D6.2b: TMT: Integrated TM/MT Technology II

Yifan He, Yanjun Ma, Josef van Genabith

Distribution: Public

EuroMatrixPlus
Bringing Machine Translation
for European Languages to the User

ICT 231720 Deliverable D6.2b

Project funded by the European Community
under the Seventh Framework Programme for
Research and Technological Development.

Project ref no.	ICT-231720
Project acronym	EuroMatrixPlus
Project full title	Bringing Machine Translation for European Languages to the User
Instrument	STREP
Thematic Priority	ICT-2007.2.2 Cognitive systems, interaction, robotics
Start date / duration	01 March 2009 / 38 Months

Distribution	Public
Contractual date of delivery	March 30, 2010
Actual date of delivery	April 20, 2012
Date of last update	April 27, 2012
Deliverable number	D6.2b
Deliverable title	TMT: Integrated TM/MT Technology II
Type	Report
Status & version	Final
Number of pages	52
Contributing WP(s)	WP6
WP / Task responsible	DCU
Internal reviewer	David Vilar, Christian Federmann
Author(s)	Yifan He, Yanjun Ma, Josef van Genabith
EC project officer	Michel Brochard
Keywords	Machine Translation, Translation Memory, Translation Quality Estimation, Re-ranking

The partners in EuroMatrixPlus are:	DFKI GmbH, Saarbrücken (DFKI) University of Edinburgh (UEDIN) Charles University (CUNI-MFF) Johns Hopkins University (JHU) Fondazione Bruno Kessler (FBK) Université du Maine, Le Mans (LeMans) Dublin City University (DCU) Lucy Software and Services GmbH (Lucy) Central and Eastern European Translation, Prague (CEET) Ludovit Stur Institute of Linguistics, Slovak Academy of Sciences (LSIL) Institute of Information and Communication Technologies, Bulgarian Academy of Sciences (IICT-BAS)

For copies of reports, updates on project activities and other EuroMatrixPlus-related information, contact:

The EuroMatrixPlus Project Co-ordinator
Prof. Dr. Hans Uszkoreit, DFKI GmbH
Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany
uszkoreit@dfki.de
Phone +49 (681) 85775-5282 - Fax +49 (681) 85775-5338

Copies of reports and other material can also be accessed via the project's homepage: http://www.euromatrixplus.net/

Contents

Chapter 1

Introduction

Task 6.2 TMT: Integrated TM/MT Technology focuses on the integration of translation memory (TM) and machine translation (MT) technologies. TMs have been the main-stay technology in the localisation and translation industries. Research on combining TM with MT technologies is therefore an important aspect of the EuroMatrixPlus project and its goal of "Bringing Machine Translation to the User", where the user is a translation professional or potentially a volunteer in more recent collaborative or crowd-souring based localisation models.

In the previous deliverable D6.2a we described our research on tree-alignment-based models of TM/MT integration published as Zhechev and van Genabith (2010a; 2010b), Zhechev (2010).

In this deliverable we present our research on TM/MT combination based on classification and translation quality estimation.

We present recommendation models published in He et al. (2010b) that, given some input to be translated, and given the best MT output and the highest fuzzy match TM segement, decide which of the two to present to the human translator (the post-editor), based on translation quality estimation models using TER to simulate expected human post-editing effort.

We present re-ranking models published in He et al. (2010c) that, given some input to be translated, and given the n-best MT outputs and the m-highest fuzzy match TM segements, use classifiers to merge and re-rank the MT and TM outputs for the human translator (the post-editor), according to translation quality estimation models using TER to simulate expected human post-editing effort. The idea is to provide the human translator with technologies that allow the translator to inspect translation alternatives, while minimising cognitive load due to the fact that "better" translations are much more likely to occur higher up in the merged and re-ranked list.

The recommendation and re-ranking research published in He et al. (2010b) operates on the level of the entire translation segment (e.g. a sentence). We present TM/MT combination models that operate on the sub-segment level: Ma et al. (2011) presents an approach that, given some input to be translated, retrieves similar segements from the TM, and uses the translations of matching sub-sequences between the input and TM fuzzy match to constrain the MT output for the input. In contrast to previous work which used fuzzy match thresholds we use a discriminative learning model find optimal sequences. He et al. (2011) presents an evolution of this model using a rich linguistic feature set improving on the results established for the earlier model.

Below we provide a slightly more extended description of our approaches, including a user study and evaluation of our technology (He et al., 2010a) involving translation professionals. For further detail, the full publications are attached below.

Chapter 2

TM/MT Combination Models

ACL 2010 He et al. (2010b) presents an approach to recommend the better translation between the 1-best MT and 1-best TM output to the translator. We recast translation recommendation as a binary classification (rather than regression) problem using SVMs, perform RBF kernel parameter optimization, employ posterior probability-based confidence estimation to support user-based tuning for precision and recall, and experiment with feature sets involving MT-, TM- and system-independent features.

COLING 2010 He et al. (2010c) presents an approach to re-rank k-best MT- and TM-outputs, so that the translator has access to a larger pool of candidate translations. We solve the translation re-ranking problem using the ranking SVM with system-independent features. We use automatic MT evaluation metrics (TER) to simulate post-editing effort.

AMTA 2010 He et al. (2010a) presents a user study of the effectiveness of the TM/MT translation recommendation model and reports on the reaction of users. Based on the performance statistics and the users' comments, we find that translation recommendation can reduce the workload of professional post-editors and improve the acceptance of MT in the localisation industry.

ACL 2011 Ma et al. (2011) continues our research on the integration of MT and TM systems on the sub-segment level. We constrain the translation of an input sentence using the most similar 'translation example' retrieved from the TM. Differently from previous research (Koehn and Senellart, 2010) which used fuzzy match thresholds, the constraints are imposed using discriminative learning to optimize translation performance. We observe that using this method can not only improve the overall performance of the SMT system in terms of translation quality, but can also help produce more consistent translations (with respect to lexical and terminological choice).

MT Summit XIII He et al. (2011) improves the translation memory (TM)-inspired consistent PB-SMT approach we report in (Ma et al., 2011) using rich linguistic information including lexical, part-of-speech, dependency, and semantic role features to predict whether a TM-derived sub-segment should constrain a PB-SMT translation. Experimental results show that these features help to significantly improve translation quality over the ACL 2011 (Ma et al., 2011) baseline.

Bridging SMT and TM with Translation Recommendation

Yifan He **Yanjun Ma** **Josef van Genabith** **Andy Way**
Centre for Next Generation Localisation
School of Computing
Dublin City University
{yhe,yma,josef,away}@computing.dcu.ie

Abstract

We propose a translation recommendation framework to integrate Statistical Machine Translation (SMT) output with Translation Memory (TM) systems. The framework recommends SMT outputs to a TM user when it predicts that SMT outputs are more suitable for post-editing than the hits provided by the TM. We describe an implementation of this framework using an SVM binary classifier. We exploit methods to fine-tune the classifier and investigate a variety of features of different types. We rely on automatic MT evaluation metrics to approximate human judgements in our experiments. Experimental results show that our system can achieve 0.85 precision at 0.89 recall, excluding exact matches. Furthermore, it is possible for the end-user to achieve a desired balance between precision and recall by adjusting confidence levels.

1 Introduction

Recent years have witnessed rapid developments in statistical machine translation (SMT), with considerable improvements in translation quality. For certain language pairs and applications, automated translations are now beginning to be considered acceptable, especially in domains where abundant parallel corpora exist.

However, these advances are being adopted only slowly and somewhat reluctantly in professional localization and post-editing environments. Post-editors have long relied on translation memories (TMs) as the main technology assisting translation, and are understandably reluctant to give them up. There are several simple reasons for this: 1) TMs are useful; 2) TMs represent considerable effort and investment by a company or (even more so) an individual translator; 3) the fuzzy match score used in TMs offers a good approximation of post-editing effort, which is useful both for translators and translation cost estimation and, 4) current SMT translation confidence estimation measures are not as robust as TM fuzzy match scores and professional translators are thus not ready to replace fuzzy match scores with SMT internal quality measures.

There has been some research to address this issue, see e.g. (Specia et al., 2009a) and (Specia et al., 2009b). However, to date most of the research has focused on better confidence measures for MT, e.g. based on training regression models to perform confidence estimation on scores assigned by post-editors (cf. Section 2).

In this paper, we try to address the problem from a different perspective. Given that most post-editing work is (still) based on TM output, we propose to recommend MT outputs which are better than TM hits to post-editors. In this framework, post-editors still 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.

There are three specific goals we need to achieve within this framework. Firstly, the recommendation should have high precision, otherwise it would be confusing for post-editors and may negatively affect the lower bound of the post-editing effort. Secondly, although we have full access to the SMT system used in this paper, our method should be able to generalize to cases where SMT is treated as a black-box, which is of-

Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 622–630,
Uppsala, Sweden, 11-16 July 2010.

ten the case in the translation industry. Finally, post-editors should be able to easily adjust the recommendation threshold to particular requirements without having to retrain the model.

In our framework, we recast translation recommendation as a binary classification (rather than regression) problem using SVMs, perform RBF kernel parameter optimization, employ posterior probability-based confidence estimation to support user-based tuning for precision and recall, experiment with feature sets involving MT-, TM- and system-independent features, and use automatic MT evaluation metrics to simulate post-editing effort.

The rest of the paper is organized as follows: we first briefly introduce related research in Section 2, and review the classification SVMs in Section 3. We formulate the classification model in Section 4 and present experiments in Section 5. In Section 6, we analyze the post-editing effort approximated by the TER metric (Snover et al., 2006). Section 7 concludes the paper and points out avenues for future research.

2 Related Work

Previous research relating to this work mainly focuses on predicting the MT quality.

The first strand is confidence estimation for MT, initiated by (Ueffing et al., 2003), in which posterior probabilities on the word graph or N-best list are used to estimate the quality of MT outputs. The idea is explored more comprehensively in (Blatz et al., 2004). These estimations are often used to rerank the MT output and to optimize it directly. Extensions of this strand are presented in (Quirk, 2004) and (Ueffing and Ney, 2005). The former experimented with confidence estimation with several different learning algorithms; the latter uses word-level confidence measures to determine whether a particular translation choice should be accepted or rejected in an interactive translation system.

The second strand of research focuses on combining TM information with an SMT system, so that the SMT system can produce better target language output when there is an exact or close match in the TM (Simard and Isabelle, 2009). This line of research is shown to help the performance of MT, but is less relevant to our task in this paper.

A third strand of research tries to incorporate confidence measures into a post-editing environment. To the best of our knowledge, the first paper in this area is (Specia et al., 2009a). Instead of modeling on translation quality (often measured by automatic evaluation scores), this research 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' judgement of the translation quality between 'good' and 'bad', or among three levels of post-editing effort.

Our research is more similar in spirit to the third strand. However, we use outputs and features from the TM explicitly; therefore instead of having to solve a regression problem, we only have to solve a much easier binary prediction problem which can be integrated into TMs in a straightforward manner. Because of this, the precision and recall scores reported in this paper are not directly comparable to those in (Specia et al., 2009b) as the latter are computed on a pure SMT system without a TM in the background.

3 Support Vector Machines for Translation Quality Estimation

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

$$\begin{array}{ll} \min\limits_{w,b,\xi} & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum\limits_{i=1}^{l}\xi_i \\ \text{s. t.} & y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geqslant 1 - \xi_i \\ & \xi_i \geqslant 0 \end{array} \quad (1)$$

where $(\mathbf{x}_i, y_i) \in R^n \times \{+1, -1\}$ are l training instances that are mapped by the function ϕ to a higher dimensional space. $\mathbf{w}$ is the weight vector, ξ is the relaxation variable and $C > 0$ is the penalty parameter.

Solving SVMs is viable using the 'kernel trick': finding a kernel function K in (1) with $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j)$. We perform our experiments with the Radial Basis Function (RBF) kernel, as in (2):

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

When using SVMs with the RBF kernel, we have two free parameters to tune on: the cost parameter C in (1) and the radius parameter γ in (2).

In each of our experimental settings, the parameters C and γ are optimized by a brute-force grid

search. The classification result of each set of parameters is evaluated by cross validation on the training set.

4 Translation Recommendation as Binary Classification

We use an SVM binary classifier to predict the relative quality of the SMT output to make a recommendation. The SVM classifier uses features from the SMT system, the TM and additional linguistic features to estimate whether the SMT output is better than the hit from the TM.

4.1 Problem Formulation

As we treat translation recommendation as a binary classification problem, we have a pair of outputs from TM and MT for each sentence. Ideally the classifier will recommend the output that needs less post-editing effort. As large-scale annotated data is not yet available for this task, we use automatic TER scores (Snover et al., 2006) as the measure for the required post-editing effort. In the future, we hope to train our system on HTER (TER with human targeted references) scores (Snover et al., 2006) once the necessary human annotations are in place. In the meantime we use TER, as TER is shown to have high correlation with HTER.

We label the training examples as in (3):

$$y = \begin{cases} +1 & \text{if } TER(\text{MT}) < TER(\text{TM}) \\ -1 & \text{if } TER(\text{MT}) \geq TER(\text{TM}) \end{cases} \quad (3)$$

Each instance is associated with a set of features from both the MT and TM outputs, which are discussed in more detail in Section 4.3.

4.2 Recommendation Confidence Estimation

In classical settings involving SVMs, confidence levels are represented as margins of binary predictions. However, these margins provide little insight for our application because the numbers are only meaningful when compared to each other. What is more preferable is a probabilistic confidence score (e.g. 90% confidence) which is better understood by post-editors and translators.

We use the techniques proposed by (Platt, 1999) and improved by (Lin et al., 2007) to obtain the posterior probability of a classification, which is used as the confidence score in our system.

Platt's method estimates the posterior probability with a sigmod function, as in (4):

$$Pr(y = 1|\mathbf{x}) \approx P_{A,B}(f) \equiv \frac{1}{1 + exp(Af + B)} \quad (4)$$

where $f = f(\mathbf{x})$ is the decision function of the estimated SVM. A and B are parameters that minimize the cross-entropy error function F on the training data, as in Eq. (5):

$$\min_{z=(A,B)} F(z) = -\sum_{i=1}^{l}(t_i log(p_i) + (1 - t_i) log(1 - p_i)),$$
$$\text{where } p_i = P_{A,B}(f_i), \text{ and } t_i = \begin{cases} \frac{N_+ + 1}{N_+ + 2} \text{ if } y_i = +1 \\ \frac{1}{N_- + 2} \text{ if } y_i = -1 \end{cases} \quad (5)$$

where $z = (A, B)$ is a parameter setting, and N_+ and N_- are the numbers of observed positive and negative examples, respectively, for the label y_i. These numbers are obtained using an internal cross-validation on the training set.

4.3 The Feature Set

We use three types of features in classification: the MT system features, the TM feature and system-independent features.

4.3.1 The MT System Features

These features include those typically used in SMT, namely the phrase-translation model scores, the language model probability, the distance-based reordering score, the lexicalized reordering model scores, and the word penalty.

4.3.2 The TM Feature

The TM feature is the fuzzy match (Sikes, 2007) cost of the TM hit. The calculation of fuzzy match score itself 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 (6), as most of the current implementations are based on edit distance while allowing some additional flexible matching.

$$h_{fm}(t) = \min_{e} \frac{EditDistance(s, e)}{Len(s)} \quad (6)$$

where s is the source side of t, the sentence to translate, and e is the source side of an entry in the TM. For fuzzy match scores F, this fuzzy match cost h_{fm} roughly corresponds to $1-F$. The difference in calculation does not influence classification, and allows direct comparison between a pure TM system and a translation recommendation system in Section 5.4.2.

4.3.3 System-Independent Features

We use several features that are independent of the translation system, which are useful when a third-party translation service is used or the MT system is simply treated as a black-box. These features are source and target side LM scores, pseudo source fuzzy match scores and IBM model 1 scores.

Source-Side Language Model Score and Perplexity. We compute the language model (LM) score and perplexity of the input source sentence on a LM trained on the source-side training data of the SMT system. The inputs that have lower perplexity or higher LM score are more similar to the dataset on which the SMT system is built.

Target-Side Language Model Perplexity. We compute the LM probability and perplexity of the target side as a measure of fluency. Language model perplexity of the MT outputs are calculated, and LM probability is already part of the MT systems scores. LM scores on TM outputs are also computed, though they are not as informative as scores on the MT side, since TM outputs should be grammatically perfect.

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 system 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. We compute this score for both the MT output and the TM hit.

The IBM Model 1 Score. The fuzzy match score does not measure whether the hit could be a correct translation, i.e. it does not take into account the correspondence between the source and target, but rather only the source-side information. For the TM hit, the IBM Model 1 score (Brown et al., 1993) serves as a rough estimation of how good a translation it is on the word level; for the MT output, on the other hand, it is a black-box feature to estimate translation quality when the information from the translation model is not available. We compute bidirectional (source-to-target and target-to-source) model 1 scores on both TM and MT outputs.

5 Experiments

5.1 Experimental Settings

Our raw data set is an English–French translation memory with technical translation from Symantec, consisting of 51K sentence pairs. We randomly selected 43K to train an SMT system and translated the English side of the remaining 8K sentence pairs. The average sentence length of the training set is 13.5 words and the size of the training set is comparable to the (larger) TMs used in the industry. Note that we remove the exact matches in the TM from our dataset, because exact matches will be reused and not presented to the post-editor in a typical TM setting.

As for the SMT system, we use a standard log-linear PB-SMT model (Och and Ney, 2002): GIZA++ implementation of IBM word alignment model 4,[1] the refinement and phrase-extraction heuristics described in (Koehn et al., 2003), minimum-error-rate training (Och, 2003), a 5-gram language model with Kneser-Ney smoothing (Kneser and Ney, 1995) 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 train the SVM classifier using the libSVM (Chang and Lin, 2001) toolkit. The SVM-training and testing is performed on the remaining 8K sentences with 4-fold cross validation. We also report 95% confidence intervals.

The SVM hyper-parameters are tuned using the training data of the first fold in the 4-fold cross validation via a brute force grid search. More specifically, for parameter C in (1) we search in the range $[2^{-5}, 2^{15}]$, and for parameter γ (2) we search in the range $[2^{-15}, 2^{3}]$. The step size is 2 on the exponent.

5.2 The Evaluation Metrics

We measure the quality of the classification by precision and recall. Let A be the set of recommended MT outputs, and B be the set of MT outputs that have lower TER than TM hits. We standardly define precision P, recall R and F-value as in (7):

[1] More specifically, we performed 5 iterations of Model 1, 5 iterations of HMM, 3 iterations of Model 3, and 3 iterations of Model 4.

$$P = \frac{|A \bigcap B|}{|A|}, R = \frac{|A \bigcap B|}{|B|} \text{ and } F = \frac{2PR}{P+R} \quad (7)$$

5.3 Recommendation Results

In Table 1, we report recommendation performance using MT and TM system features (SYS), system features plus system-independent features (ALL:SYS+SI), and system-independent features only (SI).

Table 1: Recommendation Results

	Precision	Recall	F-Score
SYS	82.53±1.17	96.44±0.68	88.95±.56
SI	82.56±1.46	95.83±0.52	88.70±.65
ALL	83.45±1.33	95.56±1.33	89.09±.24

From Table 1, we observe that MT and TM system-internal features are very useful for producing a stable (as indicated by the smaller confidence interval) recommendation system (SYS). Interestingly, only using some simple system-external features as described in Section 4.3.3 can also yield a system with reasonably good performance (SI). We expect that the performance can be further boosted by adding more syntactic and semantic features. Combining all the system-internal and -external features leads to limited gains in Precision and F-score compared to using only system-internal features (SYS) only. This indicates that at the default confidence level, current system-external (resp. system-internal) features can only play a limited role in informing the system when current system-internal (resp. system-external) features are available. We show in Section 5.4.2 that combing both system-internal and -external features can yield higher, more stable precision when adjusting the confidence levels of the classifier. Additionally, the performance of system SI is promising given the fact that we are using only a limited number of simple features, which demonstrates a good prospect of applying our recommendation system to MT systems where we do not have access to their internal features.

5.4 Further Improving Recommendation Precision

Table 1 shows that classification recall is very high, which suggests that precision can still be improved, even though the F-score is not low. Considering that TM is the dominant technology used by post-editors, a recommendation to replace the hit from the TM would require more confidence, i.e. higher precision. Ideally our aim is to obtain a level of 0.9 precision at the cost of some recall, if necessary. We propose two methods to achieve this goal.

5.4.1 Classifier Margins

We experiment with different margins on the training data to tune precision and recall in order to obtain a desired balance. In the basic case, the training example would be marked as in (3). If we label both the training and test sets with this rule, the accuracy of the prediction will be maximized.

We try to achieve higher precision by enforcing a larger bias towards negative examples in the training set so that some borderline positive instances would actually be labeled as negative, and the classifier would have higher precision in the prediction stage as in (8).

$$y = \begin{cases} +1 & \text{if } TER(\text{SMT}) + b < TER(\text{TM}) \\ -1 & \text{if } TER(\text{SMT}) + b \geqslant TER(\text{TM}) \end{cases} \quad (8)$$

We experiment with b in [0, 0.25] using MT system features and TM features. Results are reported in Table 2.

Table 2: Classifier margins

	Precision	Recall
TER+0	83.45±1.33	95.56±1.33
TER+0.05	82.41±1.23	94.41±1.01
TER+0.10	84.53±0.98	88.81±0.89
TER+0.15	85.24±0.91	87.08±2.38
TER+0.20	87.59±0.57	75.86±2.70
TER+0.25	89.29±0.93	66.67±2.53

The highest accuracy and F-value is achieved by $TER + 0$, as all other settings are trained on biased margins. Except for a small drop in $TER{+}0.05$, other configurations all obtain higher precision than $TER + 0$. We note that we can obtain 0.85 precision without a big sacrifice in recall with b=0.15, but for larger improvements on precision, recall will drop more rapidly.

When we use b beyond 0.25, the margin becomes less reliable, as the number of positive examples becomes too small. In particular, this causes the SVM parameters we tune on in the first fold to become less applicable to the other folds. This is one limitation of using biased margins to

obtain high precision. The method presented in Section 5.4.2 is less influenced by this limitation.

5.4.2 Adjusting Confidence Levels

An alternative to using a biased margin is to output a confidence score during prediction and to threshold on the confidence score. It is also possible to add this method to the SVM model trained with a biased margin.

We use the SVM confidence estimation techniques in Section 4.2 to obtain the confidence level of the recommendation, and change the confidence threshold for recommendation when necessary. This also allows us to compare directly against a simple baseline inspired by TM users. In a TM environment, some users simply ignore TM hits below a certain fuzzy match score F (usually from 0.7 to 0.8). This fuzzy match score reflects the confidence of recommending the TM hits. To obtain the confidence of recommending an SMT output, our baseline (FM) uses fuzzy match costs $h_{FM} \approx 1-F$ (cf. Section 4.3.2) for the TM hits as the level of confidence. In other words, the higher the fuzzy match cost of the TM hit is (lower fuzzy match score), the higher the confidence of recommending the SMT output. We compare this baseline with the three settings in Section 5.

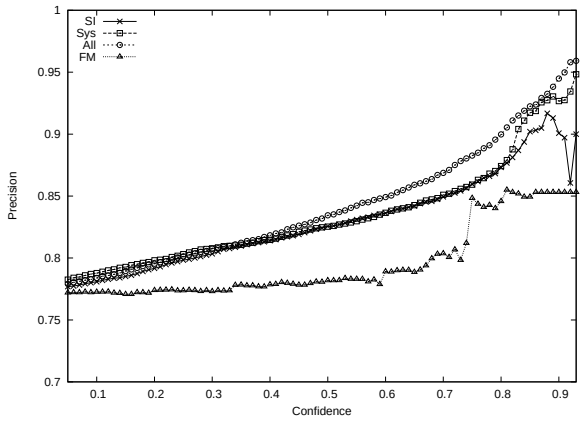

Figure 1: Precision Changes with Confidence Level

Figure 1 shows that the precision curve of FM is low and flat when the fuzzy match costs are low (from 0 to 0.6), indicating that it is unwise to recommend an SMT output when the TM hit has a low fuzzy match cost (corresponding to higher fuzzy match score, from 0.4 to 1). We also observe that the precision of the recommendation receives a boost when the fuzzy match costs for the TM hits are above 0.7 (fuzzy match score lower than 0.3), indicating that SMT output should be recommended when the TM hit has a high fuzzy match cost (low fuzzy match score). With this boost, the precision of the baseline system can reach 0.85, demonstrating that a proper thresholding of fuzzy match scores can be used effectively to discriminate the recommendation of the TM hit from the recommendation of the SMT output.

However, using the TM information only does not always find the easiest-to-edit translation. For example, an excellent SMT output should be recommended even if there exists a good TM hit (e.g. fuzzy match score is 0.7 or more). On the other hand, a misleading SMT output should not be recommended if there exists a poor but useful TM match (e.g. fuzzy match score is 0.2).

Our system is able to tackle these complications as it incorporates features from the MT and the TM systems simultaneously. Figure 1 shows that both the SYS and the ALL setting consistently outperform FM, indicating that our classification scheme can better integrate the MT output into the TM system than this naive baseline.

The SI feature set does not perform well when the confidence level is set above 0.85 (cf. the descending tail of the SI curve in Figure 1). This might indicate that this feature set is not reliable enough to extract the best translations. However, when the requirement on precision is not that high, and the MT-internal features are not available, it would still be desirable to obtain translation recommendations with these black-box features. The difference between SYS and ALL is generally small, but ALL performs steadily better in [0.5, 0,8].

Table 3: Recall at Fixed Precision

	Recall
SYS @85PREC	88.12±1.32
SYS @90PREC	52.73±2.31
SI @85PREC	87.33±1.53
ALL @85PREC	88.57±1.95
ALL @90PREC	51.92±4.28

5.5 Precision Constraints

In Table 3 we also present the recall scores at 0.85 and 0.9 precision for SYS, SI and ALL models to demonstrate our system's performance when there is a hard constraint on precision. Note that our system will return the TM entry when there is an exact match, so the overall precision of the system

is above the precision score we set here in a mature TM environment, as a significant portion of the material to be translated will have a complete match in the TM system.

In Table 3 for MODEL@K, the recall scores are achieved when the prediction precision is better than K with 0.95 confidence. For each model, precision at 0.85 can be obtained without a very big loss on recall. However, if we want to demand further recommendation precision (more conservative in recommending SMT output), the recall level will begin to drop more quickly. If we use only system-independent features (SI), we cannot achieve as high precision as with other models even if we sacrifice more recall.

Based on these results, the users of the TM system can choose between precision and recall according to their own needs. As the threshold does not involve training of the SMT system or the SVM classifier, the user is able to determine this trade-off at runtime.

Table 4: Contribution of Features

	Precision	Recall	F Score
SYS	82.53±1.17	96.44±0.68	88.95±.56
+M1	82.87±1.26	96.23±0.53	89.05±.52
+LM	82.82±1.16	96.20±1.14	89.01±.23
+PS	83.21±1.33	96.61±0.44	89.41±.84

5.6 Contribution of Features

In Section 4.3.3 we suggested three sets of system-independent features: features based on the source- and target-side language model (LM), the IBM Model 1 (M1) and the fuzzy match scores on pseudo-source (PS). We compare the contribution of these features in Table 4.

In sum, all the three sets of system-independent features improve the precision and F-scores of the MT and TM system features. The improvement is not significant, but improvement on every set of system-independent features gives some credit to the capability of SI features, as does the fact that SI features perform close to SYS features in Table 1.

6 Analysis of Post-Editing Effort

A natural question on the integration models is whether the classification reduces the effort of the translators and post-editors: after reading these recommendations, will they translate/edit less than they would otherwise have to? Ideally this question would be answered by human post-editors in a large-scale experimental setting. As we have not yet conducted a manual post-editing experiment, we conduct two sets of analyses, trying to show which type of edits will be required for different recommendation confidence levels. We also present possible methods for human evaluation at the end of this section.

6.1 Edit Statistics

We provide the statistics of the number of edits for each sentence with 0.95 confidence intervals, sorted by TER edit types. Statistics of positive instances in classification (i.e. the instances in which MT output is recommended over the TM hit) are given in Table 5.

When an MT output is recommended, its TM counterpart will require a larger average number of total edits than the MT output, as we expect. If we drill down, however, we also observe that many of the saved edits come from the *Substitution* category, which is the most costly operation from the post-editing perspective. In this case, the recommended MT output actually saves more effort for the editors than what is shown by the TER score. It reflects the fact that TM outputs are not actual translations, and might need heavier editing.

Table 6 shows the statistics of negative instances in classification (i.e. the instances in which MT output is not recommended over the TM hit). In this case, the MT output requires considerably more edits than the TM hits in terms of all four TER edit types, i.e. insertion, substitution, deletion and shift. This reflects the fact that some high quality TM matches can be very useful as a translation.

6.2 Edit Statistics on Recommendations of Higher Confidence

We present the edit statistics of recommendations with higher confidence in Table 7. Comparing Tables 5 and 7, we see that if recommended with higher confidence, the MT output will need substantially less edits than the TM output: e.g. 3.28 fewer substitutions on average.

From the characteristics of the high confidence recommendations, we suspect that these mainly comprise harder to translate (i.e. different from the SMT training set/TM database) sentences, as indicated by the slightly increased edit operations

Table 5: Edit Statistics when Recommending MT Outputs in Classification, confidence=0.5

	Insertion	Substitution	Deletion	Shift
MT	0.9849 ± 0.0408	2.2881 ± 0.0672	0.8686 ± 0.0370	1.2500 ± 0.0598
TM	0.7762 ± 0.0408	4.5841 ± 0.1036	3.1567 ± 0.1120	1.2096 ± 0.0554

Table 6: Edit Statistics when NOT Recommending MT Outputs in Classification, confidence=0.5

	Insertion	Substitution	Deletion	Shift
MT	1.0830 ± 0.1167	2.2885 ± 0.1376	1.0964 ± 0.1137	1.5381 ± 0.1962
TM	0.7554 ± 0.0376	1.5527 ± 0.1584	1.0090 ± 0.1850	0.4731 ± 0.1083

Table 7: Edit Statistics when Recommending MT Outputs in Classification, confidence=0.85

	Insertion	Substitution	Deletion	Shift
MT	1.1665 ± 0.0615	2.7334 ± 0.0969	1.0277 ± 0.0544	1.5549 ± 0.0899
TM	0.8894 ± 0.0594	6.0085 ± 0.1501	4.1770 ± 0.1719	1.6727 ± 0.0846

on the MT side. TM produces much worse edit-candidates for such sentences, as indicated by the numbers in Table 7, since TM does not have the ability to automatically reconstruct an output through the combination of several segments.

6.3 Plan for Human Evaluation

Evaluation with human post-editors is crucial to validate and improve translation recommendation. There are two possible avenues to pursue:

- Test our system on professional post-editors. By providing them with the TM output, the MT output and the one recommended to edit, we can measure the true accuracy of our recommendation, as well as the post-editing time we save for the post-editors;
- Apply the presented method on open domain data and evaluate it using crowdsourcing. It has been shown that crowdsourcing tools, such as the Amazon Mechanical Turk (Callison-Burch, 2009), can help developers to obtain good human judgements on MT output quality both cheaply and quickly. Given that our problem is related to MT quality estimation in nature, it can potentially benefit from such tools as well.

7 Conclusions and Future Work

In this paper we present a classification model to integrate SMT into a TM system, in order to facilitate the work of post-editors. Insodoing we handle the problem of MT quality estimation as binary prediction instead of regression. From the post-editors' perspective, they can continue to work in their familiar TM environment, use the same cost-estimation methods, and at the same time benefit from the power of state-of-the-art MT. We use SVMs to make these predictions, and use grid search to find better RBF kernel parameters.

We explore features from inside the MT system, from the TM, as well as features that make no assumption on the translation model for the binary classification. With these features we make glass-box and black-box predictions. Experiments show that the models can achieve 0.85 precision at a level of 0.89 recall, and even higher precision if we sacrifice more recall. With this guarantee on precision, our method can be used in a TM environment without changing the upper-bound of the related cost estimation.

Finally, we analyze the characteristics of the integrated outputs. We present results to show that, if measured by number, type and content of edits in TER, the recommended sentences produced by the classification model would bring about less post-editing effort than the TM outputs.

This work can be extended in the following ways. Most importantly, it is useful to test the model in user studies, as proposed in Section 6.3. A user study can serve two purposes: 1) it can validate the effectiveness of the method by measuring the amount of edit effort it saves; and 2) the byproduct of the user study – post-edited sentences – can be used to generate HTER scores to train a better recommendation model. Furthermore, we want to experiment and improve on the adaptability of this method, as the current experiment is on 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) at Dublin City University. We thank Symantec for providing the TM database and the anonymous reviewers for their insightful comments.

References

John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. 2004. Confidence estimation for machine translation. In *The 20th International Conference on Computational Linguistics (Coling-2004)*, pages 315 – 321, Geneva, Switzerland.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263 – 311.

Chris Callison-Burch. 2009. Fast, cheap, and creative: Evaluating translation quality using Amazon's Mechanical Turk. In *The 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP-2009)*, pages 286 – 295, Singapore.

Chih-Chung Chang and Chih-Jen Lin, 2001. *LIBSVM: a library for support vector machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273 – 297.

R. Kneser and H. Ney. 1995. Improved backing-off for m-gram language modeling. In *The 1995 International Conference on Acoustics, Speech, and Signal Processing (ICASSP-95)*, pages 181 – 184, Detroit, MI.

Philipp. Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *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.

Philipp Koehn, 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 *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.

Vladimir Iosifovich Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707 – 710.

Hsuan-Tien Lin, Chih-Jen Lin, and Ruby C. Weng. 2007. A note on platt's probabilistic outputs for support vector machines. *Machine Learning*, 68(3):267 – 276.

Franz Josef Och 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.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *The 41st Annual Meeting on Association for Computational Linguistics (ACL-2003)*, pages 160 – 167.

John C. Platt. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers*, pages 61 – 74.

Christopher B. Quirk. 2004. Training a sentence-level machine translation confidence measure. In *The Fourth International Conference on Language Resources and Evaluation (LREC-2004)*, pages 825 – 828, Lisbon, Portugal.

Richard Sikes. 2007. Fuzzy matching in theory and practice. *Multilingual*, 18(6):39 – 43.

Michel Simard and Pierre Isabelle. 2009. Phrase-based machine translation in a computer-assisted translation environment. In *The Twelfth Machine Translation Summit (MT Summit XII)*, pages 120 – 127, Ottawa, Ontario, Canada.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *The 2006 conference of the Association for Machine Translation in the Americas (AMTA-2006)*, pages 223 – 231, Cambridge, MA.

Lucia Specia, Nicola Cancedda, Marc Dymetman, Marco Turchi, and Nello Cristianini. 2009a. Estimating the sentence-level quality of machine translation systems. In *The 13th Annual Conference of the European Association for Machine Translation (EAMT-2009)*, pages 28 – 35, Barcelona, Spain.

Lucia Specia, Craig Saunders, Marco Turchi, Zhuoran Wang, and John Shawe-Taylor. 2009b. Improving the confidence of machine translation quality estimates. In *The Twelfth Machine Translation Summit (MT Summit XII)*, pages 136 – 143, Ottawa, Ontario, Canada.

Andreas Stolcke. 2002. SRILM-an extensible language modeling toolkit. In *The Seventh International Conference on Spoken Language Processing*, volume 2, pages 901 – 904, Denver, CO.

Nicola Ueffing and Hermann Ney. 2005. Application of word-level confidence measures in interactive statistical machine translation. In *The Ninth Annual Conference of the European Association for Machine Translation (EAMT-2005)*, pages 262 – 270, Budapest, Hungary.

Nicola Ueffing, Klaus Macherey, and Hermann Ney. 2003. Confidence measures for statistical machine translation. In *The Ninth Machine Translation Summit (MT Summit IX)*, pages 394 – 401, New Orleans, LA.

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 beginning 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

Coling 2010: Poster Volume, pages 374–382,
Beijing, August 2010

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):

$$\begin{array}{ll} \min\limits_{w,b,\xi} & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum\limits_{i=1}^{l}\xi_i \\ \text{subject to:} & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant 1 - \xi_i \\ & \xi_i \geqslant 0 \end{array} \quad (1)$$

where $(\mathbf{x}_i, y_i) \in R^n \times \{1, -1\}$ are l training instances. $\mathbf{w}$ is the weight vector, ξ 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 τ coefficient with the the gold standard ranking r^*.

Kendall's τ 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 $\mathbf{M} \bigcup \mathbf{T}$ where for each translation output $d \in \mathbf{M} \bigcup \mathbf{T}$, $(d_i, d_j) \in r(s)$ iff $d_i <_{r(s)} d_j$, we can rewrite the ranking constraints as optimization constraints in an SVM, as in Eq. (2).

$$\begin{array}{l} \min\limits_{w,b,\xi} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum\xi \\ \text{subject to:} \\ \forall(d_i, d_j) \in r(s_1) : \mathbf{w}(\Phi(s_1, d_i) - \Phi(s_1, d_j)) \geqslant 1 - \xi_{i,j,1} \\ ... \\ \forall(d_i, d_j) \in r(s_n) : \mathbf{w}(\Phi(s_n, d_i) - \Phi(s_n, d_j)) \geqslant 1 - \xi_{i,j,n} \\ \xi_{i,j,k} \geqslant 0 \end{array} \quad (2)$$

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 τ.

When the instances are not linearly separable, we use a mapping function ϕ 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 τ 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 $\mathbf{M} = \{m_1, m_2, ..., m_n\}$, a TM-produced m-best list $\mathbf{T} = \{t_1, t_2, ..., t_m\}$ for a input sentence s, we define the gold standard using the TER metric (Snover et al., 2006): for each $d \in \mathbf{M} \bigcup \mathbf{T}$, $(d_i, d_j) \in r(s)$ 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

SVM*light*[1] 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 γ in Eq. (3). We optimize C and γ 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

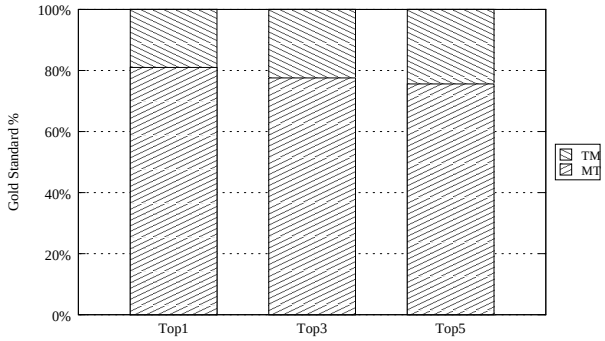

Figure 1: MT and TM's percentage in gold standard

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 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: PREC@k and HIT@k.

Relevant Translations. We borrow the idea of *relevence* 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 $TER(m) \leq TER(t)$. 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 (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 PREC@k= n/k for s. We test 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 HIT@k. For a source sentence s, 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 kth place in the list. HIT@k equals to 1.0 at the end of the list.

We report the mean PREC@k and HIT@k for all s with the 0.95 confidence interval.

5.5 Experimental Results

In Table 1 we report PREC@k and HIT@k for $k = 1..10$. The ranking receives 0.8747 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.

[1] `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 $k = 1...6$, 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 $k = 1...5$ 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 $k = 2...5$. 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 ± 0.0376	4.2461 ± 0.0960	2.9173 ± 0.1027	1.1275 ± 0.0509
MT-Top1	0.9959 ± 0.0385	2.2793 ± 0.0628	0.8940 ± 0.0353	1.2821 ± 0.0575
Rank-Top1	1.0674 ± 0.0414	2.6990 ± 0.0699	1.1246 ± 0.0412	1.2800 ± 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 τ 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 ± 0.0250	3.7395 ± 0.0887	2.9561 ± 0.0966	0.9738 ± 0.0505
TM-Mean-Top3	0.6718 ± 0.0200	5.1428 ± 0.0559	3.6192 ± 0.0649	1.3233 ± 0.0310
MT-Best–in-Top3	0.7696 ± 0.0351	1.9210 ± 0.0610	0.7706 ± 0.0332	1.0842 ± 0.0545
MT-Mean-Top3	1.1296 ± 0.0229	2.4405 ± 0.0368	0.9341 ± 0.0209	1.3797 ± 0.0344
Rank-Best-in-Top3	0.8170 ± 0.0355	2.0744 ± 0.0608	0.8410 ± 0.0338	1.0399 ± 0.0529
Rank-Mean-Top3	1.0942 ± 0.0234	2.7437 ± 0.0392	1.0786 ± 0.0231	1.3309 ± 0.0334

Table 5: Edit Statistics on Ranked MT and TM Outputs

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

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) 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.

Improving the Post-Editing Experience using Translation Recommendation: A User Study

[†]Yifan He **[†]Yanjun Ma** **[‡]Johann Roturier** **[†]Andy Way** **[†]Josef van Genabith**
[†]Centre for Next Generation Localisation, School of Computing, Dublin City University
[‡]Symantec Corporation Ireland
[†]{yhe,yma,away,josef}@computing.dcu.ie
[‡]johann_roturier@symantec.com

Abstract

We report findings from a user study with professional post-editors using a translation recommendation framework (He et al., 2010) to integrate Statistical Machine Translation (SMT) output with Translation Memory (TM) systems. The framework recommends SMT outputs to a TM user when it predicts that SMT outputs are more suitable for post-editing than the hits provided by the TM.

We analyze the effectiveness of the model as well as the reaction of potential users. Based on the performance statistics and the users' comments, we find that translation recommendation can reduce the workload of professional post-editors and improve the acceptance of MT in the localization industry.

1 Introduction

Recent years have witnessed rapid developments in statistical machine translation (SMT), with considerable improvements in translation quality. For certain language pairs and applications, automated translations are now beginning to be considered acceptable, especially in domains where abundant parallel corpora exist.

However, these advances are being adopted only slowly and somewhat reluctantly in professional localization and post-editing environments. Post-editors have long relied on translation memories (TMs) as the main technology to assist translation, and are understandably reluctant to give them up. There are several simple reasons for this: 1) TMs are useful as long as they are maintained; 2) TMs represent considerable effort and investment by a company or (even more so) an individual translator; 3) translators accept that the fuzzy match (Sikes, 2007) score used in TMs offers a good approximation of post-editing effort, which is useful for translation cost estimation; 4) translators are used to working with TMs and using something else could potentially have a negative impact on their productivity, at least in the short term, and 5) current SMT translation confidence estimation measures are not as robust as TM fuzzy match scores, and professional translators are thus not ready to replace fuzzy match scores with SMT internal quality measures.

One solution to promote the recent advances in statistical MT (such as (Koehn et al., 2003)) is to combine the strength of both worlds by integrating SMT with TMs. One of the main challenges of this integration is to establish a measure of confidence regarding the quality of MT output, similar to the TM fuzzy match scores for post-editors.

In our research we follow the approach of (He et al., 2010). Given that most post-editing work is (still) based on TM output, they propose to recommend MT outputs which are better (in terms of estimated post-editing effort) than TM hits to post-editors. In this framework, post-editors still 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 labour.

(He et al., 2010) recast translation recommendation as a binary classification (rather than regression) problem using Support Vector Machines

(SVMs: (Cortes and Vapnik, 1995)) max-margin binary classifiers, perform Radial Basis Function (RBF) kernel parameter optimization to find the optimal meta-parameters for the classifier, employ posterior probability-based confidence estimation to support user-based tuning for precision and recall, experiment with feature sets involving MT-, TM- and system-independent features, and use automatic MT evaluation metrics to simulate post-editing effort. However, the evaluation in (He et al., 2010) suffers from lack of human-annotated data. Instead they use the TER automatic evaluation metric (Snover et al., 2006) to approximate human judgement. Despite the fact that the correlations between automatic evaluation metrics and human judgements are improving, professional post-editors are the ones that hold the final verdict over the quality of MT/TM integration. In order to draw grounded conclusions on the performance of the (He et al., 2010) recommendation framework, it is essential to conduct user studies to show whether or not systems developed using automatic evaluation metrics are confirmed by human judgements. Our experimental results support validation of the approach to approximate post-editing effort using an automatic evaluation metric (TER) in the translation recommendation model in (He et al. 2010): the model obtains more than 90% precision at above 75% recall against the judgements by professional human post-editing.

In this paper, we report the results of the human evaluation along with their behaviour and comments during the evaluation of a translation recommendation system similar to (He et al., 2010). We report findings on whether a high performance recommendation system trained on data annotated according to an automatic evaluation metric tallies with the judgements of professional post-editors.

The rest of the paper is organized as follows: we briefly introduce related research in Section 2, and review the classification model we adopted in Section 3. We describe the methodology of our user study in Section 4, and present an analysis of recommendation performance and user behaviour in Sections 5 and 6, respectively. Section 7 concludes and points out avenues for future research.

2 Related Work

The translation recommendation system we experiment with is an implementation of the translation recommendation model proposed in (He et al., 2010), which we review in more detail in Section 3.

Besides the translation recommendation model, there are several other models that try to combine the merits of TM and MT systems. The first strand is to design MT confidence estimation measures that are friendly to the TM environment, such as (Specia et al., 2009a) and (Specia et al., 2009b), both of which focus on improving confidence measures for MT, e.g. based on training regression models to perform confidence estimation on scores assigned by post-editors.

The second strand of research focuses on combining TM information into an SMT system, so that the SMT system can produce better translations when there is an exact or close match in the TM (Simard and Isabelle, 2009). This line of research is shown to help the performance of MT, but is less relevant to our task in this paper.

Moreover, (Koehn and Haddow, 2009) presents a post-editing environment using information from the phrase-based SMT system Moses (Koehn et al., 2007), instead of the fuzzy match information from TMs. Although all these approaches try to tackle the TM–MT integration task from different perspectives, we concentrate on evaluating the method of (He et al., 2010) in this paper.

The research presented in this paper focuses on aspects of a user study of post-editors working with MT and TMs. In this respect, it is related to (Guerberof, 2009), which compares the post-editing effort required for MT and TM outputs respectively, as well as (Tatsumi, 2009), which studies the correlation between automatic evaluation scores and post-editing effort. Our work differs in that this paper measures how the integration of TM and MT systems can help post-editors, not how post-editors perform using separate TM or MT systems.

3 The Translation Recommendation System

In this section we briefly review the translation recommendation system presented by (He et al., 2010). They use an SVM binary classifier to predict the rel-

ative quality of the SMT output to make a recommendation. The SVM classifier uses features from the SMT system, the TM and additional linguistic features to estimate whether the SMT output is better than the hit from the TM.

3.1 Problem Formulation

(He et al., 2010) treat translation recommendation as a binary classification between TM and SMT outputs, where the classifier recommends the output that is predicted to require less post-editing effort. They use automatic TER scores (Snover et al., 2006) as the measure for the required post-editing effort.

They label the training examples as in (1):

$$y = \begin{cases} +1 & \text{if } TER(\text{MT}) < TER(\text{TM}) \\ -1 & \text{if } TER(\text{MT}) \geq TER(\text{TM}) \end{cases} \quad (1)$$

Each instance is associated with a set of features from both the MT and TM outputs, which are discussed in more detail in Section 3.3.

3.2 Recommendation Confidence Estimation

In classical settings involving SVMs, confidence levels are represented as margins of binary predictions. However, these margins provide little insight for translation applications because the numbers are only meaningful when compared to each other. What is more preferable is a probabilistic confidence score (e.g. 90% confidence) which is better understood by post-editors and translators.

(He et al., 2010) use the techniques proposed by (Platt, 1999) and improved by (Lin et al., 2007) to obtain the posterior probability of a classification as a recommendation confidence score.

Platt's method estimates the posterior probability with a sigmoid function, as in (2):

$$Pr(y = 1|\mathbf{x}) \approx P_{A,B}(f) \equiv \frac{1}{1 + \exp(Af + B)} \quad (2)$$

where $f = f(\mathbf{x})$ is the decision function of the estimated SVM. A and B are parameters that minimize the cross-entropy error function F on the training data, as in (3):

$$\min_{z=(A,B)} F(z) = -\sum_{i=1}^{l} (t_i \log(p_i) + (1 - t_i) \log(1 - p_i)),$$
$$\text{where } p_i = P_{A,B}(f_i), \text{ and } t_i = \begin{cases} \frac{N_+ + 1}{N_+ + 2} \text{ if } y_i = +1 \\ \frac{1}{N_- + 2} \text{ if } y_i = -1 \end{cases} \quad (3)$$

where $z = (A, B)$ is a parameter setting, and N_+ and N_- are the numbers of observed positive and negative examples, respectively, for the label y_i. These numbers are obtained using an internal cross-validation on the training set.

3.3 The Feature Set

(He et al., 2010) use three types of features in classification: the MT system features, the TM feature and system-independent features.

3.3.1 The MT System Features

These features include those typically used in SMT, namely the phrase-translation model scores, the language model probability, the distance-based reordering score, the lexicalized reordering model scores, and the word penalty.

3.3.2 The TM Feature

The TM feature is the fuzzy match (Sikes, 2007) cost of the TM hit. The calculation of fuzzy match score itself is one of the core technologies in TM systems and varies among different vendors. (He et al., 2010) 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 (4), as most of the current implementations are based on edit distance while allowing some additional flexible matching.

$$h_{\text{FM}}(t) = \min_{e} \frac{\text{EditDistance}(s, e)}{\text{Len}(s)} \quad (4)$$

where s is the source side of t, the sentence to translate, and e is the source side of an entry in the TM. For fuzzy match scores F, this fuzzy match cost h_{fm} roughly corresponds to $1 - F$.

3.3.3 System-Independent Features

(He et al., 2010) use several features that are independent of the translation system, which are useful when a third-party translation service is used or the MT system is simply treated as a black-box:

- Source-Side Language Model Score and Perplexity
- Target-Side Language Model Perplexity
- The Pseudo-Source Fuzzy Match Score: they translate the output back to obtain a pseudo source sentence. They compute the fuzzy match score between the original source sentence and this pseudo-source
- The IBM Model 1 (Brown et al., 1993) scores in both directions

4 Evaluation Methodology

We conduct a human evaluation on TM–MT integration with professional post-editors. In this section we introduce the evaluation data we use, the post-editors, the evaluation environment and the questionnaire which we give to the post-editors after they have completed the evaluation.

4.1 Data

Our raw data set is an English–French translation memory which consists of 51K sentence pairs of technical translation from Symantec. We randomly selected 43K to train an SMT system and used this system to translate the English side of the remaining 8K sentence pairs as recommendation candidates. We train SVM translation recommendation models with 4-fold cross validation on these 8K sentence pairs, and randomly select 300 from the cross validation test sets for human evaluation.

More specifically, for the SMT system, we use a standard log-linear PB-SMT model (Och and Ney, 2002): GIZA++ implementation of IBM word alignment model 4, the refinement and phrase-extraction heuristics described in (Koehn et al., 2003), minimum-error-rate training (Och, 2003), a 5-gram language model with Kneser-Ney smoothing (Kneser and Ney, 1995) trained with SRILM (Stolcke, 2002) on the target side of the training data, and Moses (Koehn et al., 2007) to decode.

For the translation recommendation model, we output a confidence level using the method in Section 3.2 and all the features in Section 3.3.

4.2 The Post-editors

Five professional post-editors help us to complete this study. Four of them are full-time post-editors, and one is a part-time post-editor. All of the editors are hired through the localization vendors of the IT security company and have experience on post-editing machine-generated segments (including TM, Rule-based MT or Statistical MT).

4.3 The Evaluation Environment

We design an evaluation environment to present the 300 English segments translated into French using the TM and MT systems to the post-editors. The environment is a web application developed in Python with the Django framework.[1]

Each post-editor is given a username and password to log in the system. After login, there is only one English segment together with its two French translations (from TM and MT) shown on each page. The two French translations are shuffled randomly: so translation 1 can either be output from MT or TM, and depending on the choice in 1, the same for translation 2the remaining option is provided as translation 2. In a production setting, we would present the recommended translation more favorably than the other, but as this experiment tries to evaluate the performance of the TM/MT integration technique, we need to keep it blind. A snapshot of the interface is shown in Figure 1.

The post-editors' operations in the system are recorded with a time stamp in the database, which allows us to analyze the time they spend on each segment. The system allows the users to log in and out of the environment so that their previous work is not lost. They are presented with the last segment they worked on after they log in again.

Each post-editor is provided with an introduction to the task before the experiment begins. Note that the post-editors are asked to choose the sentence that is most suitable for post-editing (which is also emphasized in the introduction to the task). The post-editors are told that even if a French translation does not fully translate the English segment, they may still select it because they would spend less time post-editing it into a grammatical French segment whose meaning would match the that of the English

[1] http://www.djangoproject.com

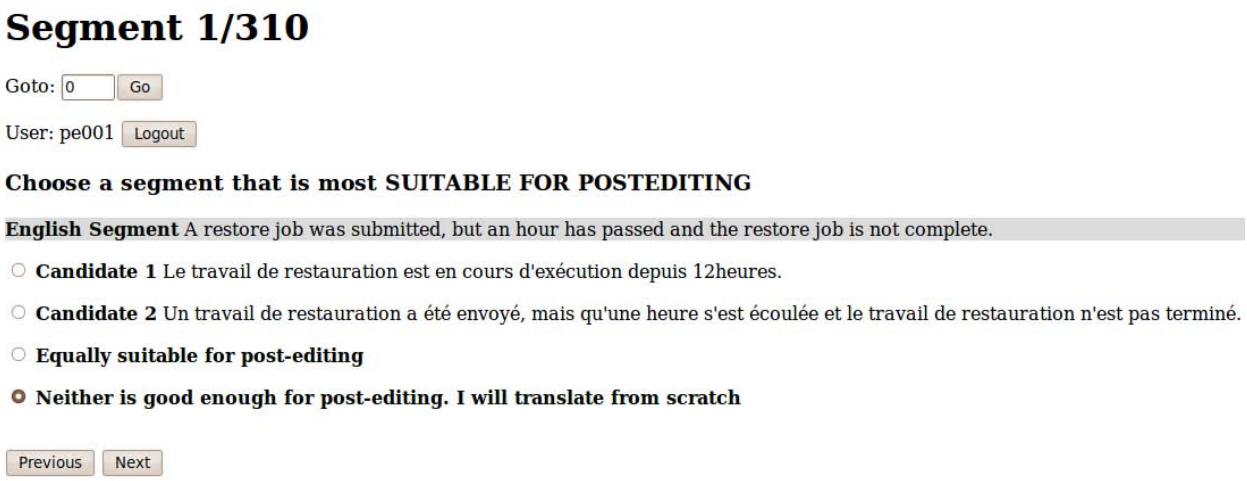

Figure 1: Interface of the Evaluation Environment

segment.

To control data quality and to measure intra-annotator correlation, we pre-select 10 segments from the 300 and make them appear twice in the environment. Therefore the post-editors are actually presented with 310 segments.

4.4 Questionnaire

After they have finished rating the 310 segments, the post-editors are presented with five questionnaire questions:

- Whether they are a full-time post-editor,
- If they are full-time post-editors, how long have they worked as a full-time post-editor,
- The number of words they have translated,
- Whether they have edited MT output professionally,
- What they think of MT (five choices: no idea, very useful, sometimes useful, not useful, and useless).

5 Analysis of Integration Performance

In this section we investigate the effectiveness of the translation recommendation model according to the judgements of professional post-editors. We also compare the results with the result on a gold standard approximated by TER scores to show whether it is valid to use automatic evaluation metric scores (to approximate post-editing effort) instead of human judgement in this task.

5.1 Precision and Recall of Translation Recommendation

We measure the precision and recall of the automatic translation recommendation, using the judgements of individual post-editors as a gold standard. We report the precision and recall numbers in Table 1. The precision can be further improved at the cost of recall, as we set the confidence threshold to 0.75 in Table 2. In these calculations, we discard the segments which the post-editors choose to translate from scratch, as translation recommendation cannot improve the post-editor's productivity in such cases, no matter what it recommends. When the post-editor chooses 'tie', we determine that the TM output should be preserved, in accordance with the conservative gold standard in (He et al., 2010).

Table 1: Precision and Recall of Recommendation, Individual Post-editors, confidence = 0.5

Post-Editor ID	Precision	Recall
PE01	0.8812	0.9223
PE02	0.9315	0.9315
PE03	0.8945	0.9138
PE04	0.9123	0.9369
PE05	0.8734	0.9409

In Table 1, the automatic recommendation obtains over 0.9 recall according to all post-editors. The pre-

Table 2: Precision and Recall of Recommendation, Individual Post-editors, confidence = 0.75

Post-Editor ID	Precision	Recall
PE01	0.9379	0.7824
PE02	0.9643	0.7621
PE03	0.9415	0.7629
PE04	0.9500	0.7703
PE05	0.9153	0.7864

cision of recommendation is always above 0.87. Table 2 shows that when the post-editors require more recommendation confidence, the translation recommendation can always obtain 0.9 precision at the cost of reducing recall. With these results on recommendation precision, there is a rather strong guarantee that the integrated MT-TM system will not waste the assets in the TM system and will not change the upperbound of related cost estimation, even at the sentence level, because the recommended SMT outputs are, in fact, more suitable for post-editing from the post-editors' perspective.

5.2 Precision and Recall on Consensus Preferences

The localization industry might expect even stronger confidence in the recommendation, so we measure recommendation precision on the segments where there is a consensus among the post-editors that the MT output should be used to post-edit.

To reflect consensus, we first discard the segments which the majority of the post-editors (more than 3 in this experiment) choose to post-edit from scratch. For the rest of the segments, we consider that MT output should be recommended, if N post-editors prefer to post-edit the MT output. Otherwise, we consider that the TM output should be recommended.

We report the precision and recall numbers on a series of confidence thresholds for $N = 3$ and $N = 4$ post-editors in Tables 3 and 4, respectively.

Table 3 shows that if we consider the consensus among 3 post-editors, precision is still high. Besides the capability of the recommendation system, there is also the reason that there are actually a larger number of segments to be recommended when we consider the consensus among 3 post-editors, rather than 1 post-editor. When we analyze Table 4, precision begins to drop. The reason for this is that this is an inherently more difficult task, and that the post-editor PE01 chooses to edit a larger number of segments from scratch (cf. Section 6.1), which renders it more difficult for the remaining post-editors to reach a consensus for $N = 4$.

Table 3: Precision and Recall of Recommendation, Consensus Preferences of $N = 3$ Post-Editors

Threshold	Precision	Recall
0.5	0.9110	0.9348
0.6	0.9412	0.9043
0.7	0.9606	0.8478
0.8	0.9689	0.6783
0.85	0.9695	0.5522

Table 4: Precision and Recall of Recommendation, Consensus Preferences of $N = 4$ Post-Editors

Threshold	Precision	Recall
0.5	0.8263	0.9420
0.6	0.8507	0.9082
0.7	0.8768	0.8599
0.8	0.8944	0.6957
0.85	0.8931	0.5652

5.3 The TER score and the Preference of Post-Editors

We measure the TER score of the TM and MT outputs, and sort them according to the post-editors' preferences in Table 5. The TER score is an edit-distance based metric that calculates the number of insertions, deletions, substitutions and shifts required to transform an MT output to a reference sentence, and is therefore expected to be a reasonable automatic metric to approximate post-editing effort. We report the results in Table 5, where the scores are averaged among the five post-editors.

Table 5: TER Scores Sorted by Preference

	TM	MT	Tie	Scratch
TM Output	25.00	57.37	19.16	70.33
MT Output	31.85	25.90	20.93	41.74

In Table 5, TER scores are shown to correlate well with post-editors' preferences: when the post-editor prefers MT, the MT output obtains a lower TER score, and vice versa. This validates the method

of (He et al., 2010), where the TER score is used to generate a gold standard for the translation recommendation system. The TER scores also demonstrate that the sentences which the users would translate from scratch are more difficult to translate in nature than the rest, as is shown by the deterioration of more than 15 TER points compared to the MT-output.

5.4 Comparison with a TER-Approximated Gold Standard

We present the precision numbers at recommendation confidence [0.5, 0.85] in Figure 2. Series PE01 – PE05 use the judgement of the corresponding post-editor as the gold standard; series CONSENSUS_3 and CONSENSUS_4 use the consensus of 3 or 4 post-editors as the gold standard; series TER uses the gold standard approximated by TER scores, as in (1). By presenting results on human-annotated and metric-approximated gold standards head-to-head, we are able to see the relationship between these gold standards.

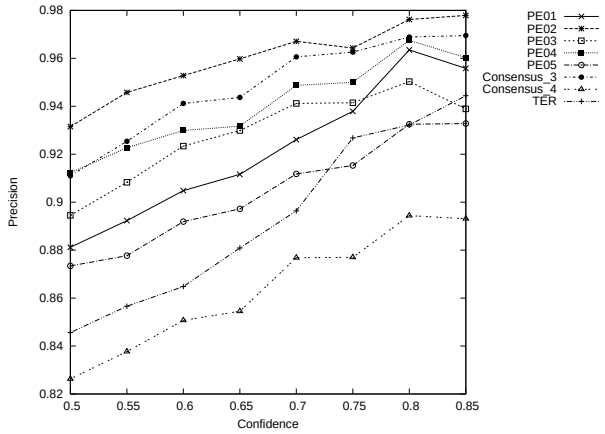

Figure 2: Recommendation Precision According to Human-Annotated and TER-Approximated Gold Standards

In Figure 2, we find that although the post-editors have different preferences regarding MT and TM outputs (i.e. some reuse MT outputs more than others), the trend of precision on the variation of confidence levels remains similar among the post-editors, and also applies to the TER-approximated gold standard. This again validates the approach of (He et al., 2010), which uses TER scores to approximate human judgements to prepare the training data and perform evaluation. Note that when calculating precision, the denominator is the total number of segments recommended by the recommendation model, no matter whether the post-editors have consensus judgements on them or not. If we limit the denominator to the number of segments where post-editors do reach a consensus judgement (on whether using the MT or the TM output), the precision will be 0.9641 for CONSENSUS_3 and 0.9848 for CONSENSUS_4.

6 Analysis of User Behaviour

Besides the performance of recommendation, we are also interested in the users' reaction to the TM-MT scheme using this system, as well as what they think about the TM and MT technologies. We report statistics of their behaviour along with their ideas and comments on TM and MT.

6.1 Experience of Post-Editors

We list the years of experience as translators of the post-editors along with the number of sentences they prefer to translate from scratch in our experiment in Table 6, because the latter is an indication of the willingness to reuse a computer-generated translation. We also present the number of MT outputs (out of 300) selected by post-editors to work on.

Table 6: Participants' Experience and Preference

Post-Editor ID	Years	Scratch	MT
PE01	5	59	193
PE02	3	11	248
PE03	12	22	232
PE04	8	33	222
PE05	part-time	23	220

The results show that the willingness to reuse automatic outputs varies considerably among post-editors. PE01 is willing to translate one-fifth of the sentences from scratch in this experiment, which is more than five-times the number of PE02. This preference does not correlate well with the years of experience, suggesting that this is more related to the particular habits of post-editors, rather than to their experience in the industry. The result also shows that all the post-editors select more MT outputs to post-edit than the other options.

6.2 Inter-annotator Agreement

To gauge the validity of human evaluation results, we computed the inter-rater agreement measured by Fleiss' Kappa coefficient (Fleiss, 1981) which can assess the agreement between multiple raters as opposed to Cohen's Kappa coefficient (Cohen, 1960) which works with two raters.

Fleiss' Kappa coefficient for our five post-editors is 0.464 ± 0.024, indicating a moderate agreement. We also obtained Fleiss' Kappa coefficient for each category as shown in Table 7. From this table, we can observe moderate agreements among post-editors in selecting TM or MT output as the most suitable for post-editing. There is also a moderate agreement in making their decision to translate from scratch. However, there is only a fair agreement in determining whether TM and MT outputs are equally good for post-editing ("Tie").

Table 7: Annotator agreement for each category

Category	Kappa
TM	0.519
MT	0.516
Tie	0.285
Scratch	0.426

6.3 Intra-annotator Agreement

We have ten duplicate samples in our evaluation intended to measure the level of intrinsic agreement for each post-editor. Both percentage of agreement and Cohen's Kappa are calculated as shown in Table 9. From this table, we can observe that all five post-editors achieved almost perfect intrinsic agreement, indicating that the evaluation results are highly reliable.

Table 8: Intra-annotator Agreement

Post-Editor ID	Agreement	Kappa
PE01	90%	0.87
PE02	100%	1.0
PE03	90%	0.87
PE04	80%	0.73
PE05	90%	0.87

6.4 Correlation between Sentence Length and Evaluation Time

Our evaluation interface is capable of logging the time spent by the post-editors in evaluating each sentence. One may expect that post-editors may spend more time in evaluating longer sentences and less time evaluating shorter sentences. We calculated Pearson's product moment correlation between the evaluation time and sentence length as shown in Table 9. The results appear to be inconclusive: we observe a high correlation between the evaluation time and sentence length for PE02 and PE05; however, for the other three post-editors, there is a low correlation. These inconclusive results can partly be attributed to the fact that we did not compel the post-editors to conduct their evaluation in one session. We expect to achieve more conclusive results in future work, which would happen in a real working post-editing environment.

Table 9: Pearson's Product Moment Correlation

Post-Editor ID	PMCC (r)	r-square
PE01	0.2246	0.0505
PE02	0.6957	0.4840
PE03	0.3916	0.1534
PE04	0.0746	0.0056
PE05	0.4907	0.2408
Average	0.2274	0.0517

6.5 Post-editors' Comments on MT and TM

We requested post-editors to comment on their attitude to MT and TM. In our questionnaire, all post-editors claim that they have post-edited MT outputs and think that MT is sometimes useful, which might represent the current state of MT penetration in the localization industry.

However, the more interesting comment comes from one of our post-editors in private communication. Although the post-editor does not know which of the two candidates we present in the evaluation interface is from the MT system, he claims after completing the evaluation that he has found that the TM outputs are more suitable for post-editing, although in fact every post-editor prefers MT outputs in the experiment (cf. Table 6).

This comment is revealing for two reasons. First

of all, the post-editor obviously mistakes MT outputs for TM outputs, which indicates that in this closed-domain setting mainly composed of simple short sentences, a state-of-the-art phrase-based SMT system is able to produce outputs that are not only correct on the word-to-word level, but also grammatically acceptable enough to be recognized as human translations in the TM, and therefore that the SMT output can be smoothly integrated into the TM environment.

Furthermore, the comment also shows how much the post-editors subconsciously trust the TM. This may be an explanation for the relatively low acceptance of MT technology in the localization industry, and demonstrates the need for TM–MT integration techniques such as translation recommendation.

7 Conclusions and Future Work

In this paper, we evaluate the effectiveness of translation recommendation (He et al., 2010) in the context of TM–MT integration with professional post-editors.

We find that a translation recommendation model trained on automatic evaluation metric scores can obtain a precision above 0.9 and a recall above 0.75 with proper thresholds according to each of the post-editors. The model shows precision above 0.8 when we evaluate against the consensus of post-editors. This supports validation of the method of (He et al., 2010) which uses automatic evaluation metrics to approximate actual post-editing effort.

From the analysis of user behaviour, we note that the users show consistency in their judgements according to both the inter-annotator agreement and the intra-annotator agreement. The recommended MT outputs are incorrectly recognized as TM outputs by one post-editor, which shows both the potential and the necessity for TM–MT integration.

This work can be extended in several ways. First of all, in this paper we concentrated on proprietary data and professional post-editors, according to the major paradigm in the localization industry. However, at the same time this limits the number of annotators we can hire, as well as the types of evaluations we can perform. We can obtain more comprehensive results by experimenting on open-domain data sets, and applying crowd-sourcing technologies such as Amazon Mechanical Turk[2] (Callison-Burch, 2009).

Secondly, during the evaluation we were able to collect a number of human judgements for training a new translation recommendation system. We plan to train a new recommendation model and to compare the difference with models trained on automatic metric scores, when we have collected more human-annotated data.

Finally, this experiment can also be extended by measuring the actual post-editing time instead of the judgement time, which can lead to a more precise approximation of reduced post-editing effort when using translation recommendation to integrate MT outputs into a TM system.

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) at Dublin City University. We thank the anonymous reviewers for their insightful comments.

References

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263 – 311.

Chris Callison-Burch. 2009. Fast, cheap, and creative: Evaluating translation quality using Amazon's Mechanical Turk. In *The 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP-2009)*, pages 286 – 295, Singapore.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. 20(1):37–46.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

Joseph L. Fleiss. 1981. *Statistical methods for rates and proportions*. John Wiley & Sons.

Ana Guerberof. 2009. Productivity and quality in mt post-editing. In *MT Summit XII - Workshop: Beyond Translation Memories: New Tools for Translators MT*, Ottawa, Ontario, Canada.

Yifan He, Yanjun Ma, Josef van Genabith, and Andy Way. 2010. Bridging TM and SMT with translation recommendation. In *ACL 2010*, Uppsala, Sweden.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *The 1995 International Conference on Acoustics, Speech,*

[2] `https://www.mturk.com`

and Signal Processing (ICASSP-95), pages 181 – 184, Detroit, MI.

Philipp Koehn and Barry Haddow. 2009. Interactive assistance to human translators using statistical machine translation methods. In *The Twelfth Machine Translation Summit (MT-Summit XII)*, pages 73 – 80, Ottawa, Ontario, Canada.

Philipp. Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *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.

Philipp Koehn, 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 *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.

Vladimir Iosifovich Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710.

Hsuan-Tien Lin, Chih-Jen Lin, and Ruby C. Weng. 2007. A note on Platt's probabilistic outputs for support vector machines. *Machine Learning*, 68(3):267–276.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *The 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, pages 295–302, Philadelphia, PA.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *The 41st Annual Meeting on Association for Computational Linguistics (ACL-2003)*, pages 160–167, Sapporo, Japan.

John C. Platt. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers*, pages 61–74.

Richard Sikes. 2007. Fuzzy matching in theory and practice. *Multilingual*, 18(6):39 – 43.

Michel Simard and Pierre Isabelle. 2009. Phrase-based machine translation in a computer-assisted translation environment. In *The Twelfth Machine Translation Summit (MT Summit XII)*, pages 120 – 127, Ottawa, Ontario, Canada.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *The 2006 Conference of the Association for Machine Translation in the Americas (AMTA-2006)*, pages 223–231, Cambridge, MA.

Lucia Specia, Nicola Cancedda, Marc Dymetman, Marco Turchi, and Nello Cristianini. 2009a. Estimating the sentence-level quality of machine translation systems. In *The 13th Annual Conference of the European Association for Machine Translation (EAMT-2009)*, pages 28 – 35, Barcelona, Spain.

Lucia Specia, Craig Saunders, Marco Turchi, Zhuoran Wang, and John Shawe-Taylor. 2009b. Improving the confidence of machine translation quality estimates. In *The Twelfth Machine Translation Summit (MT Summit XII)*, pages 136 – 143, Ottawa, Ontario, Canada.

Andreas Stolcke. 2002. SRILM-an extensible language modeling toolkit. In *The Seventh International Conference on Spoken Language Processing*, volume 2, pages 901–904, Denver, CO.

Midori Tatsumi. 2009. Correlation between automatic evaluation metric scores, post-editing speed, and some other factors. In *The Twelfth Machine Translation Summit (MT-Summit XII)*, pages 332 – 339, Ottawa, Ontario, Canada.

Consistent Translation using Discriminative Learning: A Translation Memory-inspired Approach*

Yanjun Ma[†] **Yifan He**[‡] **Andy Way**[‡] **Josef van Genabith**[‡]
[†] Baidu Inc., Beijing, China
yma@baidu.com
[‡]Centre for Next Generation Localisation
School of Computing, Dublin City University
{yhe,away,josef}@computing.dcu.ie

Abstract

We present a discriminative learning method to improve the consistency of translations in phrase-based Statistical Machine Translation (SMT) systems. Our method is inspired by Translation Memory (TM) systems which are widely used by human translators in industrial settings. We constrain the translation of an input sentence using the most similar 'translation example' retrieved from the TM. Differently from previous research which used simple fuzzy match thresholds, these constraints are imposed using discriminative learning to optimise the translation performance. We observe that using this method can benefit the SMT system by not only producing consistent translations, but also improved translation outputs. We report a 0.9 point improvement in terms of BLEU score on English–Chinese technical documents.

1 Introduction

Translation consistency is an important factor for large-scale translation, especially for domain-specific translations in an industrial environment. For example, in the translation of technical documents, lexical as well as structural consistency is essential to produce a fluent target-language sentence. Moreover, even in the case of translation errors, consistency in the errors (e.g. repetitive error patterns) are easier to diagnose and subsequently correct by translators.

In phrase-based SMT, translation models and language models are automatically learned and/or generalised from the training data, and a translation is produced by maximising a weighted combination of these models. Given that global contextual information is not normally incorporated, and that training data is usually noisy in nature, there is no guarantee that an SMT system can produce translations in a consistent manner.

On the other hand, TM systems – widely used by translators in industrial environments for enterprise localisation by translators – can shed some light on mitigating this limitation. TM systems can assist translators by retrieving and displaying previously translated similar 'example' sentences (displayed as source-target pairs, widely called 'fuzzy matches' in the localisation industry (Sikes, 2007)). In TM systems, fuzzy matches are retrieved by calculating the similarity or the so-called 'fuzzy match score' (ranging from 0 to 1 with 0 indicating no matches and 1 indicating a full match) between the input sentence and sentences in the source side of the translation memory.

When presented with fuzzy matches, translators can then avail of useful chunks in previous translations while composing the translation of a new sentence. Most translators only consider a few sentences that are most similar to the current input sentence; this process can inherently improve the consistency of translation, given that the new translations produced by translators are likely to be similar to the target side of the fuzzy match they have consulted.

Previous research as discussed in detail in Sec-

*This work was done while the first author was in the Centre for Next Generation Localisation at Dublin City University.

Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, pages 1239–1248,
Portland, Oregon, June 19-24, 2011.

tion 2 has focused on using fuzzy match score as a threshold when using the target side of the fuzzy matches to constrain the translation of the input sentence. In our approach, we use a more fine-grained discriminative learning method to determine whether the target side of the fuzzy matches should be used as a constraint in translating the input sentence. We demonstrate that our method can consistently improve translation quality.

The rest of the paper is organized as follows: we begin by briefly introducing related research in Section 2. We present our discriminative learning method for consistent translation in Section 3 and our feature design in Section 4. We report the experimental results in Section 5 and conclude the paper and point out avenues for future research in Section 6.

2 Related Research

Despite the fact that TM and MT integration has long existed as a major challenge in the localisation industry, it has only recently received attention in main-stream MT research. One can loosely combine TM and MT at sentence (called segments in TMs) level by choosing one of them (or both) to recommend to the translators using automatic classifiers (He et al., 2010), or simply using fuzzy match score or MT confidence measures (Specia et al., 2009).

One can also tightly integrate TM with MT at the sub-sentence level. The basic idea is as follows: given a source sentence to translate, we firstly use a TM system to retrieve the most similar 'example' source sentences together with their translations. If matched chunks between input sentence and fuzzy matches can be detected, we can directly re-use the corresponding parts of the translation in the fuzzy matches, and use an MT system to translate the remaining chunks.

As a matter of fact, implementing this idea is pretty straightforward: a TM system can easily detect the word alignment between the input sentence and the source side of the fuzzy match by retracing the paths used in calculating the fuzzy match score. To obtain the translation for the matched chunks, we just require the word alignment between source and target TM matches, which can be addressed using state-of-the-art word alignment techniques. More importantly, albeit not explicitly spelled out in previous work, this method can potentially increase the consistency of translation, as the translation of new input sentences is closely informed and guided (or constrained) by previously translated sentences.

There are several different ways of using the translation information derived from fuzzy matches, with the following two being the most widely adopted: 1) to add these translations into a phrase table as in (Biçici and Dymetman, 2008; Simard and Isabelle, 2009), or 2) to mark up the input sentence using the relevant chunk translations in the fuzzy match, and to use an MT system to translate the parts that are not marked up, as in (Smith and Clark, 2009; Koehn and Senellart, 2010; Zhechev and van Genabith, 2010). It is worth mentioning that translation consistency was not explicitly regarded as their primary motivation in this previous work. Our research follows the direction of the second strand given that consistency can no longer be guaranteed by constructing another phrase table.

However, to categorically reuse the translations of matched chunks without any differentiation could generate inferior translations given the fact that the context of these matched chunks in the input sentence could be completely different from the source side of the fuzzy match. To address this problem, both (Koehn and Senellart, 2010) and (Zhechev and van Genabith, 2010) used fuzzy match score as a threshold to determine whether to reuse the translations of the matched chunks. For example, (Koehn and Senellart, 2010) showed that reusing these translations as large rules in a hierarchical system (Chiang, 2005) can be beneficial when the fuzzy match score is above 70%, while (Zhechev and van Genabith, 2010) reported that it is only beneficial to a phrase-based system when the fuzzy match score is above 90%.

Despite being an informative measure, using fuzzy match score as a threshold has a number of limitations. Given the fact that fuzzy match score is normally calculated based on Edit Distance (Levenshtein, 1966), a low score does not necessarily imply that the fuzzy match is harmful when used to constrain an input sentence. For example, in longer sentences where fuzzy match scores tend to be low, some chunks and the corresponding translations within the sentences can still be useful. On

the other hand, a high score cannot fully guarantee the usefulness of a particular translation. We address this problem using discriminative learning.

3 Constrained Translation with Discriminative Learning

3.1 Formulation of the Problem

Given a sentence $\mathbf{e}$ to translate, we retrieve the most similar sentence $\mathbf{e}'$ from the translation memory associated with target translation $\mathbf{f}'$. The m common "phrases" $\bar{\mathbf{e}}_{\mathbf{1}}^{\mathbf{m}}$ between $\mathbf{e}$ and $\mathbf{e}'$ can be identified. Given the word alignment information between $\mathbf{e}'$ and $\mathbf{f}'$, one can easily obtain the corresponding translations $\bar{\mathbf{f}}'^{\mathbf{m}}_{\mathbf{1}}$ for each of the phrases in $\bar{\mathbf{e}}_{\mathbf{1}}^{\mathbf{m}}$. This process can derive a number of "phrase pairs" $< \bar{e}_m, \bar{f}'_m >$, which can be used to specify the translations of the matched phrases in the input sentence. The remaining words without specified translations will be translated by an MT system.

For example, given an input sentence $e_1 e_2 \cdots e_i e_{i+1} \cdots e_I$, and a phrase pair $< \bar{e}, \bar{f}' >$, $\bar{e} = e_i e_{i+1}$, $\bar{f}' = f'_j f'_{j+1}$ derived from the fuzzy match, we can mark up the input sentence as:

$e_1 e_2 \cdots$ <tm="$f'_j f'_{j+1}$"> $e_i e_{i+1}$ </tm> $\cdots e_I$.

Our method to constrain the translations using TM fuzzy matches is similar to (Koehn and Senellart, 2010), except that the word alignment between $\mathbf{e}'$ and $\mathbf{f}'$ is the intersection of bidirectional GIZA++ (Och and Ney, 2003) posterior alignments. We use the intersected word alignment to minimise the noise introduced by word alignment of only one direction in marking up the input sentence.

3.2 Discriminative Learning

Whether the translation information from the fuzzy matches should be used or not (i.e. whether the input sentence should be marked up) is determined using a discriminative learning procedure. The translation information refers to the "phrase pairs" derived using the method described in Section 3.1. We cast this problem as a binary classification problem.

3.2.1 Support Vector Machines

SVMs (Cortes and Vapnik, 1995) are binary classifiers that classify an input instance based on decision rules which minimise the regularised error function in (1):

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i \\ \text{s. t.} \quad & y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geqslant 1 - \xi_i \\ & \xi_i \geqslant 0 \end{aligned} \tag{1}$$

where $(\mathbf{x}_i, y_i) \in R^n \times \{+1, -1\}$ are l training instances that are mapped by the function ϕ to a higher dimensional space. $\mathbf{w}$ is the weight vector, ξ is the relaxation variable and $C > 0$ is the penalty parameter.

Solving SVMs is viable using a kernel function K in (1) with $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j)$. We perform our experiments with the Radial Basis Function (RBF) kernel, as in (2):

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

When using SVMs with the RBF kernel, we have two free parameters to tune on: the cost parameter C in (1) and the radius parameter γ in (2).

In each of our experimental settings, the parameters C and γ are optimised by a brute-force grid search. The classification result of each set of parameters is evaluated by cross validation on the training set.

The SVM classifier will thus be able to predict the usefulness of the TM fuzzy match, and determine whether the input sentence should be marked up using relevant phrase pairs derived from the fuzzy match before sending it to the SMT system for translation. The classifier uses features such as the fuzzy match score, the phrase and lexical translation probabilities of these relevant phrase pairs, and additional syntactic dependency features. Ideally the classifier will decide to mark up the input sentence if the translations of the marked phrases are accurate when taken contextual information into account. As large-scale manually annotated data is not available for this task, we use automatic TER scores (Snover et al., 2006) as the measure for training data annotation.

We label the training examples as in (3):

$$y = \begin{cases} +1 & \text{if } TER(\text{w. markup}) < TER(\text{w/o markup}) \\ -1 & \text{if } TER(\text{w/o markup}) \geq TER(\text{w. markup}) \end{cases} \tag{3}$$

Each instance is associated with a set of features which are discussed in more detail in Section 4.

3.2.2 Classification Confidence Estimation

We use the techniques proposed by (Platt, 1999) and improved by (Lin et al., 2007) to convert classification margin to posterior probability, so that we can easily threshold our classifier (cf. Section 5.4.2).

Platt's method estimates the posterior probability with a sigmoid function, as in (4):

$$Pr(y=1|\mathbf{x}) \approx P_{A,B}(f) \equiv \frac{1}{1+exp(Af+B)} \quad (4)$$

where $f = f(\mathbf{x})$ is the decision function of the estimated SVM. A and B are parameters that minimise the cross-entropy error function F on the training data, as in (5):

$$\min_{z=(A,B)} F(z) = -\sum_{i=1}^{l}(t_i log(p_i) + (1-t_i)log(1-p_i)),$$
$$\text{where } p_i = P_{A,B}(f_i), \text{and } t_i = \begin{cases} \frac{N_+ +1}{N_+ +2} \text{ if } y_i = +1 \\ \frac{1}{N_- +2} \text{ if } y_i = -1 \end{cases} \quad (5)$$

where $z = (A, B)$ is a parameter setting, and N_+ and N_- are the numbers of observed positive and negative examples, respectively, for the label y_i. These numbers are obtained using an internal cross-validation on the training set.

4 Feature Set

The features used to train the discriminative classifier, all on the sentence level, are described in the following sections.

4.1 The TM Feature

The TM feature is the fuzzy match score, which indicates the overall similarity between the input sentence and the source side of the TM output. If the input sentence is similar to the source side of the matching segment, it is more likely that the matching segment can be used to mark up the input sentence.

The calculation of the fuzzy match score itself 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, normalised by the length of the source as in (6), as most of the current implementations are based on edit distance while allowing some additional flexible matching.

$$h_{fm}(\mathbf{e}) = \min_{\mathbf{s}} \frac{EditDistance(\mathbf{e},\mathbf{s})}{Len(\mathbf{e})} \quad (6)$$

where $\mathbf{e}$ is the sentence to translate, and $\mathbf{s}$ is the source side of an entry in the TM. For fuzzy match scores F, h_{fm} roughly corresponds to $1 - F$.

4.2 Translation Features

We use four features related to translation probabilities, i.e. the phrase translation and lexical probabilities for the phrase pairs $< \bar{e}_m, \bar{f}'_m >$ derived using the method in Section 3.1. Specifically, we use the phrase translation probabilities $p(\bar{f}'_m|\bar{e}_m)$ and $p(\bar{e}_m|\bar{f}'_m)$, as well as the lexical translation probabilities $p_{lex}(\bar{f}'_m|\bar{e}_m)$ and $p_{lex}(\bar{e}_m|\bar{f}'_m)$ as calculated in (Koehn et al., 2003). In cases where multiple phrase pairs are used to mark up one single input sentence $\mathbf{e}$, we use a unified score for each of the four features, which is an average over the corresponding feature in each phrase pair. The intuition behind these features is as follows: phrase pairs $< \bar{e}_m, \bar{f}'_m >$ derived from the fuzzy match should also be reliable with respect to statistically produced models.

We also have a count feature, i.e. the number of phrases used to mark up the input sentence, and a binary feature, i.e. whether the phrase table contains at least one phrase pair $< \bar{e}_m, \bar{f}'_m >$ that is used to mark up the input sentence.

4.3 Dependency Features

Given the phrase pairs $< \bar{e}_m, \bar{f}'_m >$ derived from the fuzzy match, and used to translate the corresponding chunks of the input sentence (cf. Section 3.1), these translations are more likely to be coherent in the context of the particular input sentence if the matched parts on the input side are syntactically and semantically related.

For matched phrases $\bar{e}_m$ between the input sentence and the source side of the fuzzy match, we define the contextual information of the input side using dependency relations between words e_m in $\bar{e}_m$ and the remaining words e_j in the input sentence $\mathbf{e}$.

We use the Stanford parser to obtain the dependency structure of the input sentence. We add a pseudo-label SYS_PUNCT to punctuation marks, whose governor and dependent are both the punctuation mark. The dependency features designed to capture the context of the matched input phrases $\bar{e}_m$ are as follows:

Coverage features measure the coverage of dependency labels on the input sentence in order to obtain a bigger picture of the matched parts in the input. For each dependency label *L*, we consider its head or modifier as *covered* if the corresponding input word e_m is covered by a matched phrase $\bar{e}_m$. Our coverage features are the frequencies of governor and dependent coverage calculated separately for each dependency label.

Position features identify whether the head and the tail of a sentence are matched, as these are the cases in which the matched translation is not affected by the preceding words (when it is the head) or following words (when it is the tail), and is therefore more reliable. The feature is set to 1 if this happens, and to 0 otherwise. We distinguish among the possible dependency labels, the head or the tail of the sentence, and whether the aligned word is the governor or the dependent. As a result, each permutation of these possibilities constitutes a distinct binary feature.

The consistency feature is a single feature which determines whether matched phrases $\bar{e}_m$ belong to a consistent dependency structure, instead of being distributed discontinuously around in the input sentence. We assume that a consistent structure is less influenced by its surrounding context. We set this feature to 1 if every word in $\bar{e}_m$ is dependent on another word in $\bar{e}_m$, and to 0 otherwise.

5 Experiments

5.1 Experimental Setup

Our data set is an English–Chinese translation memory with technical translation from Symantec, consisting of 87K sentence pairs. The average sentence length of the English training set is 13.3 words and the size of the training set is comparable to the larger TMs used in the industry. Detailed corpus statistics about the training, development and test sets for the SMT system are shown in Table 1.

The composition of test subsets based on fuzzy match scores is shown in Table 2. We can see that sentences in the test sets are longer than those in the training data, implying a relatively difficult translation task. We train the SVM classifier using the libSVM (Chang and Lin, 2001) toolkit. The SVM-training and validation is on the same training sentences[1] as the SMT system with 5-fold cross validation.

	Train	Develop	Test
SENTENCES	86,602	762	943
ENG. TOKENS	1,148,126	13,955	20,786
ENG. VOC.	13,074	3,212	3,115
CHI. TOKENS	1,171,322	10,791	16,375
CHI. VOC.	12,823	3,212	1,431

Table 1: Corpus Statistics

Scores	Sentences	Words	W/S
(0.9, 1.0)	80	1526	19.0750
(0.8, 0.9]	96	1430	14.8958
(0.7, 0.8]	110	1596	14.5091
(0.6, 0.7]	74	1031	13.9324
(0.5, 0.6]	104	1811	17.4135
(0, 0.5]	479	8972	18.7307

Table 2: Composition of test subsets based on fuzzy match scores

The SVM hyper-parameters are tuned using the training data of the first fold in the 5-fold cross validation via a brute force grid search. More specifically, for parameter C in (1), we search in the range $[2^{-5}, 2^{15}]$, while for parameter γ (2) we search in the range $[2^{-15}, 2^{3}]$. The step size is 2 on the exponent.

We conducted experiments using a standard log-linear PB-SMT model: GIZA++ implementation of IBM word alignment model 4 (Och and Ney, 2003), the refinement and phrase-extraction heuristics described in (Koehn et al., 2003), minimum-error-rate training (Och, 2003), a 5-gram language model with Kneser-Ney smoothing (Kneser and Ney, 1995) trained with SRILM (Stolcke, 2002) on the Chinese side of the training data, and Moses (Koehn et al., 2007) which is capable of handling user-specified translations for some portions of the input during decoding. The maximum phrase length is set to 7.

5.2 Evaluation

The performance of the phrase-based SMT system is measured by BLEU score (Papineni et al., 2002) and TER (Snover et al., 2006). Significance test-

[1]We have around 87K sentence pairs in our training data. However, for 67.5% of the input sentences, our MT system produces the same translation irrespective of whether the input sentence is marked up or not.

ing is carried out using approximate randomisation (Noreen, 1989) with a 95% confidence level.

We also measure the quality of the classification by precision and recall. Let A be the set of predicted markup input sentences, and B be the set of input sentences where the markup version has a lower TER score than the plain version. We standardly define precision P and recall R as in (7):

$$P = \frac{|A \bigcap B|}{|A|}, R = \frac{|A \bigcap B|}{|B|} \quad (7)$$

5.3 Cross-fold translation

In order to obtain training samples for the classifier, we need to label each sentence in the SMT training data as to whether marking up the sentence can produce better translations. To achieve this, we translate both the marked-up versions and plain versions of the sentence and compare the two translations using the sentence-level evaluation metric TER.

We do not make use of additional training data to translate the sentences for SMT training, but instead use cross-fold translation. We create a new training corpus T by keeping 95% of the sentences in the original training corpus, and creating a new test corpus H by using the remaining 5% of the sentences. Using this scheme we make 20 different pairs of corpora (T_i, H_i) in such a way that each sentence from the original training corpus is in exactly one H_i for some $1 \leq i \leq 20$. We train 20 different systems using each T_i, and use each system to translate the corresponding H_i as well as the marked-up version of H_i using the procedure described in Section 3.1. The development set is kept the same for all systems.

5.4 Experimental Results

5.4.1 Translation Results

Table 3 contains the translation results of the SMT system when we use discriminative learning to mark up the input sentence (MARKUP-DL). The first row (BASELINE) is the result of translating plain test sets without any markup, while the second row is the result when all the test sentences are marked up. We also report the oracle scores, i.e. the upperbound of using our discriminative learning approach. As we can see from this table, we obtain significantly inferior results compared to the the Baseline system if we categorically mark up all the input sentences using phrase pairs derived from fuzzy matches. This is reflected by an absolute 1.4 point drop in BLEU score and a 1.8 point increase in TER. On the other hand, both the oracle BLEU and TER scores represent as much as a 2.5 point improvement over the baseline. Our discriminative learning method (MARKUP-DL), which automatically classifies whether an input sentence should be marked up, leads to an increase of 0.7 absolute BLEU points over the BASELINE, which is statistically significant. We also observe a slight decrease in TER compared to the BASELINE. Despite there being much room for further improvement when compared to the Oracle score, the discriminative learning method appears to be effective not only in maintaining translation consistency, but also a statistically significant improvement in translation quality.

	TER	BLEU
BASELINE	39.82	45.80
MARKUP	41.62	44.41
MARKUP-DL	39.61	46.46
ORACLE	37.27	48.32

Table 3: Performance of Discriminative Learning (%)

5.4.2 Classification Confidence Thresholding

To further analyse our discriminative learning approach, we report the classification results on the test set using the SVM classifier. We also investigate the use of classification confidence, as described in Section 3.2.2, as a threshold to boost classification precision if required. Table 4 shows the classification and translation results when we use different confidence thresholds. The default classification confidence is 0.50, and the corresponding translation results were described in Section 5.4.1. We investigate the impact of increasing classification confidence on the performance of the classifier and the translation results. As can be seen from Table 4, increasing the classification confidence up to 0.70 leads to a steady increase in classification precision with a corresponding sacrifice in recall. The fluctuation in classification performance has an impact on the translation results as measured by BLEU and TER. We can see that the best BLEU as well as TER scores are achieved when we set the classification confidence to 0.60, representing a modest improve-

	Classification Confidence						
	0.50	0.55	0.60	0.65	0.70	0.75	0.80
BLEU	46.46	46.65	**46.69**	46.59	46.34	46.06	46.00
TER	39.61	39.46	**39.32**	39.36	39.52	39.71	39.71
P	60.00	68.67	70.31	**74.47**	72.97	64.28	88.89
R	32.14	29.08	22.96	**17.86**	13.78	9.18	4.08

Table 4: The impact of classification confidence thresholding

ment over the default setting (0.50). Despite the higher precision when the confidence is set to 0.7, the dramatic decrease in recall cannot be compensated for by the increase in precision.

We can also observe from Table 4 that the recall is quite low across the board, and the classification results become unstable when we further increase the level of confidence to above 0.70. This indicates the degree of difficulty of this classification task, and suggests some directions for future research as discussed at the end of this paper.

5.4.3 Comparison with Previous Work

As discussed in Section 2, both (Koehn and Senellart, 2010) and (Zhechev and van Genabith, 2010) used fuzzy match score to determine whether the input sentences should be marked up. The input sentences are only marked up when the fuzzy match score is above a certain threshold. We present the results using this method in Table 5. From this table, we can see an inferior performance compared to the BASELINE results (cf. Table 3) when the fuzzy match score is below 0.70. A modest gain can only be achieved when the fuzzy match score is above 0.8. This is slightly different from the conclusions drawn in (Koehn and Senellart, 2010), where gains are observed when the fuzzy match score is above 0.7, and in (Zhechev and van Genabith, 2010) where gains are only observed when the score is above 0.9. Comparing Table 5 with Table 4, we can see that our classification method is more effective. This confirms our argument in the last paragraph of Section 2, namely that fuzzy match score is not informative enough to determine the usefulness of the subsentences in a fuzzy match, and that a more comprehensive set of features, as we have explored in this paper, is essential for the discriminative learning-based method to work.

	Fuzzy Match Scores				
	0.50	0.60	0.70	0.80	0.90
BLEU	45.13	45.55	45.58	45.84	45.82
TER	40.99	40.62	40.56	40.29	40.07

Table 5: Performance using fuzzy match score for classification

FM Scores	w. markup	w/o markup
[0,0.5]	37.75	62.24
(0.5,0.6]	40.64	59.36
(0.6,0.7]	40.94	59.06
(0.7,0.8]	46.67	53.33
(0.8,0.9]	54.28	45.72
(0.9,1.0]	44.14	55.86

Table 6: Percentage of training sentences with markup vs without markup grouped by fuzzy match (FM) score ranges

To further validate our assumption, we analyse the training sentences by grouping them according to their fuzzy match score ranges. For each group of sentences, we calculate the percentage of sentences where markup (and respectively without markup) can produce better translations. The statistics are shown in Table 6. We can see that for sentences with fuzzy match scores lower than 0.8, more sentences can be better translated without markup. For sentences where fuzzy match scores are within the range $(0.8, 0.9]$, more sentences can be better translated with markup. However, within the range $(0.9, 1.0]$, surprisingly, actually more sentences receive better translation without markup. This indicates that fuzzy match score is not a good measure to predict whether fuzzy matches are beneficial when used to constrain the translation of an input sentence.

5.5 Contribution of Features

We also investigated the contribution of our different feature sets. We are especially interested in the contribution of dependency features, as they re-

Example 1	
w/o markup	after policy name , type the name of the policy (it shows new host integrity policy by default) .
Translation	在 “ 策略 ” 名称 后面 ， 键入 策略 的 名称 (名称 显示 为 “ 新 主机 完整性 策略 默认 ） 。
w. markup	after policy name <tm translation=“， 键入 策略 名称 （ 默认 显示 “ 新 主机 完整性 策略 ” ） 。”>, type the name of the policy (it shows new host integrity policy by default) .< /tm>
Translation	在 “ 策略 ” 名称 后面 ， 键入 策略 名称 （ 默认 显示 “ 新 主机 完整性 策略 ” ） 。
Reference	在 “ 策略 名称 ” 后面 ， 键入 策略 名称 （ 默认 显示 “ 新 主机 完整性 策略 ” ） 。
Example 2	
w/o markup	changes apply only to the specific scan that you select .
Translation	更改 仅 适用于 特定 扫描 的 规则 。
w. markup	changes apply only to the specific scan that you select <tm translation=“。”>.< /tm>
Translation	更改 仅 适用于 您 选择 的 特定 扫描 。
Reference	更改 只 应用于 您 选择 的 特定 扫描 。

flect whether translation consistency can be captured using syntactic knowledge. The classification and

	TER	BLEU	P	R
TM+TRANS	40.57	45.51	52.48	27.04
+DEP	39.61	46.46	60.00	32.14

Table 7: Contribution of Features (%)

translation results using different features are reported in Table 7. We observe a significant improvement in both classification precision and recall by adding dependency (DEP) features on top of TM and translation features. As a result, the translation quality also significantly improves. This indicates that dependency features which can capture structural and semantic similarities are effective in gauging the usefulness of the phrase pairs derived from the fuzzy matches. Note also that without including the dependency features, our discriminative learning method cannot outperform the BASELINE (cf. Table 3) in terms of translation quality.

5.6 Improved Translations

In order to pinpoint the sources of improvements by marking up the input sentence, we performed some manual analysis of the output. We observe that the improvements can broadly be attributed to two reasons: 1) the use of long phrase pairs which are missing in the phrase table, and 2) deterministically using highly reliable phrase pairs.

Phrase-based SMT systems normally impose a limit on the length of phrase pairs for storage and speed considerations. Our method can overcome this limitation by retrieving and reusing long phrase pairs on the fly. A similar idea, albeit from a different perspective, was explored by (Lopez, 2008), where he proposed to construct a phrase table on the fly for each sentence to be translated. Differently from his approach, our method directly translates part of the input sentence using fuzzy matches retrieved on the fly, with the rest of the sentence translated by the pre-trained MT system. We offer some more insights into the advantages of our method by means of a few examples.

Example 1 shows translation improvements by using long phrase pairs. Compared to the reference translation, we can see that for the underlined phrase, the translation without markup contains (i) word ordering errors and (ii) a missing right quotation mark. In Example 2, by specifying the translation of the final punctuation mark, the system correctly translates the relative clause ‘that you select’. The translation of this relative clause is missing when translating the input without markup. This improvement can be partly attributed to the reduction in search errors by specifying the highly reliable translations for phrases in an input sentence.

6 Conclusions and Future Work

In this paper, we introduced a discriminative learning method to tightly integrate fuzzy matches retrieved using translation memory technologies with phrase-based SMT systems to improve translation consistency. We used an SVM classifier to predict whether phrase pairs derived from fuzzy matches could be used to constrain the translation of an in-

put sentence. A number of feature functions including a series of novel dependency features were used to train the classifier. Experiments demonstrated that discriminative learning is effective in improving translation quality and is more informative than the fuzzy match score used in previous research. We report a statistically significant 0.9 absolute improvement in BLEU score using a procedure to promote translation consistency.

As mentioned in Section 2, the potential improvement in sentence-level translation consistency using our method can be attributed to the fact that the translation of new input sentences is closely informed and guided (or constrained) by previously translated sentences using global features such as dependencies. However, it is worth noting that the level of gains in translation consistency is also dependent on the nature of the TM itself; a self-contained coherent TM would facilitate consistent translations. In the future, we plan to investigate the impact of TM quality on translation consistency when using our approach. Furthermore, we will explore methods to promote translation consistency at document level.

Moreover, we also plan to experiment with phrase-by-phrase classification instead of sentence-by-sentence classification presented in this paper, in order to obtain more stable classification results. We also plan to label the training examples using other sentence-level evaluation metrics such as Meteor (Banerjee and Lavie, 2005), and to incorporate features that can measure syntactic similarities in training the classifier, in the spirit of (Owczarzak et al., 2007). Currently, only a standard phrase-based SMT system is used, so we plan to test our method on a hierarchical system (Chiang, 2005) to facilitate direct comparison with (Koehn and Senellart, 2010). We will also carry out experiments on other data sets and for more language pairs.

Acknowledgments

This work is supported by Science Foundation Ireland (Grant No 07/CE/I1142) and part funded under FP7 of the EC within the EuroMatrix+ project (grant No 231720). The authors would like to thank the reviewers for their insightful comments and suggestions.

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, MI.

Ergun Biçici and Marc Dymetman. 2008. Dynamic translation memory: Using statistical machine translation to improve translation memory. In *Proceedings of the 9th Internation Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, pages 454–465, Haifa, Israel.

Chih-Chung Chang and Chih-Jen Lin, 2001. *LIBSVM: a library for support vector machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

David Chiang. 2005. A hierarchical Phrase-Based model for Statistical Machine Translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 263–270, Ann Arbor, MI.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

Yifan He, Yanjun Ma, Josef van Genabith, and Andy Way. 2010. Bridging SMT and TM with translation recommendation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 622–630, Uppsala, Sweden.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 181–184, Detroit, MI.

Philipp Koehn and Jean Senellart. 2010. Convergence of translation memory and statistical machine translation. In *Proceedings of AMTA Workshop on MT Research and the Translation Industry*, pages 21–31, Denver, CO.

Philipp Koehn, Franz Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of the 2003 Human Language Technology Conference and the North American Chapter of the Association for Computational Linguistics*, pages 48–54, Edmonton, AB, Canada.

Philipp Koehn, 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 Vol-*

ume Proceedings of the Demo and Poster Sessions, pages 177–180, Prague, Czech Republic.

Vladimir Iosifovich Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710.

Hsuan-Tien Lin, Chih-Jen Lin, and Ruby C. Weng. 2007. A note on platt's probabilistic outputs for support vector machines. *Machine Learning*, 68(3):267–276.

Adam Lopez. 2008. Tera-scale translation models via pattern matching. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 505–512, Manchester, UK, August.

Eric W. Noreen. 1989. *Computer-Intensive Methods for Testing Hypotheses: An Introduction*. Wiley-Interscience, New York, NY.

Franz Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan.

Karolina Owczarzak, Josef van Genabith, and Andy Way. 2007. Labelled dependencies in machine translation evaluation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 104–111, Prague, Czech Republic.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of Machine Translation. In *40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, PA.

John C. Platt. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers*, pages 61–74.

Richard Sikes. 2007. Fuzzy matching in theory and practice. *Multilingual*, 18(6):39–43.

Michel Simard 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.

James Smith and Stephen Clark. 2009. EBMT for SMT: A new EBMT-SMT hybrid. In *Proceedings of the 3rd International Workshop on Example-Based Machine Translation*, pages 3–10, Dublin, Ireland.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas (AMTA-2006)*, pages 223–231, Cambridge, MA, USA.

Lucia Specia, Craig Saunders, Marco Turchi, Zhuoran Wang, and John Shawe-Taylor. 2009. 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.

Andreas Stolcke. 2002. SRILM – An extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 901–904, Denver, CO.

Ventsislav Zhechev and Josef van Genabith. 2010. Seeding statistical machine translation with translation memory output through tree-based structural alignment. In *Proceedings of the Fourth Workshop on Syntax and Structure in Statistical Translation*, pages 43–51, Beijing, China.

Rich Linguistic Features for Translation Memory-Inspired Consistent Translation

Yifan He **Yanjun Ma**[†] **Andy Way**[‡*] **Josef van Genabith**
CNGL, School of Computing, Dublin City University, Dublin, Ireland
{yhe,josef}@computing.dcu.ie
[†] Baidu Inc., China
yma@baidu.com
[‡] Applied Language Solutions, Delph, UK
andy.way@appliedlanguage.com

Abstract

We improve translation memory (TM)-inspired consistent phrase-based statistical machine translation (PB-SMT) using rich linguistic information including lexical, part-of-speech, dependency, and semantic role features to predict whether a TM-derived sub-segment should constrain PB-SMT translation. Besides better translation consistency, for English-to-Chinese Symantec TMs we report a 1.01 BLEU point improvement over a regular state-of-the-art PB-SMT system, and a 0.45 BLEU point improvement over a TM-constrained PB-SMT system without access to rich linguistic information, both statistically significant ($p < 0.01$). We analyze the system output and summarize the benefits of using linguistic annotations to characterise the nature of translation consistency.

1 Introduction

In the world of localization and professional translation, translation consistency is a much desired property. Given a particular domain, consistency in translation is characterized not only by correctness and fluency, but also by adhering to specific terminology translation, language patterns, and even error patterns (which are easier to identify in the post-editing stage). However, state-of-the-art statistical machine translation (SMT) systems do not explicitly model translation consistency, as the objective of these systems is to produce translations that maximize a weighted combination of translation model and language model scores (among others). Translation memories (TMs), widely used in the localization industry, assist translators by retrieving and displaying previously translated similar "example" segments (displayed as source-target pairs, called "fuzzy matches"). When presented with fuzzy matches, translators can avail of useful complete matching sub-segments in previous translations while composing the translation of a new segment. This improves the consistency of translation, as new translations produced by translators are based on the target side of the fuzzy match they have consulted, and translators will build their translations around terminologies already used in the TM.

It is, therefore, natural to resort to TMs for consistent translation, and to incorporate fully matching sub-segments from fuzzy match examples into the SMT pipeline (cf. (Koehn and Senellart, 2010), (Zhechev and van Genabith, 2010), and (Ma et al., 2011)). Although these methods have led to improved translations, they only use very simple features (such as a threshold on the fuzzy match score of the complete TM segment) to determine whether matching sub-segments from the fuzzy match are suitable for use in the SMT pipeline. Here, we propose a rich set of linguistic features to select TM fuzzy matches that contain useful sub-segments that improve translation consistency in an SMT pipeline.[1] We assume that many factors are rele-

[*]Work done while at CNGL, School of Computing, DCU.

[1]In Ma et al. (2011), we considered a richer set of features – including features from the translation model and source-side dependency relations – but a thorough exploration of features is not conducted, and linguistically-motivated features are limited

vant in deciding whether a full TM segment contains sub-segments that should be reused in translation: translation model, lexical, syntactic (dependency), and semantic features can all be helpful in predicting the consistency of translation, and improve translation quality. We explore a rich set of linguistic features in a consistency-oriented constrained translation task following the paradigm proposed by Ma et al. (2011). We show that our method leads to translations of better quality, reflected by a 1.01 BLEU point improvement over an out-of-the-box phrase-based SMT (PB-SMT) system, and a 0.45 BLEU point improvement over the system of Ma et al. (2011), all statistically significant ($p < 0.01$). Furthermore, our approach provides insight into the linguistic properties of consistent translation pairs.

The paper is organized as follows: we review related work in Section 2, and summarize the approach of Ma et al. (2011) in Section 3. We introduce and compare features induced from translation models and linguistically-oriented features for consistent translation in Section 4. We present experimental results and analyze linguistic properties of consistent translation in Section 5. In Section 6, we conclude and point out some possible avenues for future work.

2 Related Work

Our approach extends the line of research proposed by Ma et al. (2011), which improves the consistency of translations in PB-SMT systems by constraining the SMT system with consistent phrase pairs induced from TMs. Whether the consistent phrase pairs should be used is determined through discriminative learning. As the research in this paper builds on this previous work of ours, we review it in detail in Section 3. Prior to Ma et al. (2011), several proposals used translation information derived from TM fuzzy matches, such as (i) adding such translations into a phrase table as in Biçici and Dymetman (2008)[2] and Simard and Isabelle (2009), or (ii) marking up the input segment using the relevant sub-segment translations in the fuzzy match, and using an MT system to translate the parts that are not marked up, as in Smith and Clark (2009), Koehn and Senellart (2010), and Zhechev and van Genabith (2010). However, these do not include a classification step that determines whether consistent phrase pairs should be used.

to dependency labels.

[2]Note that discontinuous phrase pairs are used in Biçici and Dymetman (2008), whereas we use continuous phrase pairs here.

3 Constrained Translation via Markup Classification

Ma et al. (2011) tightly integrate TM with MT at the sub-segment level in the following way: given a segment $\mathbf{e}$ to translate, the most similar segment $\mathbf{e}'$ from the TM associated with the target translation $\mathbf{f}'$ is retrieved, and the m longest common subsequences ("phrases") $\bar{\mathbf{e}}_{\mathbf{1}}^{\mathbf{m}}$ between $\mathbf{e}$ and $\mathbf{e}'$ are identified. Then a set of "consistent phrase pairs" $\{(\bar{e}_i, \bar{f}_i)\}_{i=1}^{m}$ is derived using the word alignment information between $\mathbf{e}'$ and $\mathbf{f}'$. These "consistent phrase pairs" are used to mark up the matched sub-segments in the source segment with the predetermined translations, as described in Ma et al. (2011). If a classifier predicts that the markup will lead to improved translation quality and translation, the consistent phrase translation will be reused directly in the translation process. Below we explain how consistent phrase pairs are defined, and how markup classification is performed. Based on these, we discuss why linguistic features are essential for this task.

3.1 Consistent Phrase Pair Identification

We use the method of Ma et al. (2011) to extract consistent phrase pairs: extracted phrase pairs are the intersections of bidirectional GIZA++ posterior alignments (Och and Ney, 2003) between the source and the target side of the TM fuzzy match. We use the intersected word alignment to minimize the noise introduced by word alignment in one direction only, in order to ensure translation consistency.

3.2 Markup Classification

Following (Ma et al., 2011), we use Support Vector Machines (SVMs, (Cortes and Vapnik, 1995)) to determine whether constraining translation with our consistent phrase pairs can help improve translation quality. We treat constrained translation as a binary classification problem, and use the SVM classifier to decide whether we should mark up a segment or not. We label training data using the automatic TER score (Snover et al., 2006), as in (1).

$$y = \begin{cases} +1 & \text{if TER(w. markup)} < \text{TER(w/o markup)} \\ -1 & \text{if TER(w/o markup)} \geq \text{TER(w. markup)} \end{cases} \quad (1)$$

Each data point is associated with a set of features which are discussed in more detail in Section 4.

We perform our experiments with the Radial Basis Function (RBF) kernel, and use Platt's method (Platt, 1999) (as improved by (Lin et al., 2007)) to fit the SVM output to a sigmoid function, to obtain probabilistic outputs from the SVM.

3.3 Rich Linguistic Features for Markup Classification

A close look at the markup classification procedure shows that the accuracy of classification (and ultimately, the quality and consistency of the output) is determined by how well we can capture the characteristics of a sub-segment that is a "consistent translation".

When integrating sub-segments from TMs into the SMT pipeline, previous work focused mainly on using information from the translation models of the TM or MT systems (cf. Section 2).[3] However, linguistic annotations are potentially stronger indicators as to whether a fuzzy match sub-segment can constitute a consistent translation. For example, in industrial translation, brand and product names are often kept constant in the original form. A markup classifier which is only informed by translation model features may fail to capture this information, but a sequence of NN POS-tags would be a strong indicator.

Following this intuition, we explore a rich set of linguistic features including lexical information, part-of-speech, dependency, and semantic roles to improve translation consistency prediction.

4 Translation Features and Linguistic Features

4.1 Translation Features

We use both the TM fuzzy match score and features derived from the SMT model to predict the quality of consistent phrase pairs, following (Ma et al., 2011).

4.1.1 The TM Feature

The TM feature is the fuzzy match score, which indicates the overall similarity between the input segment and the source side of the TM output. We compute fuzzy match cost h_{fm} as the minimum edit distance (Levenshtein, 1966) between the source and TM entry, normalized by the length of the source, as in (2).

$$h_{\text{fm}}(\mathbf{e}) = \min_{\mathbf{s}} \frac{\text{LevenshteinDistance}(\mathbf{e}, \mathbf{s})}{\text{Len}(\mathbf{e})} \quad (2)$$

where $\mathbf{e}$ is the segment to translate, and $\mathbf{s}$ is the source side of an entry in the TM. For fuzzy match scores F, h_{fm} roughly corresponds to $1 - F$.

4.1.2 Translation Features

We use the six features induced from the SMT translation model, following (Ma et al., 2011): the phrase translation and lexical probabilities for the consistent phrase pairs (cf. Section 3.1) in both directions derived using the method in Section 3, a count feature (i.e. the number of phrases used to mark up the input segment), and a binary feature (i.e. whether the phrase table contains at least one phrase pair $< \bar{e}_m, \bar{f'}_m >$ that is used to mark up the input segment).

4.2 Linguistic Features

The linguistic features measure how well the marked-up portion covers the source segment. The assessments could be (but are not limited to) coverage measures, such as the percentage of content words that are marked up (lexical level) or the number of covered nouns (Part-of-speech (POS) level), as well as position-related properties, such as whether the marked-up sub-segment is at the beginning or the end of the segment.

Lexical Features Lexical features capture the surface-level properties of the marked-up translation. We use the following indicators given a segment and its markup: *Coverage* measures the percentage of words covered by the marked-up segment, which we calculate on both the source and the target side; *Alphabetical Words* measures the percentage of words that are alphabetical (i.e. not numbers and punctuation marks) in the source side of

[3] In Ma et. al (2011), we tentatively used information deduced from dependency relations, and reported that these features are more beneficial to markup classification than plain translation model features.

marked-up sub-segments; *Punctuation Marks* measures the percentage of words in the source side of marked-up sub-segments that are punctuation marks; *Content Words* calculates the percentage of content words in the source side of marked up sub-segments, for which we use the snowball stop words list[4] to identify function words; and finally *Position* comprises two binary features which fire if marked-up sub-segments cover the head or the tail of the source segment.

POS Features For the POS features, we simply extend the calculation of lexical features to the POS level. The POS tags in our experiments are obtained using the Stanford Parser.[5] For ease of discussion, we define $\text{POS}_i(\bar{e}_m)$ as the number of words in the input segment e that are marked up with translations from TM, and have the POS tag POS_i. We also define $\#\text{POS}_i(e)$ as the number of words in e that have the POS tag POS_i. We calculate the *POS Coverage* for each POS tag in the input segment, where $\text{POS\_Coverage\_POS}_i = \#\text{POS}_i(\bar{e}_m)/\#\text{POS}_i(e)$. We also use a binary feature *POS Position* to indicate whether the consistent phrase pair covers the head or the tail of a segment, where POS_Head_POS_i=1 iff the first word of the source segment is marked up and has the POS tag POS_i.

Dependency Features We use dependency relations (obtained using the Stanford parser) to establish the roles of matched parts in the input sentence in terms of syntactic dependencies. The dependency features include *DEP Coverage*, *DEP Position*, and *DEP Consistency*, all of which follow the definitions in Ma et al. (2011).

Semantic Role Features Our semantic role labels are obtained using the Suda SRL labeler,[6] with constituent trees produced by the Stanford parser as input. The labels follow the PropBank (Palmer et al., 2005) annotation. Following POS features, for each predicate identified in a segment, we define $\text{SEM}_i(\bar{e}_m)$ as the number of words in the input segment e having the role SEM_i that are marked up with translations from TM, and define $\#\text{SEM}_i(e)$ as the number of words in e that have the SEM role SEM_i. The features include *SEM Partial Coverage* which calculates the marked-up percentage for each argument label,[7] *SEM Complete Coverage* – a binary feature that fires if the phrase with an argument label is completely covered by the markup (i.e. $\text{SEM\_COMPLETE\_SEM}_i$=1 iff SEM_PARTIAL_SEM_i=1.0) – *SEM Position* which fires if an argument at the beginning or the end of the segment is covered by the markup, and *SEM Predicate* which fires only if the segment has no predicate.

5 Experiments

We use the same data set as in Ma et al. (2011), an English–Chinese TM with technical translation from Symantec, consisting of 87K segment pairs. The average segment length of the English training set is 13.3 words and the size of the training set is comparable to the larger TMs used in the industry.

We obtain training samples using the cross-fold translation technique in Ma et al. (2011), so the word aligner, the translation models, and the classifier are all trained on the same training corpus. We train the SVM classifier using the libSVM (Chang and Lin, 2001) toolkit. As for SVM parameters, we set $c = 2.0$ and $\gamma = 0.125$.

We conducted experiments using a standard log-linear PB-SMT (Och and Ney, 2004) system Moses,[8] which is capable of handling user-specified translations for portions of the input during decoding. The maximum phrase length is set to 7.

5.1 Evaluation

The performance of the phrase-based SMT system is measured by BLEU score (Papineni et al., 2002) and TER (Snover et al., 2006). Significance testing is carried out using approximate randomization (Noreen, 1989).

We also measure the quality of the classification using precision and recall. Let A be the set of predicted markup input segments, and B be the set of input segments where the markup version has a lower TER score than the plain version. We stan-

[4] http://snowball.tartarus.org/algorithms/english/stop.txt

[5] http://nlp.stanford.edu/software/lex-parser.shtml

[6] http://nlp.suda.edu.cn/~jhli/

[7] If more than one predicate is identified, the value of the feature is averaged among argument labels for each predicate.

[8] http://www.statmt.org/moses/

dardly define precision P and recall R as in (3):

$$P = \frac{|A \bigcap B|}{|A|}, R = \frac{|A \bigcap B|}{|B|} \quad (3)$$

5.2 Experimental Results

5.2.1 Feature Validation

Table 1: Contribution of Features (%)

	TER	BLEU	P	R
BASELINE	39.82	45.80	N/A	N/A
TRANS	39.80	45.84	66.67	1.02
LEX	39.65	46.20	71.43	10.20
POS	39.30	46.71*	61.54	28.57
DEP	39.81	46.14	58.25	30.61
SEM	39.74	46.35	59.09	19.90
LPDS	39.32	46.81*	61.36	41.33

We first validate the contribution of the feature sets proposed. The classification and translation results using different features are reported in Table 1. BLEU scores marked with "*" are statistically significantly better ($p < 0.01$) than the BASELINE.

We observe that using translation model-derived features (TRANS) only brings about a trivial difference in translation quality. In fact, very low recall indicates that the SVM actually cannot obtain enough information from this feature set, and has to take advantage of the prior distribution of the samples (where we have more negative examples than positive ones) and reject almost every markup attempt to obtain the best accuracy. This shows that these features cannot capture the properties of the TM subsegments that help translation consistency.

By contrast, we observe that the linguistic features improve classification accuracy and translation quality. The improvement in BLEU scores range from 0.34 (DEP) to a statistically significant 0.91 (POS), which is the highest BLEU score obtained using a single set of features. However, we also observe that using POS features leads to a lower recall than using DEP. When we put the LEX, POS, DEP, and POS features together in the LPDS setting, we achieve the best BLEU score among all the settings, which is also significantly better than the baseline. The TER and precision numbers are marginally inferior to those obtained using the POS features alone. However, the much higher recall enables us to perform more confidence threshold-based tuning and achieve better results (cf. Section 5.2.3).

5.2.2 The Impact of Constrained Translation and Linguistic Features

We aim to obtain some insight on how much constrained translation can improve translation quality, and how much improvement is brought about by the linguistic features.

Table 2 contains the translation results[9] of the SMT system when we use discriminative learning with LPDS to mark up the input segment (LPDS), which we compare to three baselines. The first baseline (BASELINE) is the result of translating plain test sets without any markup, and the second baseline is the result when all test segments are marked up. We also report results on a third baseline: TRANS+DEP, which corresponds to the best result reported in Ma et al. (2011). Besides these baselines, we also report the oracle scores, i.e. the upperbound of using our discriminative learning approach. As is reported in Ma et al. (2011), if we categorically mark up all the input segments using phrase pairs derived from fuzzy matches, this leads to an absolute 1.4 point drop in BLEU score and a 1.8 point deterioration in TER. In contrast, both the ORACLE BLEU and TER scores represent as much as a 2.5 point improvement over the baseline.

Table 2: Performance of Discriminative Learning (%)

	TER	BLEU
BASELINE	39.82	45.80
MARKUP	41.62	44.41
TRANS	39.80	45.84
TRANS+DEP	39.63	46.36
LPDS	39.32	46.81
ORACLE	37.27	48.32

Our discriminative learning method with a full linguistic feature set (LPDS) leads to an increase

[9] Note that two of the baseline scores we report are slightly different from those in Ma et al. (2011) due to small differences in data processing, with our previous TRANS score slightly lower (BLEU:45.51%) than in this paper, and TRANS+DEP slightly higher (BLEU:46.46%) than ours. Comparing the TRANS+DEP output in Ma et al. (2011) and running a significance test, our LPDS setting still significantly outperformed the TRANS+DEP setting in that paper with respect to BLEU at $p < 0.05$.

of 1.01 absolute BLEU points over the BASELINE, which is statistically significant with $p < 0.01$. We also observe a 0.5 point improvement in TER compared to the BASELINE, which shows that the proposed method can clearly outperform a vanilla PB-SMT pipeline.

To examine the role of linguistic features in this task, we also compare the LPDS setting to the TRANS setting – which does not use any linguistic information – and to the TRANS+DEP setting, which corresponds to the setting in Ma et al. (2011). The LPDS setting outperforms both TRANS (0.97 BLEU points improvement) and TRANS+DEP (0.45 BLEU points improvement) with statistical significance at $p < 0.01$, which reiterates the essential role of rich linguistic features in the markup classification procedure; as is shown in these experiments, the more complete the set of linguistic features used in this task, the better the observed performance.

5.2.3 Translation Results with Confidence Threshold

To further analyze our discriminative learning approach, we also investigate the use of classification confidence (cf. Section 3.2) as a threshold to boost classification precision.

As can be seen from Table 3, increasing the classification confidence up to 0.65 leads to a steady increase in classification precision with a corresponding sacrifice in recall. The fluctuation in classification performance has an impact on the translation results as measured by BLEU and TER. We can see that the best BLEU and TER scores are achieved when we set the classification confidence to 0.55, representing in further improvements of 0.19 points in BLEU score and 0.22 points in TER score, compared to the default threshold of 0.50.

Compared to the BASELINE, we obtain improvements of 1.20 in BLEU and 0.72 in TER (with lower TER score), all statistically significant ($p < 0.01$), when we set the confidence to 0.55. Despite the higher precision when the confidence is set above 0.60, the dramatic decrease in recall cannot be compensated for by the increase in precision.

We also compare the effect of applying confidence thresholds to all linguistically motivated feature sets we have proposed in Figure 3. Note that the LPDS features obtain the best BLEU scores in the [0.5, 0.65] range and obtain the highest BLEU score at the confidence level of 0.55, which confirms our approach of combining a variety of linguistic features for this task. In addition, we observe that although the BLEU score of POS features is also competitive at the confidence level of 0.5, the translation quality will not improve as we set a higher threshold, because its recall is already low initially.

5.3 Translation Examples and Characteristics of Consistent Translation

From the output of our system, we identify three directions where consistent phrase pairs can improve on baseline SMT outputs: *segment skeleton*, *coherent concept*, and *consistent terminology*. We also see that using linguistic features helps to better capture these scenarios, such as *coherent concept* in our example.

Segment skeletons are consistent phrase pairs that cover most of the source segment, leaving only very few words to change. As the majority of the segment is covered by the fuzzy match from the TM, it is better to use the translation skeleton directly, rather than to using MT to recombine sub-segments from different sources from scratch.

Using the segment in Figure 1 as an example, the consistent phrase pair already covers the second part of the segment. Without accessing this information, the MT output introduces several small inconsistencies and errors in the translation, which are avoided when we reuse the TM fuzzy match.

Coherent concepts are consistent phrase pairs which convey a single, self-contained concept. As such phrases are relatively independent from the other parts of a segment, directly reusing their translations from the TM in the MT pipeline is less risky. If the whole sub-segment functions as a single semantic role in the segment, it is a strong indicator that it constitutes a coherent concept.

This also serves as evidence for the necessity of using rich linguistic features in this task. In Figure 2, the classifier using the TRANS+DEP feature set cannot identify the consistent phrase pair which actually covers `A0`, `AM-MOD`, `V`, and `A1` of the segment, and rejects the markup. In contrast, when using a full set of linguistic features, the LPDS classifier does make

Table 3: The impact of applying classification confidence threshold on the LPDS setting. Scores marked with "*" are significantly better ($p < 0.01$) than the BASELINE. The default classification confidence is 0.5.

	BASELINE	0.50	0.55	0.60	0.65	0.70	0.75
BLEU	45.80	46.81*	**47.00***	46.79*	46.47	46.11	46.03
TER	39.82	39.32	**39.10***	39.28	39.45	39.66	39.70
P	N/A	61.36	67.96	71.01	**75.00**	70.97	71.43
R	N/A	**41.33**	35.71	25.00	18.37	11.22	7.65

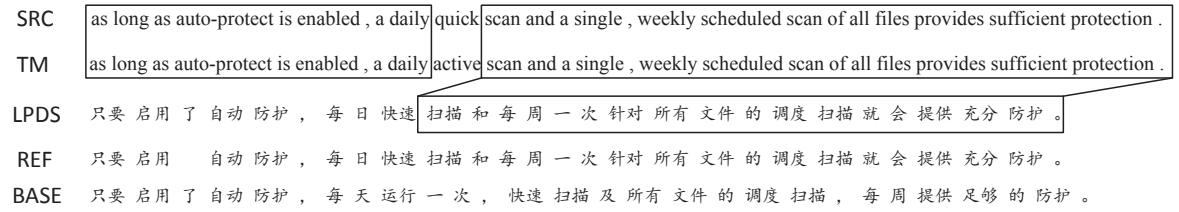

Figure 1: Consistent Translation Examples: Segment Skeleton. SRC: source segment. TM: target side of TM fuzzy match. LPDS: markup classification using combined linguistic features. REF: reference translation. BASE: plain PB-SMT. Links between SRC and TM indicate consistent phrase pairs.

the correct prediction that leads to improvement in translation quality.

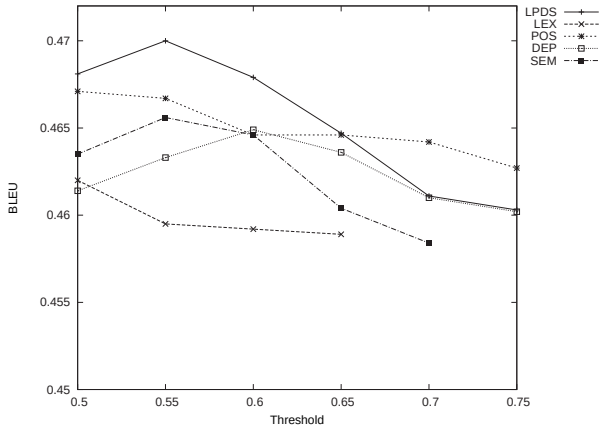

Figure 3: Confidence Threshold on Various Feature Sets

Consistent Terminology represents phrase pairs that are the translations of terminologies. In Figure 4, both LPDS and BASE translations are correct in meaning, but in the industrial environment, the LPDS translation is preferred, as it is using exactly the same Chinese expression when translating "any option" (underlined) from the previously translated segment found in the TM. This demonstrates that for enterprise translation, consistency is a dimension of translation quality that is independent from the more commonly used adequacy and fluency metrics.

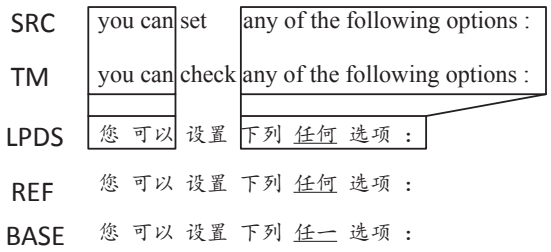

Figure 4: Consistent Translation Example: Consistent Terminology. Notations follow those of Figure 1.

6 Conclusion

We investigated a technique that exploits a rich linguistically-motivated feature set to find reusable translation sub-segments from TM fuzzy matches in a bid to improve translation consistency, extending the paradigm proposed by Ma et al. (2011). We show that by using rich linguistic features, we can better predict the reusability of consistent translation pairs derived from the TM: our method outperforms a PB-SMT baseline by 1.01 BLEU points, and a consistent translation-aware system reported in Ma et al. (2011) by 0.45 BLEU points, both statistically significant ($p < 0.01$). We also investigate the properties of consistent translations, and note that the consistent phrase pairs combine the strengths of keeping the segment skeleton, reusing coherent concept, and adhering to consistent terminology.

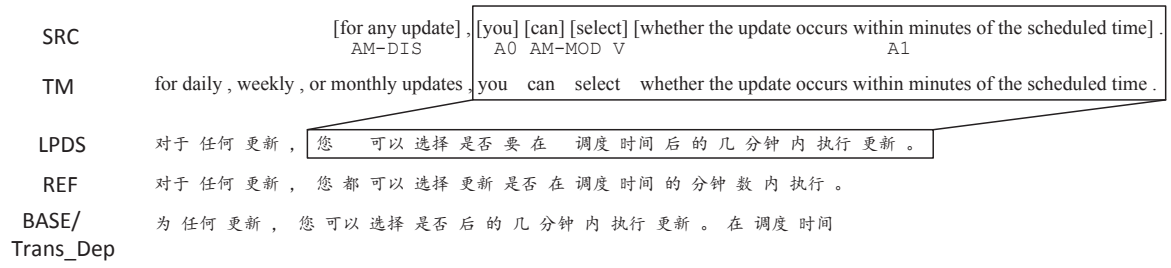

Figure 2: Consistent Translation Example:Coherent Concept. Notations follow those of Figure 1. Semantic role labels on the source side are annotated. TRANS_DEP: markup classification using the feature set in Ma et al. (2011)

There are many possibilities to explore along this line of research, such as testing this method on languages other than English, labelling the training examples using other segment-level evaluation metrics such as Meteor (Banerjee and Lavie, 2005), and testing our method on a hierarchical system (Chiang, 2005) to facilitate direct comparison with Koehn and Senellart (2010).

Acknowledgments

This work is supported by Science Foundation Ireland (Grant No 07/CE/I1142) and part funded under FP7 of the EC within the EuroMatrix+ project (Grant No 231720). We thank Junhui Li for his help with the Semantic Role Labeler.

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*.

Ergun Biçici and Marc Dymetman. 2008. Dynamic translation memory: Using statistical machine translation to improve translation memory. In *CICLing*.

Chih-Chung Chang and Chih-Jen Lin, 2001. *LIBSVM: a library for support vector machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

David Chiang. 2005. A hierarchical Phrase-Based model for Statistical Machine Translation. In *ACL*.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3).

Philipp Koehn and Jean Senellart. 2010. Convergence of translation memory and statistical machine translation. In *AMTA Workshop on MT Research and the Translation Industry*.

Vladimir Iosifovich Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8).

Hsuan-Tien Lin, Chih-Jen Lin, and Ruby C. Weng. 2007. A note on Platt's probabilistic outputs for support vector machines. *Machine Learning*, 68(3).

Yanjun Ma, Yifan He, Andy Way, and Josef van Genabith. 2011. Consistent translation using discriminative learning: A translation memory-inspired approach. In *ACL*.

Eric W. Noreen. 1989. *Computer-Intensive Methods for Testing Hypotheses: An Introduction*. Wiley-Interscience, New York, NY.

Franz Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1).

Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4).

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1).

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

John C. Platt. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers*.

Michel Simard and Pierre Isabelle. 2009. Phrase-based machine translation in a computer-assisted translation environment. In *MT Summit XII*.

James Smith and Stephen Clark. 2009. EBMT for SMT: A new EBMT-SMT hybrid. In *The 3rd International Workshop on Example-Based Machine Translation*.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *AMTA*.

Ventsislav Zhechev and Josef van Genabith. 2010. Seeding statistical machine translation with translation memory output through tree-based structural alignment. In *SSST*.

Bibliography

Here we list the references cited in the main text, as well as the papers, in which the research presented here has been published.

Yifan He, Yanjun Ma, Johann Roturier, Andy Way, and Josef van Genabith. 2010a. Improving the Post-Editing Experience using Translation Recommendation: a User Study. In *Proceedings of the The Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010)*, pages 141–150, Denver, Colorado.

Yifan He, Yanjun Ma, Josef van Genabith, and Andy Way. 2010b. Bridging SMT and TM with translation recommendation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 622–630, Uppsala, Sweden.

Yifan He, Yanjun Ma, Andy Way, and Josef van Genabith. 2010c. Integrating N-best SMT outputs into a TM system. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING-2010:Posters)*, pages 374–382, Beijing, China.

Yifan He, Yanjun Ma, Andy Way, and Josef van Genabith. 2011. Rich linguistic features for translation memory-inspired consistent translation. In *Machine Translation Summit XIII*, pages 456–463, Xiamen, China.

Philipp Koehn and Jean Senellart. 2010. Convergence of translation memory and statistical machine translation. In *AMTA Workshop on MT Research and the Translation Industry*, pages 21–31, Denver, CO.

Yanjun Ma, Yifan He, Andy Way, and Josef van Genabith. 2011. Consistent translation using discriminative learning: A translation memory-inspired approach. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 1239–1248, Portland, Oregon.

Ventsislav Zhechev and Josef van Genabith. 2010a. Maximising tm performance through sub-tree alignment and smt. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas (AMTA 10)*. Denver, CO.

Ventsislav Zhechev and Josef van Genabith. 2010b. Seeding statistical machine translation with translation memory output through tree-based structural alignment. In Dekai Wu, editor, *Proceedings of the Fourth Workshop on Syntax and Structure in Statistical Translation (SSST-4)*, page 4351. Beijing, China.

Ventsislav Zhechev. 2010. Highlighting matched and mismatched segments in translation memory output through sub-tree alignment. In *Proceedings of the Translating and the Computer Conference 2010 (T&C 10)*. London, UK. (to appear).