Automated Paraphrase Lattice Creation for HyTER Machine Translation Evaluation

Marianna Apidianaki∗†, Guillaume Wisniewski†, Anne Cocos∗ and Chris Callison-Burch∗

† LIMSI, CNRS, Univ. Paris Sud, Université Paris-Saclay, 91403 Orsay
∗ Computer and Information Science Department, University of Pennsylvania

{marapi,acocos,ccb}@seas.upenn.edu, guillaume.wisniewski@limsi.fr

Abstract

We propose a variant of a well-known machine translation (MT) evaluation metric, HyTER (Dreyer and Marcu, 2012), which exploits reference translations enriched with meaning equivalent expressions. The original HyTER metric relied on hand-crafted paraphrase networks which restricted its applicability to new data. We test, for the first time, HyTER with automatically built paraphrase lattices. We show that although the metric obtains good results on small and carefully curated data with both manually and automatically selected substitutes, it achieves medium performance on much larger and noisier datasets, demonstrating the limits of the metric for tuning and evaluation of current MT systems.

1 Introduction

Human translators and MT systems can produce multiple plausible translations for input texts. To reward meaning-equivalent but lexically divergent translations, MT evaluation metrics exploit synonyms and paraphrases, or multiple references (Papineni et al., 2002; Doddington, 2002; Denkowski and Lavie, 2010; Lo et al., 2012). The HyTER metric (Dreyer and Marcu, 2012) relies on massive reference networks encoding an exponential number of correct translations for parts of a given sentence, proposed by human annotators. The manually built networks attempt to encode the set of all correct translations for a sentence, and HyTER rewards high quality hypotheses by measuring their minimum edit distance to the set of possible translations.

HyTER spurred a lot of enthusiasm but the need for human annotations heavily reduced its applicability to new data. We propose to use an embedding-based lexical substitution model (Melamud et al., 2015) for building this type of reference networks and test, for the first time, the metric with automatically generated lattices (hereafter HyTERA). We show that HyTERA strongly correlates with HyTER with hand-crafted lattices, and approximates the hTER score (Snover et al., 2006) as measured using post-edits made by human annotators. Furthermore, we generate lattices for standard datasets from a recent WMT Metrics Shared Task and perform the first evaluation of HyTER on large and noisier datasets. The results show that it still remains an interesting solution for MT evaluation, but highlight its limits when used to evaluate recent MT systems that make far less errors of lexical choice than older systems.

2 The Original HyTER Metric

The HyTER metric (Dreyer and Marcu, 2012) computes the similarity between a translation hypothesis and a reference lattice that compactly encodes millions of meaning-equivalent translations. Formally HyTER is defined as:

\[
\text{HyTER}(x, \mathcal{Y}) = \min_{y \in \mathcal{Y}} \frac{\text{LS}(x, y)}{\text{len}(y)}
\]

where \(\mathcal{Y}\) is a set of references that can be encoded as a finite state automaton such as the one represented in Figure 1, \(x\) is a translation hypothesis and LS is the standard Levenshtein distance, defined as the minimum number of substitutions, deletions and insertions required to transform \(x\) into \(y\). We use, in all our experiments, our own im-
A window width of 1). The embeddings $c$ are context embeddings generated by skip-gram.\footnote{In the original implementation, Melamud et al. (2015) use syntactic dependencies as contexts. We define context as a fixed-width window of words to avoid the need for dependency parsing.} In our implementation, we train 300-dimensional word and context embeddings over the 4B words in the Annotated Gigaword (AGiga) corpus (Napoles et al., 2012) using the gensim \texttt{word2vec} package (Mikolov et al., 2013b,a; Řehůrek and Sojka, 2010).\footnote{The \texttt{word2vec} training parameters we use are a context window of size 3, learning rate alpha from 0.025 to 0.0001, minimum word count 100, sampling parameter \texttt{e} \footnote{Our implementation of AddCos with PPDB substitutes can be found at \url{https://github.com/acocos/lexsub_addcos}.} $= 4$, 10 negative samples per target word, and 5 training epochs.}

Each content word token in a sentence is expanded to include all its possible substitutes selected by AddCos in this specific context, and the lattice can take any path from the expanded start token to the expanded end token. We filter the paraphrase candidates according to: a) their PPDB2.0 score, an out-of-context measure of paraphrase confidence which denotes the strength of the relation between the paraphrase and the target word (hereafter, PPDBSc) (Pavlick et al., 2015); b) the substitution score assigned to paraphrases in context by the AddCos model (hereafter, AddCosSc), which shows whether the paraphrase is a good fit for the target word in a specific context.\footnote{The context $C$ is set of context embeddings for words appearing within a fixed-width window of the target $t$ in a sentence (we use a window width of 1). The embeddings $c$ are context embeddings generated by skip-gram.\footnote{In the original implementation, Melamud et al. (2015) use syntactic dependencies as contexts. We define context as a fixed-width window of words to avoid the need for dependency parsing.} In our implementation, we train 300-dimensional word and context embeddings over the 4B words in the Annotated Gigaword (AGiga) corpus (Napoles et al., 2012) using the gensim \texttt{word2vec} package (Mikolov et al., 2013b,a; Řehůrek and Sojka, 2010). Each content word token in a sentence is expanded to include all its possible substitutes selected by AddCos in this specific context, and the lattice can take any path from the expanded start token to the expanded end token. We filter the paraphrase candidates according to: a) their PPDB2.0 score, an out-of-context measure of paraphrase confidence which denotes the strength of the relation between the paraphrase and the target word (hereafter, PPDBSc) (Pavlick et al., 2015); b) the substitution score assigned to paraphrases in context by the AddCos model (hereafter, AddCosSc), which shows whether the paraphrase is a good fit for the target word in a specific context.\footnote{The \texttt{word2vec} training parameters we use are a context window of size 3, learning rate alpha from 0.025 to 0.0001, minimum word count 100, sampling parameter \texttt{e} = 4, 10 negative samples per target word, and 5 training epochs.}

We assess the quality of HYTERA to evaluate the quality of MT output both at the sentence and the system level. We first use the setting of Dreyer and Marcu (2012), in Section § 5.1, to compare the score estimated by HYTER and HYTERA to HTER scores. In Section § 5.2, we explore whether HYTERA can reliably predict human translation quality scores from the WMT16 Metrics Shared Task.
5 Comparing HyTER and HyTERA

5.1 Open MT NIST Evaluation

To evaluate the performance of HyTER, Dreyer and Marcu (2012) examine whether it can approximate the hTER score (Snover et al., 2006) that measures the number of edits required to change a system output into its post-edition. hTER scores are a good estimate of translation quality and usefulness, but require each translation hypothesis to be corrected by a human annotator. Dreyer and Marcu (2012) show that it can be closely approximated by HyTER scores. In this section, we reproduce their experiments with HyTERA to see whether it is possible to use automatically-built rather than hand-crafted references to approximate hTER scores.

Data Following Dreyer and Marcu (2012), we consider a subset of the ‘progress set’ used in the 2010 Open MT NIST evaluation. This corpus is made of 102 Arabic and 102 Chinese sentences. Each sentence is automatically translated into English by five MT systems and these translation hypotheses are post-edited by experienced LDC annotators, allowing us to compute hTER scores. The corpus also contains four references and meaning-equivalent annotations which allow for direct comparison to the original HyTER.

Experimental Setting We build meaning-equivalent lattices by applying the lexical substitution method described in Section 3 to each of the four references associated with a sentence, and considering the union of the resulting lattices. We report results for two kinds of lattices: lattices encoding all lexical substitutes available for a word in PPDB (allPars) and lattices of substitutes with PPDBSc≥2.3 (allParsFiltered) and AddCosSc≥0. As expected, the allPars lattices are much larger than the manual and the filtered lattices (cf. Table 1). In all our experiments, all corpora are down-cased and tokenized using standard Moses scripts. hTER scores are computed using TERp (Snover et al., 2009).

Sentence Level Evaluation Table 2 reports the correlation between HyTER, HyTERA and hTER at the sentence level. We also include as a baseline the correlation with the sentence-level BLEU score, estimated by the arithmetic mean of 1 to 4-gram precisions.

<table>
<thead>
<tr>
<th></th>
<th>ar2en</th>
<th>zh2en</th>
</tr>
</thead>
<tbody>
<tr>
<td>manual</td>
<td>9,454,542</td>
<td>7.8 × 10^9</td>
</tr>
<tr>
<td>allPars</td>
<td>1.8 × 10^27</td>
<td>8.5 × 10^27</td>
</tr>
<tr>
<td>allParsFiltered</td>
<td>10,803</td>
<td>3.3 × 10^30</td>
</tr>
</tbody>
</table>

Table 1: Median number of references generated by each method.

In all cases, there is a high correlation between HyTER, HyTERA and hTER, significantly higher than the correlation between BLEU and hTER. This observation shows that replacing the hand-crafted lattices with automatically built ones has only a moderate impact on the HyTER metric quality: automatic lattices result in a small drop of the correlation when evaluating hypotheses translated from Chinese, and slightly improve it for the Arabic to English condition. Overall HyTERA scores are highly correlated with hTER scores ($\rho = 0.766$ for Arabic and $\rho = 0.756$ for Chinese). More importantly, considering the filtered lattices allows to significantly reduce computation time compared to the allPars ones without hurting the quality estimation capacity of the metric.

System Level Evaluation Figure 2 shows how the five MT systems are ranked by the different metrics we consider, when translating from Arabic to English. All metrics rank the systems in the same order, except from HyTER with allParsFiltered that only inverts two systems. Note that the tested systems were selected by NIST to cover a variety of system architectures (statistical, rule-based, hybrid) and performances (Dreyer and Marcu, 2012), which makes distinction between them an easy task for all metrics. The benefits of using a metric like HyTER, which focuses on the word level, are much clearer in the sentence-based evaluation (Table 2).

5.2 WMT Metrics Evaluation

In our second set of experiments, we explore the ability of HyTERA to predict direct human judgments at the sentence level using the setting of the WMT16 Metrics Shared Task (Bojar et al., 2016). We measure the correlation between ad-
Table 2: Correlation, calculated by Spearman’s ρ, between the scores of translation hypotheses estimated by HyTER and hTER.

<table>
<thead>
<tr>
<th></th>
<th>ar → en</th>
<th>zh → en</th>
</tr>
</thead>
<tbody>
<tr>
<td>HyTER</td>
<td>0.667</td>
<td>0.672</td>
</tr>
<tr>
<td>HyTER(allPars)</td>
<td>0.673</td>
<td>0.616</td>
</tr>
<tr>
<td>HyTER(allParsFiltered)</td>
<td>0.671</td>
<td>0.605</td>
</tr>
<tr>
<td>BLEU</td>
<td>-0.563</td>
<td>-0.556</td>
</tr>
</tbody>
</table>

Figure 2: Five Arabic to English MT systems scored by different metrics.

equency scores collected on Amazon Mechanical Turk following the method advocated by Graham et al. (2016) and the translation quality estimated by applying HyTERA to the official WMT reference. Table 3 reports the results achieved by HyTERA on the six language pairs of the WMT16 Shared Task and its rank among the other metrics tested in the competition.

HyTERA obtains medium performance on the WMT16 dataset, which is much larger and noisier than the dataset used for evaluation in (Dreyer and Marcu, 2012): it is made, for each language, of 560 translations sampled from outputs of all systems taking part in the WMT15 campaign. It is important to note that the HyTER scores used in the initial HyTER evaluation were produced by experienced LDC annotators, while the WMT16 Direct Assessment (DA) adequacy judgments were collected from non-experts through crowd-sourcing (Bojar et al., 2016). HyTERA achieves higher performance than the sentBLEU baseline in four language pairs (cs/de/ru/tr-en). It obtains slightly lower correlation than sentBLEU for fi-en and ro-en, the language pairs in which correlation was lower for all metrics.

Among the metrics tested at the WMT16 shared task we find combination metrics, and metrics that have been tuned on a development dataset. The metric that performs best for most languages in the segment-level WMT16 evaluation, DPMFCOMB, combines 57 individual metrics (Yu et al., 2015). Similarly, the second highest ranked metric, METRI-CSF, combines BLEU, METEOR, the alignment-based metric UPF-COMBALT (Fomicheva et al., 2016), and fluency features. The BEER metric, found in fifth position, is a trained evaluation metric with a linear model that combines features capturing character n-grams and permutation trees (Stanojević and Sima’an, 2015).

We report the rank of HyTERA among all metrics (single and combined), and among the single ones. It is important to note that HyTERA needs no tuning, is straightforward to use and very fast to compute, especially with filtered lattices (on average 6s).

The lower performance of the metric on this dataset is also due to the different nature of the MT systems tested. While in the (Dreyer and Marcu, 2012) evaluation, the systems came from the 2010 Open MT NIST evaluation and were selected to cover a variety of architectures and performances, the systems that participated in WMT15 are, for the large part, neural MT systems (Bojar et al., 2015). As reported by Bentivogli et al. (2016), Neural MT systems make at least 17% fewer lexical choice errors than phrase-based systems, which limits the potential of HyTERA, primarily focused on capturing correct lexical choice.

6 Conclusion

We have proposed a method for automatic paraphrase lattice creation which makes the HyTER metric applicable to new datasets. We provide the first evaluation of HyTER on data from a recent
<table>
<thead>
<tr>
<th></th>
<th>HYTERA</th>
<th>SENTBLEU</th>
<th>WMT All</th>
<th>WMT Single Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>r</td>
<td>r</td>
<td>best</td>
<td>rank/15</td>
</tr>
<tr>
<td>cs → en</td>
<td>.599</td>
<td>.557</td>
<td>.713</td>
<td>10</td>
</tr>
<tr>
<td>de → en</td>
<td>.498</td>
<td>.448</td>
<td>.601</td>
<td>6</td>
</tr>
<tr>
<td>fi → en</td>
<td>.476</td>
<td>.484</td>
<td>.598</td>
<td>7</td>
</tr>
<tr>
<td>ro → en</td>
<td>.483</td>
<td>.499</td>
<td>.662</td>
<td>9</td>
</tr>
<tr>
<td>ru → en</td>
<td>.525</td>
<td>.502</td>
<td>.618</td>
<td>10</td>
</tr>
<tr>
<td>tr → en</td>
<td>.540</td>
<td>.532</td>
<td>.663</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 3: Pearson correlation between HYTERA and human judgments at the segment level on WMT16 Metrics Shared Task data on different language pairs. We compare to the scores of the SENTBLEU baseline. We report the best correlation achieved by the participating metrics and the rank of HYTERA among all 15 participants, and among the single 13 metrics left after excluding combined ones.

WMT Metrics Shared task. We show that although the metric achieves high correlation with human judgments of translation quality on small and carefully curated data, with both manual and automatically constructed paraphrase networks, it obtains medium performance on recent WMT data. The lower performance is mainly due to the noisier nature of the data and to the higher quality lexical choices made by current neural MT systems, compared to phrase-based and transfer systems, which limits the potential of the metric for system evaluation and tuning.

In its current form, the paraphrase substitution mechanism supports only lexical substitutions. It would be straightforward to extend the AddCos method to handle multi-word paraphrases by training embeddings for multi-word phrases, keeping in mind that longer substitutions might require restructuring the produced sentences to preserve grammaticality.

7 Acknowledgments

We would like to thank Markus Dreyer for sharing with us the original HYTER code.

This work has been supported by the French National Research Agency under project ANR-16-CE33-0013. This material is based in part on research sponsored by DARPA under grant number FA8750-13-2-0017 (the DEFT program) and HR0011-15-C-0115 (the LORELEI program). The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes. The views and conclusions contained in this publication are those of the authors and should not be interpreted as representing official policies or endorsements of DARPA and the U.S. Government.

References


