EED: Extended Edit Distance Measure for Machine Translation

Peter Stanchev **Weiyue Wang** **Hermann Ney**
Human Language Technology and Pattern Recognition, Computer Science Department
RWTH Aachen University, 52056 Aachen, Germany
<surname>@i6.informatik.rwth-aachen.de

Abstract

Over the years a number of machine translation metrics have been developed in order to evaluate the accuracy and quality of machine-generated translations. Metrics such as BLEU and TER have been used for decades. However, with the rapid progress of machine translation systems, the need for better metrics is growing. This paper proposes an extension of the edit distance, which achieves better human correlation, whilst remaining fast, flexible and easy to understand.

1 Introduction

Machine Translation (MT) has been a popular research topic for the past few years. It deals with the paradigm of how to automatically translate a sentence or a set of sentences from a source language to a different target language. In statistical MT, this can be formally described as finding the translation $e_1^I = e_1 \dots e_i \dots e_I$ with the highest probability for a given source language sentence $f_1^J = f_1 \dots f_j \dots f_J$:

$$\hat{e}_1^{\hat{I}} = \arg\max_{I, e_1^I} \{p(e_1^I \mid f_1^J)\} \quad (1)$$

This approach models the translation task by defining it as a search for the sentence that best suits a given criterion. For example through log-linear models as described by Och and Ney, 2002.

However, all approaches have to be evaluated to quantify the quality and accuracy of the produced translations. Naturally, the best method would be to have human experts rate each produced translation in order to evaluate the whole MT system. This is quite a costly process and is not viable for development of MT systems. For this reason a number of metrics exist that automate the process and use different scoring methods to automatically evaluate the produced translation based on a reference sentence. Two of the earliest and most popular metrics are BLEU (Papineni et al., 2002) and TER (Snover et al., 2006).

This paper introduces a new MT metric: Extended Edit Distance (EED), based on an extension of the Levenshtein distance (Levenshtein, 1966). This metric follows a number of criteria:

- It is bound between zero and one.
- Its definition is kept simple, as it does not depend on external dictionaries or language analysis.
- It has competitive human correlation.
- It is fast to compute.

The remainder of this paper is structured as follows: first, related work is reviewed in Section 2; Section 3 introduces the concept of edit distance and the different existing extensions of it; Section 4 introduces the EED metric in detail; A comparison with other metrics regarding human correlation and speed is performed in Section 5; Finally, a conclusion is drawn in Section 6.

2 Background

MT metrics compute a score based on the output of a MT system, here called "candidate", and a "reference" sentence, which is provided. The reference is a valid translation of the original source sentence to the target language, usually obtained through a human expert. A metric aims to use the pair of reference and candidate to give a numerical value to the correctness of the translation. A naïve approach would be to directly compare the candidate and reference in order to consider the translation quality. This, however, cannot be a good evaluation criterion since human language has multiple ways of expressing the same idea, and thus there is seldom one unique translation of a sentence from one language to another.

Over the years, a number of metrics have been created based on a variety of ideas and principles.

Proceedings of the Fourth Conference on Machine Translation (WMT), Volume 2: Shared Task Papers (Day 1) pages 514–520
Florence, Italy, August 1-2, 2019.

Some examples for such principles can be seen in the count-based metrics (BLEU, CHRF (Popovic, 2015)) or the edit distance based metrics (TER, CHARACTER (Wang et al., 2016), CDER (Leusch et al., 2006)).

Count-based metrics compute the n-grams of both reference and candidate and then compare them with each other using a scoring function. One of the most used metrics – BLEU, uses word level n-grams as input to a modified version of precision to evaluate the translation accuracy. Furthermore, a brevity penalty is applied if the candidate is shorter than the reference. CHRF uses the F-score to produce a scoring based on character level n-grams. In most cases, the shift from word level n-grams to the character level results in better human correlation (Popovic, 2015).

Edit distance based metrics utilise the edit distance to express the difference between the candidate and the reference. Since written language allows for the word order to be changed without significant change in meaning, the pure edit distance is too restrictive and is often extended by additional operations. TER extends it by introducing "shifts" which allow for words or phrases to be moved from one position in the candidate to another with a certain cost.

CDER gives another solution to the problem by introducing the operation of jumps. These "jumps" allow for a more flexible alignment. Of course, as in the n-gram based metrics, it is possible to apply these methods at both the word and the character level. CHARACTER uses the edit distance at the character level while keeping the shift operations at the word level with suitably adjusted costs.

3 Edit Distance

Since the metric presented in this paper belongs to the category of the edit distance based metrics, a more thorough introduction to the concept of edit distance is needed. The goal of the Levenshtein distance is to find the minimum number of operations required to transform the candidate into the reference. The Levenshtein distance in its purest form consists of three basic operations:

- Substitution: the act of switching one symbol with another
- Deletion: the removal of a symbol
- Insertion: the addition of a symbol

All of the basic operations are defined as having an uniform cost of one. To not penalise matching symbols with substitutions, substitutions can be defined via the Kroneker delta: $1 - \delta(c_n, r_m)$ with c_n and r_m standing for the symbol at position $m \in \{1, 2 \dots |r|\}, n \in \{1, 2 \dots |c|\}$ for the candidate c and reference r, respectively. The edit distance is then computed as the sum of substitution, insertion and deletion operations made.

The edit distance can be efficiently computed via the dynamic programming algorithm by Wagner and Fischer, 1974. This allows for a computation in $\mathcal{O}(cr)$.

In MT, the Levenshtein distance is not usually used in its original definition since it does not provide the required flexibility. The reason is that written language allows for multiple ways to express the same concept or idea. To alleviate this problem extensions to the edit distance have been proposed.

The most prominent extension of the edit distance, implemented by both TER and CHARACTER, is the introduction of an additional operation prior to computing the edit distance on the candidate. Namely, to permute the words in the candidate to most closely match the reference. This permutation is termed *shift*. Since computing all possible shifts of a given sentence is quite costly, in practice, the beam search algorithm is used to reduce the search space.

Another possible extension of the edit distance is to define so called *jumps*. Jumps provide the opportunity to continue the edit distance computation from a different point. A more detailed explanation of the jumps is presented in the next section.

To obtain a final score, the edit distance is normalised either over the length of the candidate or over the length of the reference. Naturally, in the case where every symbol is wrong and the normalising term is the shorter one of the candidate and the reference, the resulting score may significantly exceed 1.0. This in turn results in scores which are not easily interpretable.

4 Extended Edit Distance

One aspect of each metric is its input which usually comes in tokenized form. Punctuation marks are separated from words via a white space and abbreviation dots are kept next to the word e.g. "`e.g.`". EED additionally adds a white space at

both beginning and end of each sentence.

EED utilises the idea of jumps as an extension of the edit distance. EED operates at character level and is defined as follows:

$$\text{EED} = \min\left(\frac{(e + \alpha \cdot j) + \rho \cdot v}{|r| + \rho \cdot v}, 1\right) \quad (2)$$

where e is the sum of the edit operation with uniform cost of 1 for insertions and substitutions and 0.2 for deletions. j denotes the number of jumps performed with the corresponding control parameter $\alpha = 2.0$. v defines the number of characters that have been visited multiple times or not at all and scales over $\rho = 0.3$. The parameter values have been optimised based on the average correlation scores (both from and to English) from WMT17 and WMT18 (Bojar et al., 2017; Ma et al., 2018). EED is normalised over the length of the reference $|r|$ and the coverage penalty. To keep it within the [0,1] boundary, the minimum between 1 and the metric score is taken. This makes the metric more robust in cases of extreme discrepancy between candidate and reference length.

Jumps are a way to move between characters or blocks thereof and can be incorporated into the dynamic programming algorithm for the Levenshtein distance (Leusch et al., 2006). This provides an optimal solution for the matching between candidate and reference in reasonable computation time. In EED jumps may only be performed when a blank in the reference is reached, allowing the metric to take word boundaries into account and restricting the inter-word jumps. Figure 1 illustrates the way jumps work. Here `Die Fans` from the reference are aligned with `die Fans` from the candidate via a jump, after which normal edit distance operations are performed. When the `s` is reached, another jump is made to the blank before `n`, in order to align `nicht` to `Nicht`. Finally another jump is performed to align the period and white spaces. In total, this results in two edit operation errors (from the difference in capitalisation) and three jumps.

To further refine the metric a coverage penalty is introduced that aims to penalise characters which are aligned to more than once or not at all in the candidate. This allows the metric to penalise repetition of words in the reference with more than just the jump costs. The sum v of visits for all characters visited more than once is computed and is added, after multiplication with a scaling factor ρ to the total cost. To keep the situations where 1

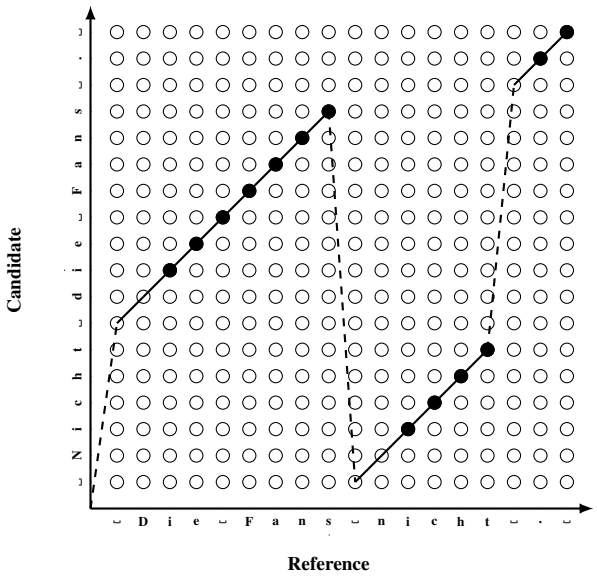

Figure 1: EED alignment lattice. Identity operations are marked with solid points, jumps with dashed lines, edit operations with full lines and blanks with ␣.

is chosen by the minimum in Equation (2) as few as possible, the coverage penalty is also used in the denominator.

Using only the length of the reference as part of the normalisation factor does not guarantee that the metric score is in the range [0,1]. This is undesirable since scores above one are not interpretable as an error measure. For this reason a number of strategies were considered to enforce this bound:

- Taking the maximum length between candidate and reference;
- Taking the average length between candidate and reference;
- Using just the candidate or just the reference;
- Cutting the score to 1.0 if it is above 1.0;
- Mapping the score to accuracy via the function $1/(1 - \text{EED})$ (Zhang et al., 2011).

Out of all of these methods, the simplest and most efficient method is to use the reference as normalisation and to cut the score if it is above one. In our experiments taking the maximum or average between candidate and reference leads to a decline in correlation. The use of accuracy mapping yields different results depending on the parameter setting of the metric and the test set used. For this reason EED uses the cut method for normalisation.

Although EED utilises the same movement technique as CDER, there are a few notable differences:

- Edit distance is performed on the character level;
- Jumps are performed only upon reaching a blank in the reference;
- An additional penalty for multiple matching of the same symbol (coverage cost) is applied

5 Results

EED is implemented in `C++` and imported in `python` via a wrapper. This implementation retains the ease of use of `python` while getting the speed from a `C++` implementation.

EED was evaluated via the scripts provided by Ma et al., 2018 as part of WMT18. The evaluation is done both on segment and system level. The data consists of about 3000 sentences per language pair as part of the `newstest2018` test set and provides one reference per translation. In total there are 14 language pairs. For the system level evaluation, direct assessment (DA) (Graham et al., 2017) was used to obtain human scores and Pearson's r is used as the correlation coefficient. The segment level uses the relative ranking (RR) which is pooled from system level DA scores. This results in DARR. The correlation coefficient used for the segment level is the Kendall's τ like formulation defined by Graham et al., 2015.

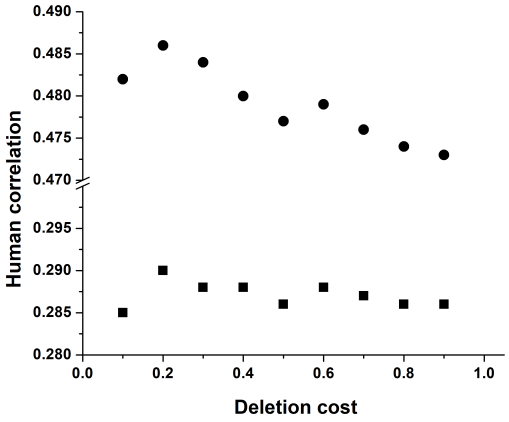

Figure 2: Human correlation variation as a function of deletion cost on WMT18 *to English* ■ and *from English* ● on segment-level.

To obtain the best possible human correlation, a parameter search was performed over ρ, α and the edit operation costs. For substitution and insertions there is no relevant correlation improvement. However, changes to the deletion cost parameter resulted in human correlation improvement. Using the WMT18 segment level test set, a parameter search was performed. Since searching over the whole search space is infeasible, the parameter search was done in a sequential manner. The results of the search are shown in Figure 2. From these results, combined with the findings on WMT16 and WMT17 (Bojar et al., 2016, 2017), the deletion cost is set to 0.2.

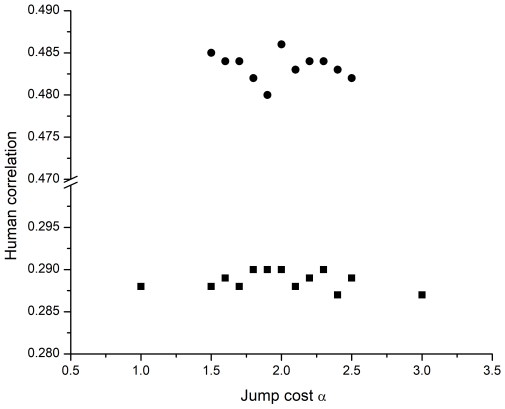

Figure 3: Human correlation variation as a function of jump cost on WMT18 *to English* ■ and *from English* ● on segment-level.

The error distribution of EED was skewed quite heavily towards performing jumps even after restricting jump operation only to blanks on the reference side. For this reason it was restricted further by increasing the jump costs. In order to determine the optimal jump penalty α, a parameter search was performed, which is presented Figure 3. It is evident that the optimal jump cost lie close to 2.0 for the *to English* direction. For the *from English* direction the optimum is clear, thus α is set to 2.0.

Similar to the deletion cost and the jump penalty, a parameter search was carried out for the coverage cost in order to increase human correlation. The results of the search are presented in Figure 4. The resulting optimum is $\rho = 0.3$.

After the parameter tuning, the performance of EED was measured by the human correlation achieved on the WMT18 test set. The results of this measurement obtained at the segment and system level and also in the directions *to English* and *from English* are presented in Tables 1 to 4. At the segment level, EED offers competitive results compared with the top-ranking metrics BEER, RUSE and CHRF +. On system level EED performs best for the *out of English* direction, fol-

	cs-en	de-en	et-en	fi-en	ru-en	zh-en	Average
# Sentences	5110	77811	56721	15648	10404	33357	33181
EED	0.297	0.486	0.335	0.227	0.284	**0.225**	0.309
BEER[1]	0.295	0.481	0.341	0.232	0.288	0.214	0.309
CHARACTER	0.256	0.450	0.286	0.185	0.244	0.202	0.271
CHRF +	0.288	0.479	0.332	0.234	0.279	0.207	0.303
ITER[2]	0.198	0.396	0.235	0.128	0.139	0.144	0.206
RUSE[3]	**0.347**	**0.498**	**0.368**	**0.273**	**0.311**	0.218	**0.336**
sentBLEU	0.233	0.415	0.285	0.154	0.228	0.178	0.248

Table 1: Segment-level human correlation measured through DARR to English on `newstest18` as part of WMT18 via absolute Kendall's τ.
[1] Stanojevic and Sima'an, 2014
[2] Panja and Naskar, 2018
[3] Shimanaka et al., 2018

	en-cs	en-de	en-et	en-fi	en-ru	en-zh	Average
# Sentences	5413	19711	32202	9809	22181	28602	19820
EED	0.508	0.674	0.572	0.503	**0.405**	**0.350**	**0.502**
BEER	**0.518**	**0.686**	0.558	0.511	0.403	0.302	0.496
CHARACTER	0.414	0.604	0.464	0.403	0.352	0.313	0.425
CHRF +	0.513	0.680	**0.573**	**0.525**	0.392	0.328	**0.502**
ITER	0.333	0.610	0.392	0.311	0.291	–	0.387
sentBLEU	0.389	0.620	0.414	0.355	0.330	0.311	0.403

Table 2: Segment-level human correlation measured through DARR from English on `newstest18` as part of WMT18 via absolute Kendall's τ.

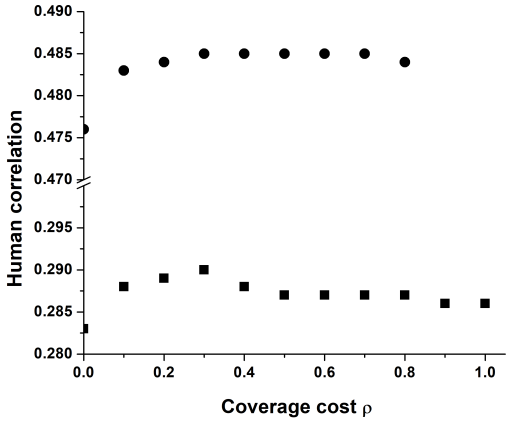

Figure 4: Human correlation variation as a function of coverage cost on WMT18 *to English* ■ and *from English* ● on segment-level.

lowed by CHARACTER and CDER. For the *to English* direction, EED is the second best after RUSE.

Apart from human correlation, EED was compared to the performance of the most common metrics. This measurement was performed by letting each metric evaluate 1M (10^6) sentence pairs and tracking the time and memory needed to complete the task. The following metrics have been tested: BEER, BLEU, CHARACTER, CHRF, EED. The results of the resource usage test are summarised in Table 5. The fastest is BLEU followed by EED. Concerning memory usage all metrics have similar memory needs, except for the shift based metrics which needed considerably more. Since CHARACTER needs more memory, candidate sentences above 200 words were restricted to 200 words for this test.

6 Conclusion

A number of different metrics have been developed over the years to help evaluate MT. Metrics such as BLEU and TER have been used for some time, but are surpassed by others both in terms of speed and human correlation.

EED as a metric provides a fast and reliable way to measure human correlation. It achieves competitive human correlation in comparison to the best metrics – BEER and CHRF and surpasses the most used metrics – BLEU and TER. Due to its simplicity and low resource usage it can be used to

	cs-en	de-en	et-en	fi-en	ru-en	zh-en	Average
# Systems	5	16	14	9	8	14	11
BEER	0.958	0.994	0.985	0.991	0.982	0.976	0.981
BLEU	0.970	0.971	0.986	0.973	0.979	0.978	0.976
CDER	0.972	0.980	**0.990**	0.984	0.980	**0.982**	0.981
CHARACTER	0.970	0.993	0.979	0.989	0.991	0.950	0.979
CHRF +	0.966	0.993	0.981	0.989	0.990	0.964	0.981
EED	0.970	0.994	0.984	0.991	**0.993**	0.974	0.984
ITER	0.975	0.990	0.975	**0.996**	0.937	0.980	0.976
NIST[1]	0.954	0.984	0.983	0.975	0.973	0.968	0.973
RUSE	**0.981**	**0.997**	**0.990**	0.991	0.988	0.981	**0.988**
TER	0.950	0.970	**0.990**	0.968	0.970	0.975	0.971

Table 3: System-level human correlation as DA to English on `newstest18` as part of WMT18 via absolute Pearson's r.
[1] Doddington, 2002

	en-cs	en-de	en-et	en-fi	en-ru	en-zh	Average
# Systems	5	16	14	12	9	14	12
BEER	0.992	**0.991**	0.980	0.961	0.988	0.928	0.973
BLEU	0.995	0.981	0.975	0.962	0.983	0.947	0.973
CDER	0.997	0.986	**0.984**	0.964	0.984	0.961	0.979
CHARACTER	0.993	0.989	0.956	0.974	0.983	**0.983**	0.980
CHRF +	0.990	0.989	0.982	0.970	0.989	0.943	0.977
EED	0.988	0.990	0.983	**0.977**	**0.990**	0.955	**0.981**
ITER	0.915	0.984	0.981	0.973	0.975	–	0.966
NIST	**0.999**	0.986	0.983	0.949	**0.990**	0.950	0.976
TER	0.997	0.988	0.981	0.942	0.987	0.963	0.976

Table 4: System-level human correlation as DA from English on `newstest18` as part of WMT18 via absolute Pearson's r.

Metric	**EED**	**BEER**	**CHRF ++**	**CHARACTER**	**BLEU**	**TER**
Sentences/s	969.9	621.5	261.7	9.5	6410.2	316.6
Memory	1.3G	1.1G	0.3G	48.4G	0.3G	8.4G

Table 5: Speed and memory comparison between metrics, as sentences per second and memory in gigabyte. Measured on 1M sentences.

quickly evaluate a MT system's output during development.

Since there are a number of metrics based on some extensions of the Levenshtein distance, a more in-depth analysis of the field is required. Furthermore, the relationship between shifts and jumps will be investigated in the future.

Acknowledgments

This work has received funding from the European Research Council (ERC) (under the European Union's Horizon 2020 research and innovation programme, grant agreement No 694537, project "SEQCLAS") and the Deutsche Forschungsgemeinschaft (DFG; grant agreement NE 572/8-1, project "CoreTec"). The GPU cluster used for the experiments was partially funded by DFG Grant INST 222/1168-1. The work reflects only the authors' views and none of the funding agencies is responsible for any use that may be made of the information it contains.

References

Ondřej Bojar, Yvette Graham, and Amir Kamran. 2017. Results of the wmt17 metrics shared task. In *Proceedings of the Second Conference on Machine Translation*, pages 489–513.

Ondřej Bojar, Yvette Graham, Amir Kamran, and Miloš Stanojević. 2016. Results of the wmt16 metrics shared task. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, volume 2, pages 199–231.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*, pages 138–145. Morgan Kaufmann Publishers Inc.

Yvette Graham, Timothy Baldwin, and Nitika Mathur. 2015. Accurate evaluation of segment-level machine translation metrics. In *HLT-NAACL*, pages 1183–1191.

Yvette Graham, Timothy Baldwin, Alistair Moffat, and Justin Zobel. 2017. Can machine translation systems be evaluated by the crowd alone. *Natural Language Engineering*, 23(1):3–30.

Gregor Leusch, Nicola Ueffing, and Hermann Ney. 2006. Cder: Efficient mt evaluation using block movements. In *11th Conference of the European Chapter of the Association for Computational Linguistics*.

Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

Qingsong Ma, Ondřej Bojar, and Yvette Graham. 2018. Results of the wmt18 metrics shared task: Both characters and embeddings achieve good performance. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 671–688.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 295–302. Association for Computational Linguistics.

Joybrata Panja and Sudip Kumar Naskar. 2018. Iter: Improving translation edit rate through optimizable edit costs. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 759–763, Belgium, Brussels. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine targeted translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Maja Popovic. 2015. CHRf: character n-gram f-score for automatic MT evaluation. In *WMT@ EMNLP*, pages 392–395.

Hiroki Shimanaka, Tomoyuki Kajiwara, and Mamoru Komachi. 2018. Ruse: Regressor using sentence embeddings for automatic machine translation evaluation. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 751–758.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, volume 200.

Milos Stanojevic and Khalil Sima'an. 2014. BEER: Better evaluation as ranking. In *WMT@ ACL*, pages 414–419.

Robert A Wagner and Michael J Fischer. 1974. The string-to-string correction problem. *Journal of the ACM (JACM)*, 21(1):168–173.

Weiyue Wang, Jan-Thorsten Peter, Hendrik Rosendahl, and Hermann Ney. 2016. CharacTER: Translation edit rate on character level. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, volume 2, pages 505–510.

Junsheng Zhang, Yunchuan Sun, Huilin Wang, and Yanqing He. 2011. Calculating statistical similarity between sentences. *Journal of Convergence Information Technology*, 6(2).