SPEDE: Probabilistic Edit Distance Metrics for MT Evaluation

Mengqiu Wang and **Christopher D. Manning**
Computer Science Department
Stanford University
Stanford, CA 94305 USA
{mengqiu,manning}@cs.stanford.edu

Abstract

This paper describes Stanford University's submission to the Shared Evaluation Task of WMT 2012. Our proposed metric (SPEDE) computes probabilistic edit distance as predictions of translation quality. We learn weighted edit distance in a probabilistic finite state machine (pFSM) model, where state transitions correspond to edit operations. While standard edit distance models cannot capture long-distance word swapping or cross alignments, we rectify these shortcomings using a novel pushdown automaton extension of the pFSM model. Our models are trained in a regression framework, and can easily incorporate a rich set of linguistic features. Evaluated on two different prediction tasks across a diverse set of datasets, our methods achieve state-of-the-art correlation with human judgments.

1 Introduction

We describe the Stanford Probabilistic Edit Distance Evaluation (SPEDE) metric, which makes predictions of translation quality by computing weighted edit distance. We model weighted edit distance in a probabilistic finite state machine (pFSM), where state transitions correspond to edit operations. The weights of the edit operations are then automatically learned in a regression framework. One of the major contributions of this paper is a novel extension of the pFSM model into a probabilistic Pushdown Automaton (pPDA), which enhances traditional edit-distance models with the ability to model phrase shift and word swapping. Furthermore, we give a new log-linear parameterization to the pFSM model, which allows it to easily incorporate rich linguistic features.

We conducted extensive experiments on a diverse set of standard evaluation data sets (NIST OpenMT06, 08; WMT06, 07, 08). Our models achieve or surpass state-of-the-art results on all test sets.

2 Related Work

Research in automatic machine translation (MT) evaluation metrics has been a key driving force behind the recent advances of statistical machine translation (SMT) systems. The early seminal work on automatic MT metrics (e.g., BLEU and NIST) is largely based on *n*-gram matches (Papineni et al., 2002; Doddington, 2002). Despite their simplicity, these measures have shown good correlation with human judgments, and enabled large-scale evaluations across many different MT systems, without incurring the huge labor cost of human evaluation (Callison-Burch et al. (2009; 2010; 2011), *inter alia*).

Later metrics that move beyond *n*-grams achieve higher accuracy and improved robustness from resources like WordNet synonyms (Miller et al., 1990), paraphrasing (Zhou et al., 2006; Snover et al., 2009; Denkowski and Lavie, 2010), and syntactic parse structures (Liu et al., 2005; Owczarzak et al., 2008; He et al., 2010). But a common problem in these metrics is they typically resort to ad-hoc tuning methods instead of principled approaches to incorporate linguistic features. Recent models use linear or SVM regression and train them against human judgments to automatic learn feature weights, and have shown state-of-the-art correlation with human judgments (Albrecht and Hwa, 2007a; Albrecht and Hwa, 2007b; Sun et al., 2008; Pado et al., 2009). The drawback, however, is they rely on time-consuming

Proceedings of the 7th Workshop on Statistical Machine Translation, pages 76–83,
Montréal, Canada, June 7-8, 2012. ©2012 Association for Computational Linguistics

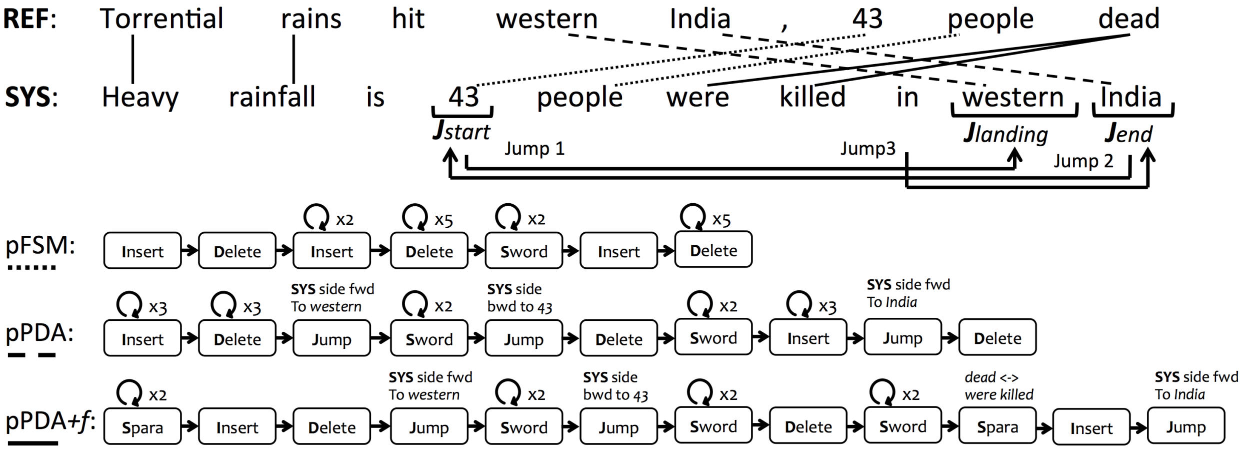

Figure 1: This diagram illustrates an example translation pair in the Chinese-English portion of OpenMT08 data set (Doc:AFP_CMN_20070703.0005, system09, sent 1). The three rows below are the best state transition (edit) sequences that transforms REF to SYS, according to the three proposed models. The corresponding alignments generated by the models (pFSM, pPDA, pPDA+*f*) are shown with different styled lines, with later models in the order generating strictly more alignments than earlier ones. The gold human evaluation score is 6.5, and model predictions are: pPDA+f 5.5, pPDA 4.3, pFSM 3.1, METEORR 3.2, TERR 2.8.

preprocessing modules to extract linguistic features (e.g., a full end-to-end textual entailment system was needed in Pado et al. (2009)), which severely limits their practical use. Furthermore, these models employ a large number of features (on the order of hundreds), and consequently make the model predictions opaque and hard to analyze.

3 pFSMs for MT Regression

We start off by framing the problem of machine translation evaluation in terms of weighted edit distance calculated using probabilistic finite state machines (pFSMs). A FSM defines a language by accepting a string of input tokens in the language, and rejecting those that are not. A probabilistic FSM defines the probability that a string is in a language, extending on the concept of a FSM. Commonly used models such as HMMs, *n*-gram models, Markov Chains, probabilistic finite state transducers and PCFGs all fall in the broad family of pFSMs (Knight and Al-Onaizan, 1998; Eisner, 2002; Kumar and Byrne, 2003; Vidal et al., 2005). Unlike all the other applications of FSMs where tokens in the language are words, in our language tokens are edit operations. A string of tokens that our FSM accepts is an edit sequence that transforms a reference translation (denoted as *ref*) into a system translation (*sys*).

Our pFSM has a unique start and stop state, and one state per edit operation (i.e., *Insert*, *Delete*, *Substitution*). The probability of an edit sequence **e** is generated by the model is the product of the state transition probabilities in the pFSM, formally described as:

$$w(\mathbf{e} \mid \mathbf{s}, \mathbf{r}) = \frac{1}{Z} \prod_{i=1}^{|\mathbf{e}|} \exp \theta \cdot \mathbf{f}(e_{i-1}, e_i, \mathbf{s}, \mathbf{r}) \qquad (1)$$

We featurize each of the state changes with a log-linear parameterization; **f** is a set of binary feature functions defined over pairs of neighboring states (by the Markov assumption) and the input sentences, and θ are the associated feature weights; *r* and *s* are shorthand for *ref* and *sys*; *Z* is a partition function. In this basic pFSM model, the feature functions are simply identity functions that emit the current state, and the state transition sequence of the previous state and the current state.

The feature weights are then automatically learned by training a global regression model where some translational equivalence judgment score (e.g., human assessment score, or HTER (Snover et al., 2006)) for each *sys* and *ref* translation pair is the regression target ($\hat{y}$). Since the "gold" edit sequence are not given at training or prediction time, we treat the edit sequences as hidden variables and sum over them in our model. We introduce a new regression variable $y \in \mathbb{R}$ which is the log-sum of the unnormalized weights (Eqn. (1)) of all edit sequences, formally expressed as:

$$y = \log \sum_{\mathbf{e}' \subseteq \mathbf{e}^*} \prod_{i=1}^{|\mathbf{e}'|} \exp \theta \cdot \mathbf{f}(e_{i-1}, e_i, \mathbf{s}, \mathbf{r}) \qquad (2)$$

The sum over an exponential number of edit sequences in $\mathbf{e}^*$ is solved efficiently using a forward-backward style dynamic program. Any edit sequence that does not lead to a complete transformation of the translation pair has a probability of zero in our model. Our regression target then seeks to minimize the least squares error with respect to $\hat{y}$, plus a $L2$-norm regularizer term parameterized by λ:

$$\theta^* = \min_{\theta} \{ \sum_{\mathbf{s_i}, \mathbf{r_i}} [\hat{y}_i - (\frac{y}{|\mathbf{s_i}| + |\mathbf{r_i}|} + \alpha)]^2 + \lambda \|\theta\|^2 \} \qquad (3)$$

The $|\mathbf{s_i}| + |\mathbf{r_i}|$ is a length normalization term for the ith training instance, and α is a scaling constant for adjusting to different scoring standards (e.g., 7-point scale vs. 5-point scale), whose value is automatically learned. At test time, $y/(|\mathbf{s}| + |\mathbf{r}|) + \alpha$ is computed as the predicted score.

We replaced the standard substitution edit operation with three new operations: S_{word} for same word substitution, S_{lemma} for same lemma substitution, and S_{punc} for same punctuation substitution. In other words, all but the three matching-based substitutions are disallowed. The start state can transition into any of the edit states with a constant unit cost, and each edit state can transition into any other edit state if and only if the edit operation involved is valid at the current edit position (e.g., the model cannot transition into *Delete* state if it is already at the end of *ref*; similarly it cannot transition into S_{lemma} unless the lemma of the two words under edit in *sys* and *ref* match). When the end of both sentences are reached, the model transitions into the stop state and ends the edit sequence. The first row in Figure 1 starting with pFSM shows a state transition sequence for an example *sys*/*ref* translation pair. There exists a one-to-one correspondence between substitution edits and word alignments. Therefore this example state transition sequence correctly generates an alignment for the word *43* and *people*.

It is helpful to compare with the TER metric (Snover et al., 2006), which is based on the idea of word error rate measured in edit distance, to better understand the intuition behind our model. There are two major improvements in our model: 1) the edit operations in our model are weighted, as defined by the feature functions and weights; 2) the weights are automatically learned, instead of being uniform or manually set; and 3) we model state transitions, which can be understood as a bigram extension of the unigram edit distance model used in TER. For example, if in our learned model the feature for two consecutive S_{word} states has a positive weight, then our model would favor consecutive same word substitutions, whereas in the TER model the order of the substitution does not matter. The extended TER-plus (Snover et al., 2009) metric addresses the first problem but not the other two.

3.1 pPDA Extension

A shortcoming of edit distance models is that they cannot handle long-distance word swapping — a pervasive phenomenon found in most natural languages. [1] Edit operations in standard edit distance models need to obey strict incremental order in their edit position, in order to admit efficient dynamic programming solutions. The same limitation is shared by our pFSM model, where the Markov assumption is made based on the incremental order of edit positions. Although there is no known solution to the general problem of computing edit distance where long-distance swapping is permitted (Dombb et al., 2010), approximate algorithms do exist. We present a simple but novel extension of the pFSM model to a probabilistic pushdown automaton (pPDA), to capture non-nested word swapping within limited distance, which covers a majority of word swapping in observed in real data (Wu, 2010).

A pPDA, in its simplest form, is a pFSM where each control state is equipped with a stack (Esparza and Kucera, 2005). The addition of stacks for each transition state endows the machine with memory, extending its expressiveness beyond that of context-free formalisms. By construction, at any stage in a normal edit sequence, the pPDA model can "jump"

[1] The edit distance algorithm described in Cormen et al. (2001) can only handle adjacent word swapping (transposition), but not long-distance swapping.

forward within a fixed distance (controlled by a max distance parameter) to a new edit position on either side of the sentence pair, and start a new edit subsequence from there. Assuming the jump was made on the *sys* side, [2] the machine remembers its current edit position in *sys* as J_{start}, and the destination position on *sys* after the jump as $J_{landing}$.

We constrain our model so that the only edit operations that are allowed immediately following a "jump" are from the set of substitution operations (e.g., S_{word}). And after at least one substitution has been made, the device can now "jump" back to J_{start}, remembering the current edit position as J_{end}. Another constraint here is that after the backward "jump", all edit operations are permitted except for *Delete*, which cannot take place until at least one substitution has been made. When the edit sequence advances to position $J_{landing}$, the only operation allowed at that point is another "jump" forward operation to position J_{end}, at which point we also clear all memory about jump positions and reset.

An intuitive explanation is that when pPDA makes the first forward jump, a gap is left in *sys* that has not been edited yet. It remembers where it left off, and comes back to it after some substitutions have been made to complete the edit sequence. The second row in Figure 1 (starting with pPDA) illustrates an edit sequence in a pPDA model that involves three "jump" operations, which are annotated and indexed by number 1-3 in the example. "Jump 1" creates an un-edited gap between word *43* and *western*, after two substitutions, the model makes "jump 2" to go back and edit the gap. The only edit permitted immediately after "jump 2" is deleting the comma in *ref*, since inserting the word *43* in *sys* before any substitution is disallowed. Once the gap is completed, the model resumes at position J_{end} by making "jump 3", and completes the jump sequence.

The "jumps" allowed the model to align words such as *western India*, in addition to the alignments of *43 people* found by the pFSM. In practice, we found that our extension gives a big boost to model performance (*cf.* Section 5.1), with only a modest increase in computation time. [3]

3.2 Parameter Estimation

Since the least squares operator preserves convexity, and the inner log-sum-exponential function is convex, the resulting objective function is also convex. For parameter learning, we used the limited memory quasi-newton method (Liu and Nocedal, 1989) to find the optimal feature weights and scaling constant for the objective. We initialized $\theta = \vec{0}$, $\alpha = 0$, and $\lambda = 5$. We also threw away features occurring fewer than five times in training corpus. Gradient calculation was similar to other pFSM models, such as HMMs, we omitted the details here, for brevity.

4 Rich Linguistic Features

We add new substitution operations beyond those introduced in Section 3, to capture synonyms and paraphrase in the translations. Synonym relations are defined according to WordNet (Miller et al., 1990), and paraphrase matches are given by a lookup table used in TERplus (Snover et al., 2009). To better take advantage of paraphrase information at the multi-word phrase level, we extended our substitution operations to match longer phrases by adding one-to-many and many-to-many bigram block substitutions.

5 Experiments

The goal of our experiments is to test both the accuracy and robustness of the proposed new models. We then show that modeling word swapping and rich linguistics features further improve our results.

To better situate our work among past research and to draw meaningful comparison, we use exactly the same standard evaluation data sets and metrics as Pado et al. (2009), which is currently the state-of-the-art result for regression-based MT evaluation. We consider four widely used MT metrics (BLEU, NIST, METEOR (Banerjee and Lavie, 2005) (v0.7), and TER) as our baselines. Since our models are trained to regress human evaluation scores, to make a direct comparison in the same regression setting, we also train a small linear regression model for each baseline metric in the same way as descried in Pado et al. (2009). These regression models are strictly more powerful than the baseline metrics and show higher robustness and better correlation with human

[2] Recall that we transform *ref* into *sys*, and thus on the *sys* side, we can only insert but not delete. The argument applies equally to the case where the jump was made on the other side.

[3] The length of the longest edit sequence with jumps only increased by $0.5 * max(|\mathbf{s}|, |\mathbf{r}|)$ in the worst case, and by and large swapping is rare in comparison to basic edits.

Data Set		**Our Metrics**			**Baseline Metrics**						
train	test	pFSM	pPDA	pPDA+*f*	BLEUR	NISTR	TERR	METR	MTR	RTER	MT+RTER
A+C	U	54.6	55.3	**57.2**	49.9	49.5	50.1	49.1	50.1	54.5	55.6
A+U	C	59.9	63.8	**65.7**	53.9	53.1	50.3	61.1	57.3	58.0	62.7
C+U	A	**61.2**	60.4	59.8	52.5	50.4	54.5	60.1	55.2	59.9	**61.1**
MT08	MT06	**65.2**	63.4	64.5	57.6	55.1	63.8	62.1	62.6	62.2	**65.2**

Table 1: Overall results on OpenMT08 and OpenMT06 evaluation data sets. The R (as in BLEUR) refers to the regression model trained for each baseline metric, same as Pado et al. (2009). The first three rows are round-robin train/test results over three languages on OpenMT08 (A=Arabic, C=Chinese, U=Urdu). The last row are results trained on entire OpenMT08 (A+C+U) and tested on OpenMT06. Numbers in this table are Spearman's rank correlation ρ between human assessment scores and model predictions. The pPDA column describes our pPDA model with jump distance limit 5. METR is shorthand for METEORR. +*f* means the model includes synonyms and paraphrase features (*cf.* Section 4). Best results and scores that are not statistically significantly worse are highlighted in bold in each row.

judgments. [4] We also compare our models with the state-of-the-art linear regression models reported in Pado et al. (2009) that combine features from multiple MT evaluation metrics (MT), as well as rich linguistic features from a textual entailment system (RTE).

In all of our experiments, each reference and system translation sentence pair is tokenized using the PTB (Marcus et al., 1993) tokenization script, and lemmatized by the Porter Stemmer (Porter, 1980). Statistical significance tests are performed using the paired bootstrap resampling method (Koehn, 2004).

We divide our experiments into two sections, based on two different prediction tasks — predicting absolute scores and predicting pairwise preference.

5.1 Exp. 1: Predicting Absolute Scores

The first task is to evaluate a system translation on a seven point Likert scale against a single reference. Higher scores indicate translations that are closer to the meaning intended by the reference. Human ratings in the form of absolute scores are available for standard evaluation data sets such as NIST OpenMT06,08.[5] Since our model makes predictions at the granularity of a whole sentence, we focus on sentence-level evaluation. A metric's goodness is judged by how well it correlates with human judgments, and Spearman's rank correlation (ρ) is reported for all experiments in this section.

We used the NIST OpenMT06 corpus for development purposes, and reserved the NIST OpenMT08 corpus for post-development evaluation. The OpenMT06 data set contains 1,992 English translations of Arabic newswire text from 8 MT systems. For development, we used a 2-fold cross-validation scheme with splits at the first 1,000 and last 992 sentences. The OpenMT08 data set contains English translations of newswire text from three languages (Arabic has 2,769 pairs from 13 MT systems; Chinese has 1,815 pairs from 15; and Urdu has 1,519 pairs, from 7). We followed the same experimental setup as Pado et al. (2009), using a "round robin" training/testing scheme, i.e., we train a model on data from two languages, making predictions for the third. We also show results of models trained on the entire OpenMT08 data set and tested on OpenMT06.

Overall Comparison

Results of our proposed models compared against the baseline models described in Pado et al. (2009) are shown in Table 1. The pFSM and pPDA models do not use any additional information other than words and lemmas, and thus make a fair comparison with the baseline metrics. [6] We can see from the table that pFSM significantly outperforms all baselines on Urdu and Arabic, but trails behind METEORR on Chinese by a small margin (1.2 point in Spearman's ρ). On Chinese data set, the pPDA extension gives results significantly better than the best baseline metrics for Chinese (2.7 better than METEORR). It is also significantly better than pFSM (by

[4] See Pado et al. (2009) for more discussion.

[5] Available from `http://www.nist.gov`.

[6] METEORR actually has an unfair advantage in this comparison, since it uses synonym information from WordNet; TERR on the other hand has a disadvantage because it does not use lemmas. Lemma is added later in the TERplus extension (Snover et al., 2009).

3.9 points), suggesting that modeling word swapping is particularly rewarding for Chinese language. On the other hand, pPDA model does not perform better than the pFSM model on Arabic in MT08 and OpenMT06 (which is also Arabic-to-English). This observation is consistent with findings in earlier work that Chinese-English translations exhibit much more medium and long distance reordering than languages like Arabic (Birch et al., 2009).

Both the pFSM and pPDA models also significantly outperform the MtR linear regression model that combines the outputs of all four baselines, on all three source languages. This demonstrates that our regression model is more robust and accurate than a state-of-the-art system combination linear-regression model. The RteR and Mt+RteR linear regression models benefit from the rich linguistic features in the textual entailment system's output. It has access to all the features in pPDA+*f* such as paraphrase and dependency parse relations, and many more (e.g., Norm Bank, part-of-speech, negation, antonyms). However, our pPDA+*f* model rivals the performance of RteR and Mt+RteR on Arabic (with no statistically significant difference from RteR), and greatly improve over these two models on Urdu and Chinese. Most noticeably, pPDA+*f* is 7.7 points better than RteR on Chinese.

5.2 Exp. 2: Predicting Pairwise Preferences

To further test our model's robustness, we evaluate it on WMT data sets with a different prediction task in which metrics make pairwise preference judgments between translation systems. The WMT06-08 data sets are much larger in comparison to the OpenMT06 and 08 data. They contain MT outputs of over 40 systems from five different source languages (French, German, Spanish, Czech, and Hungarian). The WMT06, 07 and 08 sets contains 10,159, 5,472 and 6,856 sentence pairs, respectively. We used portions of WMT 06 and 07 data sets [7] that are annotated with absolute scores on a five point scale for training, and the WMT08 data set annotated with pairwise preference for testing.

To generate pairwise preference predictions, we first predict an absolute score for each system translation, then compare the scores between each system pair, and give preference to the higher score. We adopt the sentence-level evaluation metric used in Pado et al. (2009), which measures the consistency (accuracy) of metric predictions with human preferences. The random baseline for this task on WMT08 data set is 39.8%.

Models	WMT06	WMT07	WMT06+07
pPDA+*f*	51.6	**52.4**	52.0
BleuR	49.7	49.5	49.6
MeteorR	51.4	51.4	51.5
NistR	50.0	50.3	50.2
TerR	50.9	51.0	51.2
MtR	50.8	51.5	51.5
RteR	51.8	50.7	51.9
Mt+RteR	**52.3**	51.8	**52.5**

Table 2: Pairwise preference prediction results on WMT08 test set. Each column shows a different training data set. Numbers in this table are model's consistency with human pairwise preference judgments. Best result on each test set is highlighted in bold.

Results are shown in Table 2. Similar to the results on OpenMT experiments, our model consistently outperformed BleuR, MeteorR, NistR and TerR. Our model also gives better performance than the MtR ensemble model on all three tests; and ties with RteR in two out of the three tests but performs significantly better on the other test. The Mt+RteR ensemble model is better on two tests, but worse on the other. But overall the two systems are quite comparable, with less than 0.6% accuracy difference. The results also show that our method is stable across different training sets, with test accuracy differences less than 0.4%.

6 Conclusion

We described the SPEDE metric for sentence level MT evaluation. It is based on probabilistic finite state machines to compute weighted edit distance. Our model admits a rich set of linguistic features, and can be trained to learn feature weights automatically by optimizing a regression objective. A novel pushdown automaton extension was also presented for capturing long-distance word swapping. Our metrics achieve state-of-the-art results on a wide range of standard evaluations, and are much more lightweight than previous regression models.

[7] Available from `http://www.statmt.org`.

Acknowledgements

We gratefully acknowledge the support of Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181 and the support of the DARPA Broad Operational Language Translation (BOLT) program through IBM. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of the DARPA, AFRL, or the US government.

References

J. Albrecht and R. Hwa. 2007a. A re-examination of machine learning approaches for sentence-level MT evaluation. In *Proceedings of ACL*.

J. Albrecht and R. Hwa. 2007b. Regression for sentence-level MT evaluation with pseudo references. In *Proceedings of ACL*.

S. Banerjee and A. Lavie. 2005. Meteor: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of ACL Workshop on Intrinsic and Extrinsic Evaluation Measures*.

A. Birch, P. Blunsom, and M. Osborne. 2009. A quantitative analysis of reordering phenomena. In *Proceedings of WMT 09*.

C. Callison-Burch, P. Koehn, C. Monz, and J. Schroeder. 2009. Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*.

C. Callison-Burch, P. Koehn, C. Monz, K. Peterson, M. Przybocki, and O. Zaidan. 2010. Findings of the 2010 joint workshop on Statistical Machine Translation and metrics for Machine Translation. In *Proceedings of Joint WMT 10 and MetricsMatr Workshop at ACL*.

C. Callison-Burch, P. Koehn, C. Monz, and O. Zaidan. 2011. Findings of the 2011 workshop on statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*.

T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. 2001. *Introduction to Algorithms, Second Edition*. MIT Press.

M. Denkowski and A. Lavie. 2010. Extending the METEOR machine translation evaluation metric to the phrase level. In *Proceedings of HLT/NAACL*.

G. Doddington. 2002. Automatic evaluation of machine translation quality using n-gram cooccurrence statistics. In *Proceedings of HLT*.

Y. Dombb, O. Lipsky, B. Porat, E. Porat, and A. Tsur. 2010. The approximate swap and mismatch edit distance. *Theoretical Computer Science*, 411(43).

J. Eisner. 2002. Parameter estimation for probabilistic finite-state transducers. In *Proceedings of ACL*.

J. Esparza and A. Kucera. 2005. Quantitative analysis of probabilistic pushdown automata: Expectations and variances. In *Proceedings of the 20th Annual IEEE Symposium on Logic in Computer Science*.

Y. He, J. Du, A. Way, and J. van Genabith. 2010. The DCU dependency-based metric in WMT-MetricsMATR 2010. In *Proceedings of Joint WMT 10 and MetricsMatr Workshop at ACL*.

K. Knight and Y. Al-Onaizan. 1998. Translation with finite-state devices. In *Proceedings of AMTA*.

P. Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*.

S. Kumar and W. Byrne. 2003. A weighted finite state transducer implementation of the alignment template model for statistical machine translation. In *Proceedings of HLT/NAACL*.

D. C. Liu and J. Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Math. Programming*, 45:503–528.

D. Liu, , and D. Gildea. 2005. Syntactic features for evaluation of machine translation. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures*.

M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini. 1993. Building a large annotated corpus of english: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller. 1990. WordNet: an on-line lexical database. *International Journal of Lexicography*, 3(4).

F. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL*.

K. Owczarzak, J. van Genabith, and A. Way. 2008. Evaluating machine translation with LFG dependencies. *Machine Translation*, 21(2):95–119.

S. Pado, M. Galley, D. Jurafsky, and C. D. Manning. 2009. Robust machine translation evaluation with entailment features. In *Proceedings of ACL*.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of ACL*.

M.F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of AMTA*.

M. Snover, , N. Madnani, B. Dorr, and R. Schwartz. 2009. Fluency, adequacy, or HTER? exploring different human judgments with a tunable MT metric. In *Proceedings of WMT09 Workshop*.

S. Sun, Y. Chen, and J. Li. 2008. A re-examination on features in regression based approach to automatic MT evaluation. In *Proceedings of ACL*.

E. Vidal, F. Thollard, C. de la Higuera, F. Casacuberta, and R. C. Carrasco. 2005. Probabilistic finite-state machines part I. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1013–1025.

D. Wu, 2010. *CRC Handbook of Natural Language Processing*, chapter How to Select an Answer String?, pages 367–408. CRC Press.

L. Zhou, C.Y. Lin, and E. Hovy. 2006. Re-evaluating machine translation results with paraphrase support. In *Proceedings of EMNLP*.