

# Integrating N-best SMT Outputs into a TM System

Yifan He   Yanjun Ma   Andy Way   Josef van Genabith

Centre for Next Generation Localisation

School of Computing

Dublin City University

{yhe, yma, away, josef}@computing.dcu.ie

## Abstract

In this paper, we propose a novel framework to enrich Translation Memory (TM) systems with Statistical Machine Translation (SMT) outputs using ranking. In order to offer the human translators multiple choices, instead of only using the top SMT output and top TM hit, we merge the N-best output from the SMT system and the k-best hits with highest fuzzy match scores from the TM system. The merged list is then ranked according to the prospective post-editing effort and provided to the translators to aid their work. Experiments show that our ranked output achieve 0.8747 precision at top 1 and 0.8134 precision at top 5. Our framework facilitates a tight integration between SMT and TM, where full advantage is taken of TM while high quality SMT output is availed of to improve the productivity of human translators.

## 1 Introduction

Translation Memories (TM) are databases that store translated segments. They are often used to assist translators and post-editors in a Computer Assisted Translation (CAT) environment by returning the most similar translated segments. Professional post-editors and translators have long been relying on TMs to avoid duplication of work in translation.

With the rapid development in statistical machine translation (SMT), MT systems are begin-

ning to generate acceptable translations, especially in domains where abundant parallel corpora exist. It is thus natural to ask if these translations can be utilized in some way to enhance TMs.

However advances in MT are being adopted only slowly and sometimes somewhat reluctantly in professional localization and post-editing environments because of 1) the usefulness of the TM, 2) the investment and effort the company has put into TMs, and 3) the lack of robust SMT confidence estimation measures which are as reliable as fuzzy match scores (cf. Section 4.1.2) used in TMs. Currently the localization industry relies on TM fuzzy match scores to obtain both a good approximation of post-editing effort and an estimation of the overall translation cost.

In a forthcoming paper, we propose a translation recommendation model to better integrate MT outputs into a TM system. Using a binary classifier, we only recommend an MT output to the TM-user when the classifier is highly confident that it is better than the TM output. In this framework, post-editors continue to work with the TM while benefiting from (better) SMT outputs; the assets in TMs are not wasted and TM fuzzy match scores can still be used to estimate (the upper bound of) post-editing labor.

In the previous work, the binary predictor works on the 1-best output of the MT and TM systems, presenting either the one or the other to the post-editor. In this paper, we develop the idea further by moving from binary prediction to ranking. We use a ranking model to merge the k-best lists of the two systems, and produce a ranked merged

list for post-editing. As the list is an enriched version of the TM’s k-best list, the TM related assets are better preserved and the cost estimation is still valid as an upper bound.

More specifically, we recast SMT-TM integration as a ranking problem, where we apply the Ranking SVM technique to produce a ranked list of translations combining the k-best lists of both the MT and the TM systems. We use features independent of the MT and TM systems for ranking, so that outputs from MT and TM can have the same set of features. Ideally the translations should be ranked by their associated post-editing efforts, but given the very limited amounts of human annotated data, we use an automatic MT evaluation metric, TER (Snover et al., 2006), which is specifically designed to simulate post-editing effort to train and test our ranking model.

The rest of the paper is organized as follows: we first briefly introduce related research in Section 2, and review Ranking SVMs in Section 3. The formulation of the problem and experiments with the ranking models are presented in Sections 4 and 5. We analyze the post-editing effort approximated by the TER metric in Section 6. Section 7 concludes and points out avenues for future research.

## 2 Related Work

There has been some work to help TM users to apply MT outputs more smoothly. One strand is to improve the MT confidence measures to better predict post-editing effort in order to obtain a quality estimation that has the potential to replace the fuzzy match score in the TM. To the best of our knowledge, the first paper in this area is (Specia et al., 2009a), which uses regression on both the automatic scores and scores assigned by post-editors. The method is improved in (Specia et al., 2009b), which applies Inductive Confidence Machines and a larger set of features to model post-editors’ judgment of the translation quality between ‘good’ and ‘bad’, or among three levels of post-editing effort.

Another strand is to integrate high confidence MT outputs into the TM, so that the ‘good’ TM entries will remain untouched. In our forthcoming paper, we recommend SMT outputs to a TM user

when a binary classifier predicts that SMT outputs are more suitable for post-editing for a particular sentence.

The research presented here continues the line of research in the second strand. The difference is that we do not limit ourselves to the 1-best output but try to produce a k-best output in a ranking model. The ranking scheme also enables us to show all TM hits to the user, and thus further protects the TM assets.

There has also been work to improve SMT using the knowledge from the TM. In (Simard and Isabelle, 2009), the SMT system can produce a better translation when there is an exact or close match in the corresponding TM. They use regression Support Vector Machines to model the quality of the TM segments. This is also related to our work in spirit, but our work is in the opposite direction, i.e. using SMT to enrich TM.

Moreover, our ranking model is related to reranking (Shen et al., 2004) in SMT as well. However, our method does not focus on producing better 1-best translation output for an SMT system, but on improving the overall quality of the k-best list that TM systems present to post-editors. Some features in our work are also different in nature to those used in MT reranking. For instance we cannot use N-best posterior scores as they do not make sense for the TM outputs.

## 3 The Support Vector Machines

### 3.1 The SVM Classifier

Classical SVMs (Cortes and Vapnik, 1995) are binary classifiers that classify an input instance based on decision rules which minimize the regularized error function in (Eq. 1):

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^l \xi_i \\ \text{subject to:} \quad & y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned} \quad (1)$$

where  $(\mathbf{x}_i, y_i) \in R^n \times \{1, -1\}$  are  $l$  training instances.  $\mathbf{w}$  is the weight vector,  $\xi$  is the relaxation variable and  $C > 0$  is the penalty parameter.

### 3.2 Ranking SVM for SMT-TM Integration

The SVM classification algorithm is extended to the ranking case in (Joachims, 2002). For a cer-

tain group of instances, the Ranking SVM aims at producing a ranking  $r$  that has the maximum Kendall's  $\tau$  coefficient with the the gold standard ranking  $r^*$ .

Kendall's  $\tau$  measures the relevance of two rankings:  $\tau(r_a, r_b) = \frac{P-Q}{P+Q}$ , where  $P$  and  $Q$  are the amount of concordant and discordant pairs in  $r_a$  and  $r_b$ . In practice, this is done by building constraints to minimize the discordant pairs  $Q$ . Following the basic idea, we show how Ranking SVM can be applied to MT-TM integration as follows.

Assume that for each source sentence  $s$ , we have a set of outputs from MT,  $\mathbf{M}$  and a set of outputs from TM,  $\mathbf{T}$ . If we have a ranking  $r(s)$  over translation outputs  $\mathbf{M} \cup \mathbf{T}$  where for each translation output  $d \in \mathbf{M} \cup \mathbf{T}$ ,  $(d_i, d_j) \in r(s)$  iff  $d_i <_{r(s)} d_j$ , we can rewrite the ranking constraints as optimization constraints in an SVM, as in Eq. (2).

$$\begin{aligned} & \min_{w, b, \xi} \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum \xi \\ & \text{subject to:} \\ & \forall (d_i, d_j) \in r(s_1) : \mathbf{w}(\Phi(s_1, d_i) - \Phi(s_1, d_j)) \geq 1 - \xi_{i,j,1} \\ & \dots \\ & \forall (d_i, d_j) \in r(s_n) : \mathbf{w}(\Phi(s_n, d_i) - \Phi(s_n, d_j)) \geq 1 - \xi_{i,j,n} \\ & \xi_{i,j,k} \geq 0 \end{aligned} \quad (2)$$

where  $\Phi(s_n, d_i)$  is a feature vector of translation output  $d_i$  given source sentence  $s_n$ . The Ranking SVM minimizes the discordant number of rankings with the gold standard according to Kendall's  $\tau$ .

When the instances are not linearly separable, we use a mapping function  $\phi$  to map the features  $\mathbf{x}_i$  ( $\Phi(s_n, d_i)$  in the case of ranking) to high dimensional space, and solve the SVM with a kernel function  $K$  in where  $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ .

We perform our experiments with the Radial Basis Function (RBF) kernel, as in Eq. (3).

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2), \gamma > 0 \quad (3)$$

## 4 The Ranking-based Integration Model

In this section we present the Ranking-based SMT-TM integration model in detail. We first introduce the k-best lists in MT (called N-best list) and TM systems (called m-best list in this section) and then move on to the problem formulation and the feature set.

## 4.1 K-Best Lists in SMT and TM

### 4.1.1 The SMT N-best List

The N-best list of the SMT system is generated during decoding according to the internal feature scores. The features include language and translation model probabilities, reordering model scores and a word penalty.

### 4.1.2 The TM M-Best List and the Fuzzy Match Score

The m-best list of the TM system is generated in descending fuzzy match score. The fuzzy match score (Sikes, 2007) uses the similarity of the source sentences to predict a level to which a translation is reusable or editable.

The calculation of fuzzy match scores is one of the core technologies in TM systems and varies among different vendors. We compute fuzzy match cost as the minimum Edit Distance (Levenshtein, 1966) between the source and TM entry, normalized by the length of the source as in Eq. (4), as most of the current implementations are based on edit distance while allowing some additional flexible matching.

$$FuzzyMatch(t) = \min_e \frac{EditDistance(s, e)}{Len(s)} \quad (4)$$

where  $s$  is the source side of the TM hit  $t$ , and  $e$  is the source side of an entry in the TM.

## 4.2 Problem Formulation

Ranking lists is a well-researched problem in the information retrieval community, and Ranking SVMs (Joachims, 2002), which optimizes on the ranking correlation  $\tau$  have already been applied successfully in machine translation evaluation (Ye et al., 2007). We apply the same method here to rerank a merged list of MT and TM outputs.

Formally given an MT-produced N-best list  $\mathbf{M} = \{m_1, m_2, \dots, m_n\}$ , a TM-produced m-best list  $\mathbf{T} = \{t_1, t_2, \dots, t_m\}$  for a input sentence  $s$ , we define the gold standard using the TER metric (Snover et al., 2006): for each  $d \in \mathbf{M} \cup \mathbf{T}$ ,  $(d_i, d_j) \in r(s)$  iff  $TER(d_i) < TER(d_j)$ . We train and test a Ranking SVM using cross validation on a data set created according to this criterion. Ideally the gold standard would be created by human annotators. We choose to use TER

as large-scale annotation is not yet available for this task. Furthermore, TER has a high correlation with the HTER score (Snover et al., 2006), which is the TER score using the post-edited MT output as a reference, and is used as an estimation of post-editing effort.

### 4.3 The Feature Set

When building features for the Ranking SVM, we are limited to features that are independent of the MT and TM system. We experiment with system-independent fluency and fidelity features below, which capture translation fluency and adequacy, respectively.

#### 4.3.1 Fluency Features

**Source-side Language Model Scores.** We compute the LM probability and perplexity of the input source sentence on a language model trained on the source-side training data of the SMT system, which is also the TM database. The inputs that have lower perplexity on this language model are more similar to the data set on which the SMT system is built.

**Target-side Language Model Scores.** We compute the LM probability and perplexity as a measure of the fluency of the translation.

#### 4.3.2 Fidelity Features

**The Pseudo-Source Fuzzy Match Score.** We translate the output back to obtain a pseudo source sentence. We compute the fuzzy match score between the original source sentence and this pseudo-source. If the MT/TM performs well enough, these two sentences should be the same or very similar. Therefore the fuzzy match score here gives an estimation of the confidence level of the output.

**The IBM Model 1 Score.** We compute the IBM Model 1 score in both directions to measure the correspondence between the source and target, as it serves as a rough estimation of how good a translation it is on the word level.

## 5 Experiments

### 5.1 Experimental Settings

#### 5.1.1 Data

Our raw data set is an English–French translation memory with technical translation from a multi-national IT security company, consisting of 51K sentence pairs. We randomly select 43K to train an SMT system and translate the English side of the remaining 8K sentence pairs, which is used to run cross validation. Note that the 8K sentence pairs are from the same TM, so that we are able to create a gold standard by ranking the TER scores of the MT and TM outputs.

Duplicated sentences are removed from the data set, as those will lead to an exact match in the TM system and will not be translated by translators. The average sentence length of the training set is 13.5 words and the size of the training set is comparable to the (larger) translation memories used in the industry.

#### 5.1.2 SMT and TM systems

We use a standard log-linear PB-SMT model (Och and Ney, 2002): GIZA++ implementation of IBM word alignment model 4, the phrase-extraction heuristics described in (Koehn et al., 2003), minimum-error-rate training (Och, 2003), a 5-gram language model with Kneser-Ney smoothing trained with SRILM (Stolcke, 2002) on the English side of the training data, and Moses (Koehn et al., 2007) to decode. We train a system in the opposite direction using the same data to produce the pseudo-source sentences.

We merge distinct 5-best lists from MT and TM systems to produce a new ranking. To create the distinct list for the SMT system, we search over a 100-best list and keep the top-5 distinct outputs. Our data set consists of mainly short sentences, leading to many duplications in the N-best output of the SMT decoder. In such cases, top-5 distinct outputs are good representations of the SMT’s output.

### 5.2 Training, Tuning and Testing the Ranking SVM

We run training and prediction of the Ranking SVM in 4-fold cross validation. We use the

SVMlight<sup>1</sup> toolkit to perform training and testing.

When using the Ranking SVM with the RBF kernel, we have two free parameters to tune on: the cost parameter  $C$  in Eq. (1) and the radius parameter  $\gamma$  in Eq. (3). We optimize  $C$  and  $\gamma$  using a brute-force grid search before running cross-validation and maximize precision at top-5, with an inner 3-fold cross validation on the (outer) Fold-1 training set. We search within the range  $[2^{-6}, 2^9]$ , the step size is 2 on the exponent.

### 5.3 The Gold Standard

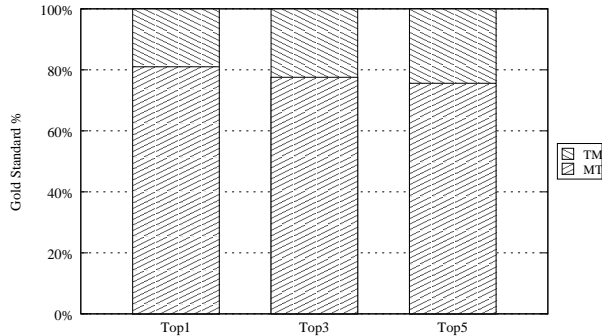


Figure 1: MT and TM’s percentage in gold standard

Figure 1 shows the composition of translations in the gold standard. Each source sentence is associated with a list of translations from two sources, i.e. MT output and TM matches. This list of translations is ranked from best to worst according to TER scores. The figure shows that over 80% of the translations are from the MT system if we only consider the top-1 translation. As the number of top translations we consider increases, more TM matches can be seen. On the one hand, this does show a large gap in quality between MT output and TM matches; on the other hand, however, it also reveals that we will have to ensure two objectives in ranking: the first is to rank the 80% MT translations higher and the second is to keep the 20% ‘good’ TM hits in the Top-5. We design our evaluation metrics accordingly.

### 5.4 Evaluation Metrics

The aim of this research is to provide post-editors with translations that in many cases are easier to

edit than the original TM output. As we formulate this as a ranking problem, it is natural to measure the quality of the ranking output by the number of better translations that are ranked high. Sometimes the top TM output is the easiest to edit; in such a case we need to ensure that this translation has a high rank, otherwise the system performance will degrade.

Based on this observation, we introduce the idea of *relevant* translations, and our evaluation metrics:  $\text{PREC@k}$  and  $\text{HIT@k}$ .

**Relevant Translations.** We borrow the idea of *relevance* from the IR community to define the idea of translations worth ranking high. For a source sentence  $s$  which has a top TM hit  $t$ , we define an MT/TM output  $m$  as relevant, if  $\text{TER}(m) \leq \text{TER}(t)$ . According to the definition, relevant translations should need no more post-edits than the original top hit from the TM system. Clearly the top TM hit is always relevant.

**PREC@k.** We calculate the precision ( $\text{PREC@k}$ ) of the ranking for evaluation. Assuming that there are  $n$  relevant translations in the top  $k$  list for a source sentence  $s$ , we have  $\text{PREC@k} = n/k$  for  $s$ . We test  $\text{PREC@k}$ , for  $k = 1..10$ , in order to evaluate the overall quality of the ranking.

**HIT@k.** We also estimate the probability of having one of the relevant translations in the top  $k$ , denoted as  $\text{HIT@k}$ . For a source sentence  $s$ ,  $\text{HIT@k}$  equals to 1 if there is at least one relevant translation in top  $k$ , and 0 otherwise. This measures the quality of the best translation in top  $k$ , which is the translation the post-editor will find and work on if she reads till the  $k$ th place in the list.  $\text{HIT@k}$  equals to 1.0 at the end of the list.

We report the mean  $\text{PREC@k}$  and  $\text{HIT@k}$  for all  $s$  with the 0.95 confidence interval.

### 5.5 Experimental Results

In Table 1 we report  $\text{PREC@k}$  and  $\text{HIT@k}$  for  $k = 1..10$ . The ranking receives 0.8747  $\text{PREC@1}$ , which means that most of the top ranked translations have at least the same quality as the top TM output. We notice that precision remains above 0.8 till  $k = 5$ , leading us to conclude that most of the *relevant* translations are ranked in the top-5 positions in the list.

<sup>1</sup><http://svmlight.joachims.org/>

Table 1: PREC@k and HIT@k of Ranking

	PREC %	HIT %
k=1	87.47±1.60	87.47±1.60
k=2	85.42±1.07	93.36±0.53
k=3	84.13±0.94	95.74±0.61
k=4	82.79±0.57	97.08±0.26
k=5	81.34±0.51	98.04±0.23
k=6	79.26±0.59	99.41±0.25
k=7	74.99±0.53	99.66±0.29
k=8	70.87±0.59	99.84±0.10
k=9	67.23±0.48	99.94±0.08
k=10	64.00±0.46	100.0±0.00

Using the HIT@k scores we can further confirm this argument. The HIT@k score grows steadily from 0.8747 to 0.9941 for  $k = 1\dots6$ , so most often there will be at least one *relevant* translation in top-6 for the post-editor to work with. After that room for improvement becomes very small.

In sum, both of the PREC@k scores and the HIT@k scores show that the ranking model effectively integrates the two translation sources (MT and TM) into one merged k-best list, and ranks the *relevant* translations higher.

Table 2: PREC@k - MT and TM Systems

	MT %	TM %
k=1	85.87±1.32	100.0±0.00
k=2	82.52±1.60	73.58±1.04
k=3	80.05±1.11	62.45±1.14
k=4	77.92±0.95	56.11±1.11
k=5	76.22±0.87	51.78±0.78

To measure whether the ranking model is effective compared to pure MT or TM outputs, we report the PREC@k of those outputs in Table 2. The k-best output used in this table is ranked by the MT or TM system, without being ranked by our model. We see the ranked outputs consistently outperform the MT outputs for all  $k = 1\dots5$  w.r.t. precision at a significant level, indicating that our system preserves some high quality hits from the TM.

The TM outputs alone are generally of much lower quality than the MT and Ranked outputs, as is shown by the precision scores for  $k = 2\dots5$ . But

TM translations obtain 1.0 PREC@1 according to the definition of the PREC calculation. Note that it does not mean that those outputs will need less post-editing (cf. Section 6.1), but rather indicates that each one of these outputs meet the lowest acceptable criterion to be *relevant*.

## 6 Analysis of Post-Editing Effort

A natural question follows the PREC and HIT numbers: after reading the ranked k-best list, will the post-editors edit less than they would have to if they did not have access to the list? This question would be best answered by human post-editors in a large-scale experimental setting. As we have not yet conducted a manual post-editing experiment, we try to measure the post-editing effort implied by our model with the edit statistics captured by the TER metric, sorted into four types: *Insertion*, *Substitution*, *Deletion* and *Shift*. We report the average number of edits incurred along with the 0.95 confidence interval.

### 6.1 Top-1 Edit Statistics

We report the results on the 1-best output of TM, MT and our ranking system in Table 3.

In the single best results, it is easy to see that the 1-best output from the MT system requires the least post-editing effort. This is not surprising given the distribution of the gold standard in Section 5.3, where most MT outputs are of better quality than the TM hits.

Moreover, since TM translations are generally of much lower quality as is indicated by the numbers in Table 3 (e.g. 2x as many substitutions and 3x as many deletions compared to MT), unjustly including very few of them in the ranking output will increase loss in the edit statistics. This explains why the ranking model has better ranking precision in Tables 1 and 2, but seems to incur more edit efforts. However, in practice post-editors can neglect an obvious ‘bad’ translation very quickly.

### 6.2 Top-k Edit Statistics

We report edit statistics of the Top-3 and Top-5 outputs in Tables 4 and 5, respectively. For each system we report two sets of statistics: the Best-statistics calculated on the best output (according

Table 3: Edit Statistics on Ranked MT and TM Outputs - Single Best

	Insertion	Substitution	Deletion	Shift
TM-Top1	$0.7554 \pm 0.0376$	$4.2461 \pm 0.0960$	$2.9173 \pm 0.1027$	$1.1275 \pm 0.0509$
MT-Top1	$0.9959 \pm 0.0385$	$2.2793 \pm 0.0628$	$0.8940 \pm 0.0353$	$1.2821 \pm 0.0575$
Rank-Top1	$1.0674 \pm 0.0414$	$2.6990 \pm 0.0699$	$1.1246 \pm 0.0412$	$1.2800 \pm 0.0570$

to TER score) in the list, and the Mean- statistics calculated on the whole Top-k list.

The Mean- numbers allow us to have a general overview of the ranking quality, but it is strongly influenced by the poor TM hits that can easily be neglected in practice. To control the impact of those TM hits, we rely on the Best- numbers to estimate the edits performed on the translations that are more likely to be used by post-editors.

In Table 4, the ranking output’s edit statistics is closer to the MT output than the Top-1 case in Table 3. Table 5 continues this tendency, in which the Best-in-Top5 Ranking output requires marginally less *Substitution* and *Deletion* operations and significantly less *Insertion* and *Shift* operations (starred) than its MT counterpart. This shows that when more of the list is explored, the advantage of the ranking model – utilizing multiple translation sources – begins to compensate for the possible large number of edits required by poor TM hits and finally leads to reduced post-editing effort.

There are several explanations to why the relative performance of the ranking model improves when  $k$  increases, as compared to other models. The most obvious explanation is that a single poor translation is less likely to hurt edit statistics on a  $k$ -best list with large  $k$ , if most of the translations in the  $k$ -best list are of good quality. We see from Tables 1 and 2 that the ranking output is of better quality than the MT and TM outputs w.r.t. precision. For a larger  $k$ , the small number of incorrectly ranked translations are less likely to be chosen as the Best- translation and hold back the Best- numbers.

A further reason is related to our ranking model which optimizes on Kendall’s  $\tau$  score. Accordingly the output might not be optimal when we evaluate the Top-1 output, but will behave better when we evaluate on the list. This is also in accordance with our aim, which is to enrich the TM

with MT outputs and help the post-editor, instead of choosing the translation for the post-editor.

### 6.3 Comparing the MT, TM and Ranking Outputs

One of the interesting findings from Tables 3 and 4 is that according to the TER edit statistics, the MT outputs generally need a smaller number of edits than the TM and Ranking outputs. This certainly confirms the necessity to integrate MT into today’s TM systems.

However, this fact should not lead to the conclusion that TMs should be replaced by MT completely. First of all, all of our experiments exclude exact TM matches, as those translations will simply be reused and not translated. While this is a realistic setting in the translation industry, it removes all sentences for which the TM works best from our evaluations.

Furthermore, Table 5 shows that the Best-in-Top5 Ranking output performs better than the MT outputs, hence there are TM outputs that lead to smaller number of edits. As  $k$  increases, the ranking model is able to better utilize these outputs.

Finally, in this task we concentrate on ranking useful translations higher, but we are not interested in how useless translations are ranked. Ranking SVM optimizes on the ranking of the whole list, which is slightly different from what we actually require. One option is to use other optimization techniques that can make use of this property to get better Top-k edit statistics for a smaller  $k$ . Another option is obviously to perform regression directly on the number of edits instead of modeling on the ranking. We plan to explore these ideas in future work.

## 7 Conclusions and Future Work

In this paper we present a novel ranking-based model to integrate SMT into a TM system, in order to facilitate the work of post-editors. In such

Table 4: Edit Statistics on Ranked MT and TM Outputs - Top 3

	Insertion	Substitution	Deletion	Shift
TM-Best-in-Top3	0.4241 $\pm$ 0.0250	3.7395 $\pm$ 0.0887	2.9561 $\pm$ 0.0966	0.9738 $\pm$ 0.0505
TM-Mean-Top3	0.6718 $\pm$ 0.0200	5.1428 $\pm$ 0.0559	3.6192 $\pm$ 0.0649	1.3233 $\pm$ 0.0310
MT-Best-in-Top3	0.7696 $\pm$ 0.0351	1.9210 $\pm$ 0.0610	0.7706 $\pm$ 0.0332	1.0842 $\pm$ 0.0545
MT-Mean-Top3	1.1296 $\pm$ 0.0229	2.4405 $\pm$ 0.0368	0.9341 $\pm$ 0.0209	1.3797 $\pm$ 0.0344
Rank-Best-in-Top3	0.8170 $\pm$ 0.0355	2.0744 $\pm$ 0.0608	0.8410 $\pm$ 0.0338	1.0399 $\pm$ 0.0529
Rank-Mean-Top3	1.0942 $\pm$ 0.0234	2.7437 $\pm$ 0.0392	1.0786 $\pm$ 0.0231	1.3309 $\pm$ 0.0334

Table 5: Edit Statistics on Ranked MT and TM Outputs

	Insertion	Substitution	Deletion	Shift
TM-Best-in-Top5	0.4239 $\pm$ 0.0250	3.7319 $\pm$ 0.0885	2.9552 $\pm$ 0.0967	0.9673 $\pm$ 0.0504
TM-Mean-Top5	0.6143 $\pm$ 0.0147	5.5092 $\pm$ 0.0473	3.9451 $\pm$ 0.0521	1.3737 $\pm$ 0.0240
MT-Best-in-Top5	0.7690 $\pm$ 0.0351	1.9163 $\pm$ 0.0610	0.7685 $\pm$ 0.0332	1.0811 $\pm$ 0.0544
MT-Mean-Top5	1.1912 $\pm$ 0.0182	2.5326 $\pm$ 0.0291	0.9487 $\pm$ 0.0165	1.4305 $\pm$ 0.0272
Rank-Best-in-Top5	0.7246 $\pm$ 0.0338*	1.8887 $\pm$ 0.0598	0.7562 $\pm$ 0.0327	0.9705 $\pm$ 0.0515*
Rank-Mean-Top5	1.1173 $\pm$ 0.0181	2.8777 $\pm$ 0.0312	1.1585 $\pm$ 0.0200	1.3675 $\pm$ 0.0260

a model, the user of the TM will be presented with an augmented k-best list, consisting of translations from both the TM and the MT systems, and ranked according to ascending prospective post-editing effort.

From the post-editors' point of view, the TM remains intact. And unlike in the binary translation recommendation, where only one translation recommendation is provided, the ranking model offers k-best post-editing candidates, enabling the user to use more resources when translating. As we do not actually throw away any translation produced from the TM, the assets represented by the TM are preserved and the related estimation of the upper bound cost is still valid.

We extract system independent features from the MT and TM outputs and use Ranking SVMs to train the ranking model, which outperforms both the TM's and MT's k-best list w.r.t. precision at  $k$ , for all  $k$ s.

We also analyze the edit statistics of the integrated k-best output using the TER edit statistics. Our ranking model results in slightly increased number of edits compared to the MT output (apparently held back by a small number of poor TM outputs that are ranked high) for a smaller  $k$ , but requires less edits than both the MT and the TM output for a larger  $k$ .

This work can be extended in a number of ways. Most importantly, We plan to conduct a user study to validate the effectiveness of the method and to gather HTER scores to train a better ranking model. Furthermore, we will try to experiment with learning models that can further reduce the number of edit operations on the top ranked translations. We also plan to improve the adaptability of this method and apply it beyond a specific domain and language pair.

## Acknowledgements

This research is supported by the Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation ([www.cngl.ie](http://www.cngl.ie)) at Dublin City University. We thank Symantec for providing the TM database and the anonymous reviewers for their insightful comments.

## References

- Cortes, Corinna and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.
- Joachims, Thorsten. 2002. Optimizing search engines using clickthrough data. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142, New York, NY, USA.

- Koehn, Philipp., Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL/HLT-2003)*, pages 48 – 54, Edmonton, Alberta, Canada.
- Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions (ACL-2007)*, pages 177–180, Prague, Czech Republic.
- Levenshtein, Vladimir Iosifovich. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710.
- Och, Franz Josef and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, pages 295–302, Philadelphia, PA, USA.
- Och, Franz Josef. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (ACL-2003)*, pages 160–167, Morristown, NJ, USA.
- Shen, Libin, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative reranking for machine translation. In *HLT-NAACL 2004: Main Proceedings*, pages 177–184, Boston, Massachusetts, USA. Association for Computational Linguistics.
- Sikes, Richard. 2007. Fuzzy matching in theory and practice. *Multilingual*, 18(6):39 – 43.
- Simard, Michel and Pierre Isabelle. 2009. Phrase-based machine translation in a computer-assisted translation environment. In *Proceedings of the Twelfth Machine Translation Summit (MT Summit XII)*, pages 120 – 127, Ottawa, Ontario, Canada.
- Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas (AMTA-2006)*, pages 223–231, Cambridge, MA, USA.
- Specia, Lucia, Nicola Cancedda, Marc Dymetman, Marco Turchi, and Nello Cristianini. 2009a. Estimating the sentence-level quality of machine translation systems. In *Proceedings of the 13th Annual Conference of the European Association for Machine Translation (EAMT-2009)*, pages 28 – 35, Barcelona, Spain.
- Specia, Lucia, Craig Saunders, Marco Turchi, Zhuoran Wang, and John Shawe-Taylor. 2009b. Improving the confidence of machine translation quality estimates. In *Proceedings of the Twelfth Machine Translation Summit (MT Summit XII)*, pages 136 – 143, Ottawa, Ontario, Canada.
- Stolcke, Andreas. 2002. SRILM-an extensible language modeling toolkit. In *Proceedings of the Seventh International Conference on Spoken Language Processing*, volume 2, pages 901–904, Denver, CO, USA.
- Ye, Yang, Ming Zhou, and Chin-Yew Lin. 2007. Sentence level machine translation evaluation as a ranking. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 240–247, Prague, Czech Republic.