Probabilistic Finite State Machines for Regression-based MT Evaluation

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

Abstract

Accurate and robust metrics for automatic evaluation are key to the development of statistical machine translation (MT) systems. We first introduce a new regression model that uses a probabilistic finite state machine (pFSM) to compute weighted edit distance as predictions of translation quality. We also propose a novel pushdown automaton extension of the pFSM model for modeling word swapping and cross alignments that cannot be captured by standard edit distance models. Our models can easily incorporate a rich set of linguistic features, and automatically learn their weights, eliminating the need for ad-hoc parameter tuning. Our methods achieve state-of-the-art correlation with human judgments on two different prediction tasks across a diverse set of standard evaluations (NIST OpenMT06,08; WMT06-08).

1 Introduction

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*). Recent studies have also confirmed that tuning MT systems against better MT metrics — using algorithms like MERT (Och, 2003) — leads to better system performance (He and Way, 2009; Liu et al., 2011).

Later metrics that move beyond n-grams achieve higher accuracy and improved robustness from resources like WordNet synonyms (Miller et al., 1990), and paraphrasing (Snover et al., 2009; Denkowski and Lavie, 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 (Kulesza and Shieber, 2004; 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 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.

In this paper, we propose a simple yet powerful probabilistic Finite State Machine (pFSM) for the task of MT evaluation. It is built on the backbone of weighted edit distance models, but learns to weight edit operations in a probabilistic 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 experiment with a set of simple features based on labeled head-modifier dependency structure, in order to test the hypothesis that modeling overall sentence structure can lead to more accurate evaluation measures.

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

2 pFSMs for MT Regression

We start off by framing the problem of machine translation evaluation in terms of weighted edit distances 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 and probabilistic finite state transducers 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 pFSM 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:

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

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 ($\hat{y}$). 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:

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

$\mathbf{e}^*$ denotes a valid edit sequence. Since the "gold" edit sequence are not given at training or prediction time, we treat the edit sequences as hidden variables and sum them out. 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:

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

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

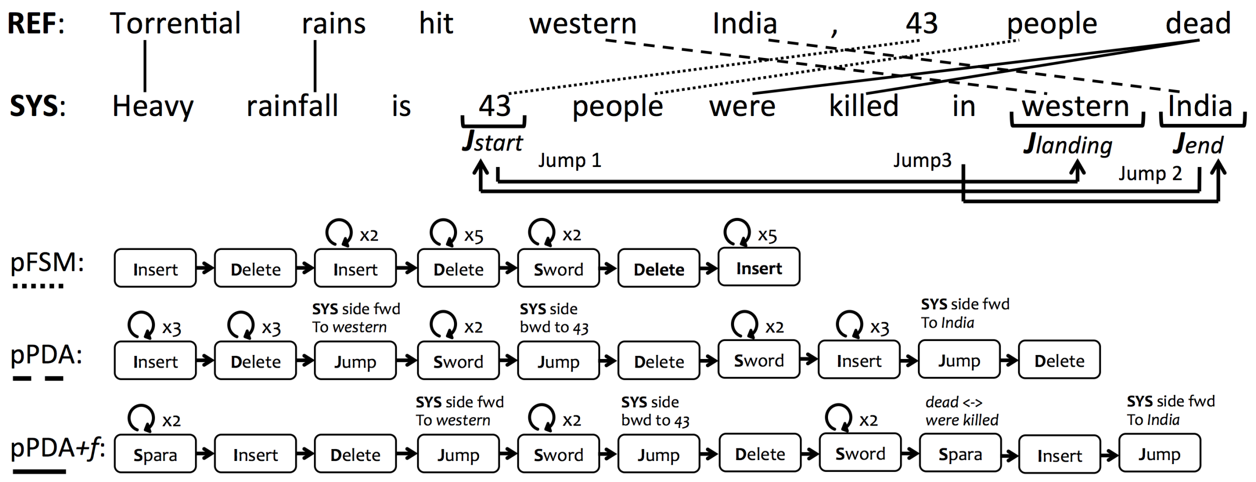
REF: Torrential rains hit western India , 43 people dead
SYS: Heavy rainfall is 43 people were killed in western India
Jstart
Jlanding
Jend
Jump 1
Jump3
Jump 2
pFSM: Insert Delete Insert x2 Delete x5 Sword x2 Delete Insert x5
pPDA: Insert x3 Delete x3 Jump (SYS side fwd To western) Sword x2 Jump (SYS side bwd to 43) Delete Sword x2 Insert x3 Jump (SYS side fwd To India) Delete
pPDA+f: Spara x2 Insert Delete Jump (SYS side fwd To western) Sword x2 Jump (SYS side bwd to 43) Delete Sword x2 Spara (dead <-> were killed) Insert Jump (SYS side fwd To India)

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 sequences 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 (on a 7-point scale), and model predictions are: pPDA+f 5.5, pPDA 4.3, pFSM 3.1, METEORR 3.2, TERR 2.8.

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. [1] 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 terms of 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.

[1]It is safe to ignore the second and third row in Figure 1 for now, their explanations are forthcoming in Section 2.2.

2.1 Soft-max Interpretation

There is also an alternative interpretation of the model as a simple soft-max approximation that is very intuitive and easy to understand. For ease of illustration, we introduce a quantity $Q(\mathbf{e} \mid \mathbf{s}, \mathbf{r})$ to be the *score* of an edit sequence, defined simply as the sum of the dot product of feature values and feature weights:

$$Q(\mathbf{e} \mid \mathbf{s}, \mathbf{r}) = \sum_{i=1}^{|\mathbf{e}|} \theta \cdot \mathbf{f}(e_{i-1}, e_i, \mathbf{s}, \mathbf{r})$$

For the regression task, the intuition is that we want y to take on the score (Q) of the **best** edit sequence:

$$y = \max_{\mathbf{e} \subseteq \mathbf{e}^*} Q(\mathbf{e} \mid \mathbf{s}, \mathbf{r})$$

But since the *max* function is non-differentiable, we replace it with a *softmax*:

$$y = \underbrace{\log \sum_{\mathbf{e} \subseteq \mathbf{e}^*} \exp}_{softmax} Q(\mathbf{e} \mid \mathbf{s}, \mathbf{r})$$

Substituting in Q, we arrive at the same objective function as (2).

2.2 Restricted 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. [2] 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 restricted 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" 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, [3] 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 *Insert*, 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 a general pPDA model without the limited distance and non-nestedness jump constraints, there could be recursive jump structures, which violates the finite state property that we are looking for. The constraints we introduced upper-bounds possible reordering, and the resulting model is finite state. In practice, we found that our extension gives a big boost to model performance (*cf.* Section 4.1), with only a modest increase in computation time. [4]

2.3 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. Two variants of the

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

[3]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.

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

forward-backward style dynamic programming algorithm were used for computing gradients in the pFSM and pPDA models, similar to other sequence models such as HMMs and CRFs. Details are omitted here for brevity.

3 Rich Linguistic Features

In this section we will add new substitution operations beyond those introduced in Section 2, to capture various linguistic phenomena. These new substitution operations correspond to new transition states in the pPDA.

3.1 Synonyms

Our first set of features matches words that have synonym relations according to WordNet (Miller et al., 1990). Synonyms have been found to be very useful in METEOR and TERplus, and can be easily built into our model as a new substitution operation S_{syn}.

3.2 Paraphrasing

Newer versions of METEOR and TERplus both found that inclusion of phrase-based matching greatly improves model robustness and accuracy (Denkowski and Lavie, 2010; Snover et al., 2009). We add a substitution operator (S_{para}) that matches words that are paraphrases. 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 n-gram block substitutions. In preliminary experiments, we found that most of the gain came from unigrams and bigrams, with little to no additional gains from trigrams. Therefore, we limited our experiments to bigram pFSM and pPDA models, and pruned the paraphrase table adopted from TERplus [5] to unigrams and bigrams, resulting in 2.5 million paraphrase pairs.

3.3 Sentence Structure

A problem that remains largely unaddressed by most popular MT evaluation metrics is the overall goodness of the translated sentence's structure (Liu et al., 2005; Owczarzak et al., 2008). Translations with good local n-gram coverage but horrible global syntactic ordering are not unusual in SMT outputs. Such translations usually score well with existing metrics but poorly among human evaluators.

In our model, when we detect consecutive bigram substitutions in the state transition, we examine the head-modifier dependency between the two words on each side of the sentence pair. A feature is triggered if and only if there is a head-modifier relation between the two words on each side, the labeled dependency on the two sides match, and it is one of subject, object or predicative relations. We deliberately left out features that model mismatches of dependency labels, because we found parsing output from translations to be usually very poor. Since parsing results are generally more reliable for more fluent translations, our hope is that by only modeling parse matches, our model will be able to pick them up as positive signals, indicating good translation quality.

4 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 (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 described 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 judgments. [6] We also compare our models with the state-of-the-art linear regression models reported in Pado et al. (2009) that

[5] Available from `www.umiacs.umd.edu/~snover/terp`.

[6] The baseline metric (e.g., BLEU) computes its raw score by taking the geometric mean of n-gram precision scores ($1 \leq n \leq 4$) scaled by a brevity penalty. The regression model learns to combine these fine-grained scores more intelligently, by optimizing their weights to regress human judgments. See Pado et al. (2009) for more discussion.

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 Penn Treebank (Marcus et al., 1993) tokenization script, and lemmatized by the Porter Stemmer (Porter, 1980). For the overall sentence structure experiment, translations are additionally part-of-speech tagged with MXPOST tagger (Ratnaparkhi, 1996), and parsed with MSTParser (McDonald et al., 2005)[7] labeled dependency parser. 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.

4.1 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.[8] 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.

4.1.1 pFSM vs. pPDA

Data Set		pFSM		pPDA			
tr	*te*	*n*1	*n*2	*j*1	*j*2	*j*5	*j*10
A+C	U	54.6	54.8	**55.6**	55.0	55.3	55.3
A+U	C	59.9	59.8	58.0	61.4	63.8	**64.0**
C+U	A	**61.2**	**61.2**	60.2	59.9	60.4	60.2

Table 1: pFSM vs. pPDA results for the round-robin approach on OpenMT08 data set over three languages (A=Arabic, C=Chinese, U=Urdu). Numbers in this table are Spearman's ρ for correlation between human assessment scores and model predictions; *tr* stands for training set, and *te* stands for test set. *nx* means the model has *x*-gram block edits. *jy* means the model has jump distance limit *y*. The Best result for each test set row is highlighted in bold.

The second and third columns under the pFSM label in Table 1 compares our bigram block edit extension for the pFSM model. Although we do not yet see a significant performance gain (or loss) from adding block edits, they will enable longer paraphrase matches in later experiments.

Columns 5 through 8 in Table 1 show experimental results validating the contribution of our pPDA extension to the pFSM model (*cf.* Section 2.2). We can see that the pPDA extension gave modest improvements on the Urdu test set, but at a small decrease in performance on the Arabic data. However, for Chinese, there is a substantial gain, particularly with jump distances of five or longer. This trend is even more pronounced at the long jump distance of 10, consistent with the observation that Chinese-English translations exhibit much more medium and long distance reordering than languages like Arabic (Birch et al., 2009).

4.1.2 Evaluating Linguistic Features

Experimental results evaluating the benefits of each linguistic feature set are presented in Table 3. The first row is the pPDA model with jump distance limit 5, without other additional features. The next three rows are the results of adding each of the three feature sets described in Section 3.

Overall, we observed that only paraphrase matching features gave a significant boost to performance.

[7]Trained on the entire Penn Treebank.

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

Data Set		Our Metrics			Baseline Metrics					Combined 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	**65.8**	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 2: Overall Comparison: Results from 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, paraphrase and parsing features (*cf.* Section 3). Best results and scores that are not statistically significantly worse are highlighted in bold in each row.

	Urdu	Chinese	Arabic
pPDA	55.3	63.8	60.4
+Synonym	55.6	63.7	**60.7**
+Tree	55.3	63.8	60.3
+Paraphrase	57.1	65.4	60.0
+Syn+Tree+Para	**57.2**	**65.8**	59.8

Table 3: Results for OpenMT08 with linguistic features, using the same round robin scheme as in Table 1. Numbers in this table are Spearman's rank correlation ρ between human assessment scores and model predictions. Best results on each test set are highlighted in bold.

The row starting with pPDA+*f* in Figure 1 shows an example where adding paraphrase features allow pPDA+*f* to find more correct alignments and make better predictions than pPDA.

No significant improvements from synonym and dependency tree matching features are evident from the results. An examination of the feature statistics in training data showed that the parse tree features have very low occurrence counts. On the Chinese+Urdu training set, for example, the features for subject, object and predicative labeled dependency matches fired only 55, 784 and 13 times, respectively. As a reference point for the scale of feature counts, the "same word" match feature fired 875,375 times on the same data set. And our qualitative assessment of the labeled dependency parser outputs was that the quality is very poor on system translations. For future work, more elaborate parse feature engineering could be a promising direction, but is outside the scope of our study.

In combination, the joint feature set of synonym, paraphrase and parse tree features gave modest improvements over the paraphrase feature alone on the Chinese test set.

4.1.3 Overall Comparison

Results of our proposed models compared against the baseline models described in Pado et al. (2009) are shown in Table 2. The pPDA+*f* model has access to paraphrase information, which is not available to the baselines, so it should not be directly compared with. But 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. [9] 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). 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. Both pFSM and pPDA learned to assign a lower negative feature weight for deletion than insertion (i.e., it is bad to insert an unseen word into system trans-

[9] 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).

lation, but worse if words from reference translation are deleted), which corresponds to the setting in METEOR where recall is given more importance than precision (Banerjee and Lavie, 2005).

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.

Consistent with our earlier observation on OpenMT08 data set that the pPDA model performs slightly worse than the pFSM model on Arabic, the same performance decrease is seen in OpenMT06 data set, which is also Arabic-to-English.

As shown earlier in Table 3, the combined set of paraphrase, parsing and synonym features in pPDA+*f* helps for Urdu and Chinese, but not for Arabic. Here we found that even though the pPDA+*f* model is still worse than pFSM on OpenMT06 tests, it did give a decent improvement to pPDA model, closing up the gap with pFSM.

Other than robustness and accuracy, simplicity is also an important trait we seek in good MT metrics. Our models only have a few tens of features (instead of hundreds of features as found in RTER and MT+RTER), which makes interpretation of the model's prediction relatively easy. On an important practical note, our model is much more lightweight than the RTER or MTR system. It runs at a much faster speed with a smaller memory footprint, hence potentially useable in MERT training.

4.2 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 [10] 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%. [11]

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 4: 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 4. 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%.

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

[11] The random baseline is not 50% for two reasons: (1) human judgments include contradictory and tie annotations; (2) transitivity constraints need to be respected in total ordering. For details, see Pado et al. (2009).

4.3 Qualitative Analysis

Example (1) shows a system and reference translation pair in the Chinese test portion of OpenMT08.

(1) **REF:** Two Jordanese sentenced$_1$ for plotting$_2$ an attack$_3$ on Americans$_4$
SYS: The name of Jordan plotting$_2$ attacks$_3$ Americans$_4$ were sentenced$_1$ to death

Human annotators give this example a score of 4.0, but TERR and METEORR both assigned erroneously low scores (1.0 and 2.2, respectively). Words with the same subscript index were aligned by pPDA model. This example exhibits a word swapping phenomenon, and our model was able to capture it correctly. TERR clearly suffered from not being able to model word swapping in this case. It also missed out the word pair *attack* and *attacks* due to the lack of lemma support. The reason why METEORR assigned such a low score for this example is because none of the matched words in the reference were adjacent to each other, causing a high fragmentation penalty. The fragmentation penalty term has two parameters that need to be manually tuned, and has a high variance across examples and data sets. This example illustrates models that require ad-hoc tuning tend not to be robust. Our pPDA model (without linguistic feature) was able to make a prediction of 3.7, much closer to human judgment.

4.4 MetricsMATR10 and WMT12 Results

An earlier version of the pFSM model that was trained on the OpenMT08 data set was submitted to the single reference sentence level track at MetricsMATR10 (Peterson and Przybocki, 2010) NIST evaluation. Even though our system was not in the most ideal state at the time of the evaluation, [12] and was trained on a small amount of data, the pFSM model still performed competitively against other metrics. Noticeably, we achieved second best results for Human-targeted Translation Edit Rate (HTER) assessment, trailing behind TERplus with no statistically significant difference. On average, our system made 5th place among 15 different sites and 7th place among 25 different metrics, averaged across 9 assessment types.

[12]Unfortunately the version we submitted in 2010 was plagued with a critical bug. More general enhancements have been made to the model since.

We submitted the version of the pPDA+f model trained on the WMT07 dataset to the "into English" segment-leval track of the WMT 2012 Shared Evaluation Metrics Task (Callison-Burch et al., 2012). Our model achieved the highest score (measured by Kendall's tau correlation) on all four language pairs (Fr-En, De-En, Es-En and Cs-En), and tied for the first place with METEOR v1.3 on average correlation.

5 Related Work

Features and Representation

One of the findings in our experimentation is that paraphrasing helps boosting model accuracy, and the idea of using paraphrases in MT evaluation was first proposed by Zhou et al. (2006). Several recent studies have introduced metrics over dependency parses (Liu et al., 2005; Owczarzak et al., 2008; He et al., 2010), but their improvements over n-gram models at the sentence level are not always consistent (Liu et al., 2005; Peterson and Przybocki, 2010). Other than string-based methods, recent work has explored more alternative representations for MT evaluation, such as network properties (Amancio et al., 2011), semantic role structures (Lo and Wu, 2011), and the quality of word order (Birch and Osborne, 2011).

Modeling

The idea of using extended edit distance models with block movements was also explored in Leusch et al. (2003). However, their model is largely empirical and not in a probabilistic learning setting. The line of work on probabilistic tree-edit distance models bears a strong connection to this work (McCallum et al., 2005; Bernard et al., 2008; Wang and Manning, 2010; Emms, 2012). In particular, our pFSM model and the log-linear parameterization were inspired by Wang and Manning (2010). Another body of literature that is closely related to this work is FSM models for word alignment (Vogel et al., 1996; Saers et al., 2010; Berg-Kirkpatrick et al., 2010). The stochastic Inversion Transduction Grammar in Saers et al. (2010) for instance, is a pFSM with special constraints. More recently, Saers and Wu (2011) further explored the connection between Linear Transduction Grammars and FSMs. There is a close tie between our pFSM model and the HMM model in Berg-Kirkpatrick et al. (2010). Both models adopted a log-linear parameterization for the state transition distribution, [13] but in their case the HMM model and the pFSM arc weights are normalized locally, and the objective is non-convex.

6 Conclusion

We described a probabilistic finite state machine based on string edits and a novel pushdown automaton extension for the task of machine translation evaluation. The models admit a rich set of linguistic features, and are trained to learn feature weights automatically by optimizing a regression objective. The proposed models achieve state-of-the-art results on a wide range of standard evaluations, and are much more lightweight than previous regression models, making them suitable candidates to be used in MERT training.

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*.

D.R. Amancio, M.G.V. Nunes, O.N. Oliveira Jr., T.A.S. Pardo, L. Antiqueira, and L. da F. Costa. 2011. Using metrics from complex networks to evaluate machine translation. *Physica A*, 390(1):131–142.

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*.

T. Berg-Kirkpatrick, A. Bouchard-Cote, J. DeNero, and D. Klein. 2010. Painless unsupervised learning with features. In *Proceedings of NAACL*.

M. Bernard, L. Boyer, A. Habrard, and M. Sebban. 2008. Learning probabilistic models of tree edit distance. *Pattern Recognition*, 41(8):2611–2629.

A. Birch and M. Osborne. 2011. Reordering metrics for MT. In *Proceedings of ACL/HLT*.

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*.

C. Callison-Burch, P. Koehn, C. Monz, M. Post, R. Soricut, and L. Specia. 2012. Findings of the 2012 workshop on Statistical Machine Translation. In *Proceedings of Seventh Workshop on Statistical Machine Translation at NAACL*.

T. Cormen, C. Leiserson, R. 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*.

M. Emms. 2012. On stochastic tree distances and their training via expectation-maximisation. In *Proceedings of International Conference on Pattern Recognition Application and Methods*.

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*.

[13]Similar parameterization was also used in much previous work, such as Riezler et al. (2000).

Y. He and A. Way. 2009. Improving the objective function in minimum error rate training. In *Proceedings of MT Summit XII*.

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 Metrics-Matr 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*.

A. Kulesza and S. Shieber. 2004. Robust machine translation evaluation with entailment features. In *Proceedings of TMI*.

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*.

G. Leusch, N. Ueffing, and H. Ney. 2003. A novel string-to-string distance measure with applications to machine translation evaluation. In *Proceedings of MT Summit I*.

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*.

C. Liu, D. Dahlmeier, and H. Ng. 2011. Better evaluation metrics lead to better machine translation. In *Proceedings of EMNLP*.

C. Lo and D. Wu. 2011. MEANT: An inexpensive, high-accuracy, semi-automatic metric for evaluating translation utility based on semantic roles. In *Proceedings of ACL/HLT*.

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.

A. McCallum, K. Bellare, and F. Pereira. 2005. A conditional random field for discriminatively-trained finite-state string edit distance. In *Proceedings of UAI*.

R. McDonald, K. Crammer, and F. Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL*.

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. A learning approach to improving sentence-level MT evaluation. 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*.

K. Peterson and M. Przybocki. 2010. NIST 2010 metrics for machine translation evaluation (MetricsMaTr10) official release of results.

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

A. Ratnaparkhi. 1996. A maximum entropy part-of-speech tagger. In *Proceedings of EMNLP*.

S. Riezler, D. Prescher, J. Kuhn, and M. Johnson. 2000. Lexicalized stochastic modeling of constraint-based grammars using log-linear measures and em training. In *Proceedings of ACL*.

M. Saers and D. Wu. 2011. Linear transduction grammars and zipper finite-state transducers. In *Proceedings of Recent Advances in Natural Language Processing*.

M. Saers, J. Nivre, and D. Wu. 2010. Word alignment with stochastic bracketing linear inversion transduction grammar. In *Proceedings of NAACL*.

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.

S. Vogel, H. Ney, and C. Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of COLING*.

M. Wang and C.D. Manning. 2010. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of COLING*.

D. Wu, 2010. *CRC Handbook of Natural Language Processing*, chapter Alignment, 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*.