001.

[14] Marsi, E. and E. Krahmer: Explorations in Sentence Fusion, In Proceedings of the 10th European Workshop on Natural Language Generation, Aberdeen, Scotland, 2005.

[15] K. Papineni, S. Roukos, T. Ward, W.-J. Zhu: BLEU: a Method for Automatic Evaluation of Machine Translation, IBM Research Report RC22176, 2001.

[16] E. Polak: Computational Methods in Optimization, New York Academic Press, 1971.

[17] G. Salton and C. Buckley: Term Weighting Approaches in Automatic Text Retrieval, Information Processing and Management, 24(5):513-523, 1988.

[18] Y. Shinyama, S. Sekine, K. Sudo, and R. Grishman: Automatic Paraphrase Acquisition from News Articles, In Proceedings of Human Language Technology (HLT 2002), Sao Diego, USA, 2002.

[19] J. Sjöbergh and Kenji Araki: Extraction based summarization using a shortest path algorithm, In Proceedings of 12th Annual Language Processing Conference (NLP 2006), Yokohama, Japan, 2006.

[20] A. Stent, M. Marge and M. Singhai: Evaluating Evaluation Methods for Generation in the Presence of Variation, In Proceedings of Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2005), 2005.

[21] M. Yamamoto and Church, K.: Using Suffix Arrays to Compute Term Frequency and Document Frequency for all Substrings in a Corpus, Computational Linguistics, 27(1):1-30, 2001.

[6] By making a proportion statistical test for the accuracies: $H_0 : p_1 = p_2$ against $H_1 : p_1 > p_2$.