M3TRA: integrating TM and MT for professional translators

Bram Bulté
CCL – KU Leuven

Tom Vanallemeersch
CCL – KU Leuven

Vincent Vandeghinste
CCL – KU Leuven

firstname.lastname@ccl.kuleuven.be

Abstract

Translation memories (TM) and machine translation (MT) both are potentially useful resources for professional translators, but they are often still used independently in translation workflows. As translators tend to have a higher confidence in fuzzy matches than in MT, we investigate how to combine the benefits of TM retrieval with those of MT, by integrating the results of both. We develop a flexible TM-MT integration approach based on various techniques combining the use of TM and MT, such as fuzzy repair, span pretranslation and exploiting multiple matches. Results for ten language pairs using the DGT-TM dataset indicate almost consistently better BLEU, METEOR and TER scores compared to the MT, TM and NMT baselines.

1 Introduction

While software for professional translators has included translation memories (TMs) since several decades, especially in the context of specialized documents, the use of machine translation (MT) in such software is more recent. Even though certain commercial translation tools now offer functionalities such as automatic fuzzy match repair, TM and MT technologies are often still used independently, i.e. either a match for a query sentence or an MT output is provided. This is not ideal, as translators tend to have a higher confidence in 'human' TM than in MT. It has to be kept in mind, however, that only exact matches provide a translation of the query sentence; 'fuzzy' matches offer a translation of a similar sentence. In contrast, MT systems provide a translation for any sentence, but they have problems with a number of, often linguistic, issues, such as complex morphological phenomena, long distance dependencies and word order (Bisazza and Federico, 2016; Sudoh et al., 2010). We investigate how to combine the confidence in fuzzy match retrieval with full sentence translation by integrating TM and MT output. We develop M3TRA,[1] a method which performs a TM match preprocessing step before running a standard phrase-based statistical MT (PBSMT) system trained on the TM. M3TRA combines different approaches, and is flexible in several respects: it applies various fuzzy match score thresholds, allows for more than one match to be used per query sentence, and can use several fuzzy metrics. It comprises two main components: (a) **fuzzy repair**, automatically editing high-scoring fuzzy matches, and (b) **span pretranslation**, constraining MT output by including certain consistently aligned spans of one or more TM matches.

We perform tests on ten language pairs which involve multiple language families, using the DGT-TM dataset (Steinberger et al., 2013). We apply PBSMT without span pretranslation as a baseline, as well as 'pure' TM and a standard NMT system, and evaluate the translations using several metrics. M3TRA is integrated in a prototype translation interface providing translators with more 'informed' MT output (Coppers et al., 2018).

The following sections describe the research context, system architecture, experimental design and results. The final sections contain a discussion, overview of work in progress and conclusions.

© 2018 The authors. This article is licensed under a Creative Commons 3.0 licence, no derivative works, attribution, CC-BY-ND.

[1] MeMory + Machine TRAnslation

Pérez-Ortiz, Sánchez-Martínez, Esplà-Gomis, Popović, Rico, Martins, Van den Bogaert, Forcada (eds.)
Proceedings of the 21st Annual Conference of the European Association for Machine Translation, p. 69–78
Alacant, Spain, May 2018.

2 Research context

The baseline approach to TM-MT integration uses MT to translate a query sentence in case no sufficiently similar translation unit is found in the TM (Simard and Isabelle, 2009). This can be augmented by using an estimation of the *usefulness* of MT and TM output (He, 2011). Other studies focus on correcting close matches from a TM using PBSMT, based on a set of learned edit operations (Hewavitharana et al., 2005). Ortega et al. (2016) propose a *patching approach* to correct TM matches with any kind of SMT system, and Espla-Gomis et al. (2015) a more translator-oriented method that offers *word keeping* recommendations based on information coming from an MT system. Example-based MT systems have also been used to leverage sub-segmental TM data (Simard and Langlais, 2001).

Of particular relevance are approaches that constrain a PBSMT system to use relevant parts of a fuzzy match (Zhechev and Van Genabith, 2010), for example by adding XML markup to Moses input (He, 2011; Koehn and Senellart, 2010; Ma et al., 2011) or by using a constrained word lattice (Li et al., 2016). Related to these are methods that augment the translation table of a PBSMT system with aligned spans from a retrieved TM match, yet without forcing the SMT system to incorporate (parts of) these aligned spans (Bicici and Dymetman, 2008; Simard and Isabelle, 2009). Alternatively, information from the fuzzy matches can also be integrated in the SMT system itself (Wang et al., 2013), for example using *sparse features* (Li et al., 2017). Recent studies focus on how to leverage TM information for NMT systems. These approaches work, for example, by imposing lexical constraints on the search algorithms used by NMT (Hokamp and Liu, 2017), by augmenting NMT systems with an additional lexical *memory* (Feng et al., 2017), or by explicitly providing the NMT system with access to retrieved TM matches (Gu et al., 2017).

M3TRA combines different elements from these approaches, which is its main novelty. In this paper we focus on (a) repairing close fuzzy matches, and (b) augmenting the MT input with information derived from the parallel corpus (the TM) used to train the MT system, thus constraining the translation of certain (parts of) sentences. We use a PBSMT system as basis for TM-MT integration because SMT allows a straightforward application of pretranslation (e.g. explicit alignment information is used in the process).

3 System architecture

M3TRA consists of four components: (a) a TM system, (b) a PBSMT engine, (c) a system for fuzzy repair (FR) and (d) a system for pretranslation span search (PSS). We elaborate on each of these components in the following sections. The sentence to translate can follow a number of routes, depending on the fuzzy match score of the best retrieved match and the success or failure of certain attempted operations (see Figure 1). First, FR is attempted for sentences that have at least one match which meets the relevant threshold (θ_{FR}). If FR is performed, it may modify the translation of the fuzzy match by deleting, inserting or substituting words. In case FR is not performed or fails, there are three options: (a) if the score of the highest match satisfies the TM threshold (θ_{TM}), the translation of the TM match becomes the final output, (b) if the score is between the TM and MT thresholds, PSS is attempted, and (c) if the score is below the MT threshold (θ_{MT}), or PSS fails (i.e. the query sentence as such becomes input to MT), the 'pure' MT output is used as final output.

Each of the four M3TRA components is described in detail below, followed by an overview of the parameter tuning process.

3.1 Translation Memory System

The TM is defined as a set $\mathcal{M}$ consisting of tuples of source and target sentences (s, t), i.e. translation units. Let q be the sentence to be translated (query sentence). It is looked up in the TM using a similarity function Sim, according to Equation 1, resulting in a set $\mathcal{M}_q$ of translation units the source sentence s of which is sufficiently similar to q, according to threshold θ_{Sim}. The best match for q is determined according to Equation 2.[2]

$$\mathcal{M}_q = \{(s,t) \in \mathcal{M} : Sim(q,s) \geq \theta_{Sim}\} \quad (1)$$

$$(s_b, t_b) = \underset{(s,t)\in\mathcal{M}_q}{\arg\max}\, Sim(q,s) \quad (2)$$

Matches are retrieved from the TM using two different similarity metrics: Levenshtein distance (Levenshtein, 1966) and METEOR (Lavie

[2] In case there are several matches with the same score, the first match encountered in the TM is taken as best match.

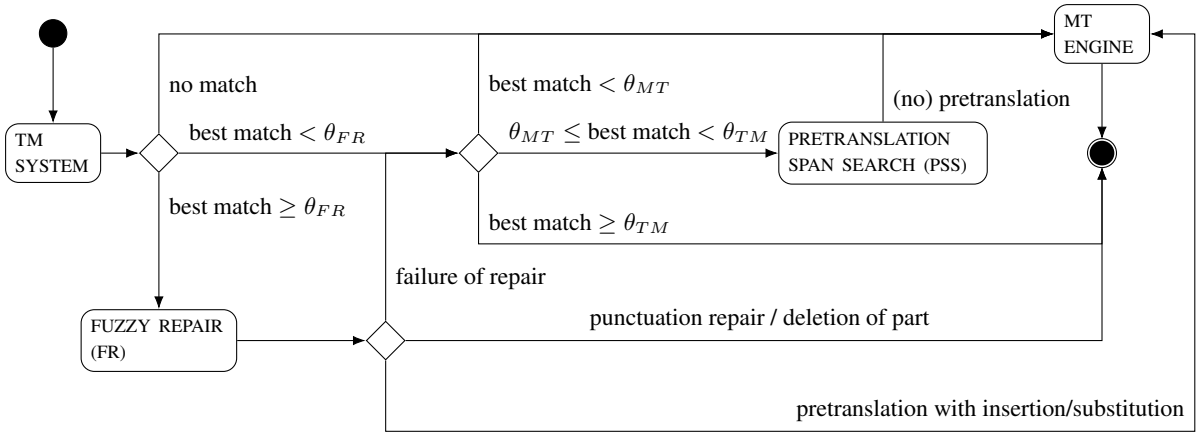


Figure 1: M3TRA workflow

and Agarwal, 2007). We limit the size of $\mathcal{M}_q$ to n, i.e. we only keep the tuples with the n best matches (plus any additional tuples with matches that have the same score as the nth best match).

As shown in Figure 1, we compare $Sim(q, s_b)$ to thresholds like θ_{FR} to decide whether to send q to FR or to PSS.

3.2 MT engine

We train a Moses PBSMT system (Koehn et al., 2007) from the TM sentence pairs.[3] We build a 5-gram KenLM language model, set the distortion limit to 6, and apply a maximal phrase length of 7.[4] During decoding, we set the maximum phrase length to 100. This is necessary to be able to pretranslate long word sequences using XML markup. The GIZA++ word alignment (using the grow-diag-final heuristic), the lexical probabilities and the principle of consistently aligned spans (Koehn, 2009) based on which the Moses phrase table is constructed are also used in the FR and PSS components (with an additional constraint, as explained later on).

3.3 Fuzzy repair

Let $\mathcal{M}_{FR}$ be the set of high-scoring translation units retrieved for q.[5] Three types of editing operations are attempted to arrive at the final output o: *substitution*, *deletion* and *insertion*. First, however, a number of specific operations aimed at repairing punctuation are performed.

Punctuation repair: since (simple) punctuation is arguably different from other linguistic phenomena, it is tackled by a dedicated subcomponent. We rank the tuples $(s, t) \in \mathcal{M}_{FR}$, according to $Sim(q, s)$, and iterate through the ranked list in order to verify whether simple punctuation issues can be resolved to produce o:

- if the only difference between q and s is due to casing, or one additional comma, we consider them as identical sentences, and set o to t; hence, we could say this is a type of 'void' repair;
- if q ends in punctuation,[6] and both s and t do not, we set o to t followed by the corresponding punctuation; if, however, t already contains punctuation in final position, we set o to t (another type of 'void' repair);
- if s and t end in punctuation, and q does not, we set o to t minus the final punctuation.

We stop iterating as soon as we produced o. In case of failure, we look at the more general mechanisms of substitution (sub), deletion (del) and insertion (ins). Since both del and ins can be considered more specific versions of sub (i.e. replacement of a part of s or t by the empty string), we focus on sub first.

Substitution: the basic idea behind the sub operation is to translate non-matching tokens of q and s in the context of tokens in t. sub is attempted when both q and s contain one sequence of one or more unmatched tokens q_i^j and $s_i^{j'}$ that end at potentially different positions j and j'. We check whether $s_i^{j'}$ is consistently aligned to a sequence

[3] Minus the development set used for tuning the parameters.
[4] These are 'default' settings.
[5] To limit potential negative effects of erroneously aligned translation units, $\mathcal{M}_{FR}$ is filtered by imposing a threshold on the percentage of aligned source tokens per translation unit.
[6] One of the tokens `. , ? ! : ; –`

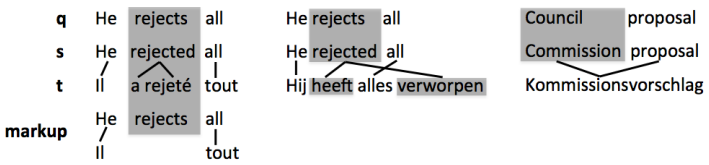


Figure 2: Examples of (attempted) substitution

t_k^l, i.e. whether each token in $s_i^{j'}$ is either aligned to a token in t_k^l or unaligned, and vice versa.[7] In addition, we impose the condition that the first and last token of $s_i^{j'}$ be aligned; the same goes for the first and last token of t_k^l. We assume that an alignment satisfying this condition, which we will call a *border-link alignment* in the remainder of this article, increases the likelihood of translation equivalence between sequences.

The sub operation is illustrated by the simplified examples in Figure 2. In the first example, both q and s contain a one-word sequence that is not shared (*rejects* and *rejected* respectively). In both cases, this sequence starts at the second position. The word *rejected* is aligned with the adjacent French target tokens *a* and *rejeté*, which in turn are only aligned with *rejected*. This allows for translating *rejects* in the context of *Il* and *tout*. In the second example, substitution fails since *rejected* is aligned with two Dutch target words, *heeft* and *verworpen*, which do not form an uninterrupted sequence. In the third example, substitution is impossible: $s_i^{j'}$ consists of *Commission*, which is aligned with *Kommissionsvorschlag*, while the German word is aligned with both *Commission* and *proposal*, the latter word not being part of $s_i^{j'}$.

To translate a span of q in the context of tokens of t, we proceed as follows. We block all retained tokens from t as pretranslation, by annotating q_1^{i-1} with the tokens of t_1^{k-1} using XML markup (unless $i = 1$), and annotating q_{j+1}^v with the tokens of t_{l+1}^w, unless j equals v; v and w stand for the number of tokens in q and t. The annotated q is then sent to the MT system, which translates q_i^j in the context of t_1^{k-1} and/or t_{l+1}^w (*Il* and *tout* in Figure 2).

To verify multiple potential substitutions, a *sliding window* is applied by a stepwise decrease of i and increase of j and j'. Each o resulting from a successful substitution is scored using the language model of the PBSMT system, in order to pick the best alternative o. The size of the sliding window is a model parameter. Two additional parameters[8] are put in place to limit the applicability of sub operations: a threshold for the maximum length of the span t_k^l and one for the maximum percentage of unaligned tokens within that span.

Deletion: the del operation consists of removing a sequence from t to yield o. If s is identical to q, apart from one additional sequence s_i^j (which may be a prefix, infix or suffix of s), and the latter has a border-link alignment with a target sequence t_k^l, the target sequence can be deleted. Two safeguard rules control the modification. If the token t_{k-1} is not aligned with a token in s, it is also deleted. The second rule is optional and ensures that t_k^l is not removed if it consists of only one token with less than 4 characters; [9] this leads o to be equal to t, which is another instance of 'void' repair.

The two safeguard rules are illustrated in Figure 3. In the leftmost example, the first occurrence of the Dutch word *de*, which precedes the sequence identified for deletion, is not aligned with any token in s. It is therefore also deleted. The rightmost example shows that the only difference between q and s is the token *the*, which has less than 4 characters. t is thus left unchanged.

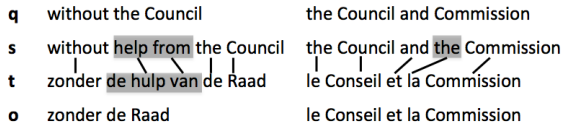


Figure 3: Examples of (attempted) deletion

Insertion: the ins operation can be performed when q is identical to s, apart from a sequence q_i^j (which may be a prefix, infix or suffix of q). Key to ins is determining where to insert the translation of q_i^j in t. For this to be possible, all of the following conditions need to be satisfied: (a) the token s_{i-1} is aligned to one or more tokens, the rightmost of which we call t_k, (b) s_i is aligned to

[7] With the understanding that at least one token in $s_i^{j'}$ is aligned.

[8] Added after a qualitative analysis of development set output.

[9] This heuristic was implemented to deal with articles in particular, in the absence of part-of-speech information.

one or more tokens, the leftmost of which we call t_l, and (c) k and l are adjacent (i.e. $l = k + 1$). If we found the insertion position k, we annotate q_1^{i-1} with the tokens in t_1^k, and annotate q_{j+1}^v with the tokens in t_{k+1}^w. This is illustrated in Figure 4. q contains an additional sequence compared to s (*European*), starting at the second position. We verify with which German word the first source token (s_{i-1}, *the*) is aligned, and with which word the second source token (*Parliament*) is aligned. As the aligned German words are adjacent, *the* can be annotated with *das* and *Parliament* with *Parlament*.

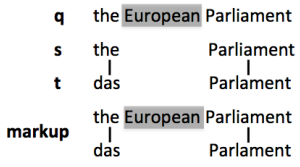


Figure 4: Example of insertion

If i is 1 (i.e. the non-matching part q_i^j is the prefix of the sentence), we apply a different procedure. If token s_1 is aligned with one or more target tokens, we annotate the sequence q_{j+1}^v with t_k^w, k being the position of the leftmost aligned token. If j is v (i.e. the non-matching part is the suffix of the sentence), and the last token of s is aligned to one or more target tokens, we annotate the sequence q_1^{i-1} with t_1^k, k being the position of the rightmost aligned token.

For any q that is not repaired and for which $Sim(q, s_b) \geq \theta_{TM}$, we set o to the most frequent t_b. Otherwise, q is sent to PSS.

3.4 Pretranslation span search

PSS consists of annotating (pretranslating) spans of q based on matches in $\mathcal{M}_q$, and subsequently constraining the MT system to respect the translations of these spans while producing o. PSS is applied in case the following condition is satisfied: $\theta_{MT} \leq Sim(q, s_b) < \theta_{TM}$ (see Figure 1). If so, a subset $\mathcal{M}_p$ is established according to Equation 3.

$$\mathcal{M}_p = \{(s,t) \in \mathcal{M}_q : Sim(q,s) \geq \theta_{PSS}\} \quad (3)$$

Based on the sentence pairs in $\mathcal{M}_p$, we define another set $\mathcal{P}_q$, which contains pretranslation tuples $(s, t, i, j, i', j', k, l)$. These are tuples for which all of the following conditions are valid: (a) the sentence pair belongs to $\mathcal{M}_p$, (b) q_i^j matches the source span $s_{i'}^{j'}$[10] and (c) $s_{i'}^{j'}$ has a border-link alignment with the target span t_k^l. A specific pair of source and target span may occur in multiple sentence pairs (see the frequency check below). Some of the tuples in $\mathcal{P}_q$ will be used for pretranslation, as described below.

Filtering pretranslation tuples: a tuple $p \in \mathcal{P}_q$ is filtered out if it satisfies one of the following conditions: (a) given all tuples $\mathcal{P}'_q \subseteq \mathcal{P}_q$ that involve the sentence pair of p, the total length of the source and target spans in $\mathcal{P}'_q$ does not satisfy a minimum length, (b) the length of the source and/or target span in p does not satisfy a minimum value, (c) the source and/or target span in p do not contain any content word (i.e. noun, adjective, verb or adverb), (d) the percentage of words aligned between the source and target span in p is too low, or (e) the one-to-many alignment score of p, defined in Equation 4, is too low. In this equation, y_x represents the number of tokens aligned to s_x, a token in the source span $s_{i'}^{j'}$ of p.

$$\frac{1}{j-i+1} \sum_{x=i}^{j} \frac{1}{y_x} \quad (4)$$

Combining pretranslation tuples: after filtering, each tuple $p \in \mathcal{P}_q$ is scored according to the weighted sum of (a) the length of the target span, (b) the frequency of the pair of source and target span, i.e. the number of tuples in $\mathcal{P}_q$ in which the pair occurs, and (c) the maximal fuzzy match score for the span pair, i.e. the maximal similarity $Sim(q, s)$ for all tuples in which the span pair occurs. The weights of the three above factors are model parameters. Subsequently, the tuples are ranked according to score, and used in the following iterative procedure. The spans of the first ranked tuple are used for pretranslation, i.e. the span t_k^l is used to annotate the q_i^j span. This tuple is removed from $\mathcal{P}_q$. The system then looks for the first ranked tuple in which the q_i^j span does not overlap with the already annotated span of q. This process is repeated until $\mathcal{P}_q$ only contains tuples with overlapping spans, or until the threshold for number of annotations has been reached. Figure 5

[10] Matching q to s given some similarity function leads to the identification of a number of matching parts. These parts are typically sequences which are identical in q and s. A matching span q_i^j refers to such a matching part, or one of its prefixes, infixes or suffixes. For instance, if two sentences have a matching part *The EC was*, matching spans include *The EC was*, *The EC*, *EC* etc.

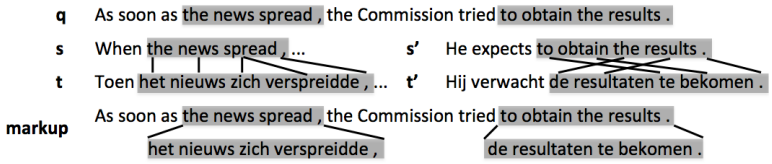


Figure 5: Example of pretranslation span search

provides an example of how two non-overlapping spans of a query sentence (*the news spread ,* and *to obtain the results .*) are pretranslated by two Dutch target spans (*het nieuws zich verspreidde ,* and *de resultaten te bekomen .*) originating from two different translation units. The PBSMT system is constrained to use these target spans in its final output.

3.5 Parameter setting and tuning

Many of M3TRA's components involve parameters (such as θ_{FR}) that can either be manually fixed or whose optimal value can be determined on the basis of an automated parameter tuning process. Initial tests were run on subsets of the development sets using random parameter initializations. Manual spot-checks of system outputs with different configurations were performed to verify the quality of the resulting translations (in comparison to pure MT output). To make the spot checks potentially more informative, differences in METEOR scores (compared to the MT baseline) were used as a criterion to select sentences with pretranslations that either led to large gains in translation quality or that appeared to result in worse translations.

In addition, a local hill-climbing algorithm was used to help determine the best parameter settings. The methodology followed here involved a stepwise narrowing of the search interval per parameter based on a combination of random initializations and runs of the hill-climber (with increasingly small step size). BLEU scores (Papineni et al., 2002) were used as tuning criterion.

4 Experimental design

This section describes the empirical tests that were carried out. We first describe the dataset and evaluation procedures, before turning to the results.

4.1 Data

We use the TM of the Directorate-General for Translation of the European Commission (Steinberger et al., 2013), for 5 language pairs in 2 directions: EN $\leftrightarrow$ NL, FR, DE, HU, PL.[11] To ensure consistency, we only use the cross-section of each of these datasets, resulting in 1.6 million sentence pairs per language combination. 2000 sentence pairs are set aside for development, and the test set consists of 3207 sentences.[12] We tokenized and lowercased all sentences before training Moses and tuning its parameters.

Table 1 shows the percentage of q's categorised on the basis of $Sim(q, s_b)$. For only 5 to 7% of q's no match is found in the TM. For the majority a match below 70% is retrieved, but for around 28-35% a high-scoring match ($> 70\%$) exists.

	None	<70	70-79	80-89	90-99
EN	5.9%	59.0%	9.4%	13.6%	12.1%
NL	5.0%	62.5%	8.9%	11.4%	12.3%
PL	6.7%	64.5%	8.0%	12.1%	8.7%
DE	6.3%	62.9%	9.6%	12.0%	9.2%
FR	4.5%	67.2%	9.3%	11.2%	7.8%
HU	6.6%	64.8%	8.7%	11.1%	8.9%

Table 1: Percentage of test sentences per match range

4.2 Baseline systems

We use three baselines to compare M3TRA with: (a) 'pure' TM matching, which involves selecting the (most frequent) t_b for q as o,[13] (b) the 'pure' Moses PBSMT system, and (c) a *standard* neural translation model.

For the neural MT model, we use OpenNMT (Klein et al., 2017) with default settings, i.e. a seq2seq RNN model with global attention consisting of 50000 words on the source as well as the target side, word embeddings of 500 dimensions, a hidden layer of 500 LSTM nodes, and learning through stochastic gradient descent with a learning rate of 1, and we ran the model for 20 epochs. We chose the best performing model, selected using a development set (different from the validation set)

[11] Note that the original source language may differ and that not all EC documents are translated directly.

[12] We were strict in filtering the test sets: any q for which a 100% match existed in any source language was left out for all language pairs.

[13] If no match is found in the TM, no translation is provided.

	EN-NL	EN-PL	EN-DE	EN-FR	EN-HU	NL-EN	PL-EN	DE-EN	FR-EN	HU-EN
θ_{TM}	0.79	0.87	0.79	0.83	0.70	0.79	0.93	0.71	0.72	0.70
θ_{FR}	0.77	0.63	0.55	0.54	0.39	0.52	0.57	0.53	0.49	0.40
Min % aligned tok FR	0.83	0.85	0.63	0.63	0.65	0.70	0.64	0.66	0.66	0.50
Window shift L	2	2	2	2	1	1	2	1	3	4
Window shift R	0	3	3	2	1	1	0	2	3	1
Max % non-aligned tok FR	0.50	0.42	0.74	0.24	0.72	0.53	0.75	0.48	0.44	0.67
θ_{PSS}	0.48	0.45	0.43	0.73	0.45	0.50	0.69	0.52	0.24	0.35
Min span length PSS	4	6	4	12	4	8	9	5	9	3
Min % aligned tok PSS	74	67	67	56	75	53	58	76	55	62
Min alignment score PSS	0.83	0.64	0.62	0.79	0.64	0.64	0.59	0.55	0.78	0.71

Table 2: Parameter settings after tuning

which was evaluated on BLEU, TER (Snover et al., 2006) and METEOR. The model that scored best on the majority of the metrics was chosen. When all three metrics differ, we chose the best scoring model according to BLEU.

4.3 Evaluation

BLEU scores are used as main evaluation criterion.[14] In addition, we report TER and METEOR scores to verify whether related yet different metrics point to similar trends. We only use one reference translation. To verify whether differences in BLEU scores between the baselines and M3TRA are statistically significant, we use the bootstrap resampling method described by Koehn (2004).

5 Results

5.1 Tuning

Table 2 provides an overview of the parameter settings that were found to lead to the highest BLEU scores on the development sets. We retained ten free parameters, the others were either fixed at certain values or disabled.[15] The results for METEOR as a fuzzy metric were found to be similar to the results using Levenshtein. For the current study, we decided to continue with Levenshtein as metric.

Looking more closely at the retained parameter settings, some observations can be made. First, θ_{TM} varies between 0.70 and 0.93. Second, the value of θ_{FR} lies between 0.39 and 0.77. Third, for any language pair at least half of the source tokens in a translation unit need to be aligned to perform FR. Fourth, for all language pairs, working with a sliding window for substitution was beneficial. Fifth, between 3 and 12 tokens per span are needed to provide beneficial pretranslations. Sixth, imposing restrictions on alignments proved to be positive for translation quality. Finally, the imposed threshold for minimum percentage of aligned words at source side varied between 50 and 83%.

5.2 Tests

Table 3 provides an overview of the evaluation scores for the ten language combinations of M3TRA compared to three baselines: pure TM, pure SMT, and NMT. For 9 of the 10 language combinations, M3TRA scores significantly better than the best baseline (SMT) in terms of BLEU. The increase in BLEU varies between 0.2 (for EN-PL; non-significant difference) and 5.47 points (for EN-HU). METEOR scores actually decrease for FR-EN, and are practically unchanged for EN-PL (+0.06). For EN-HU they increase with 3 points. TER scores consistently decrease for all language pairs. The decrease lies between 0.25 points (for EN-PL) and 5.33 points (EN-HU). Compared to the baseline SMT system, M3TRA affects between 9 and 39% of the sentences in the test set.

Looking at BLEU (see also Figure 6), baseline SMT also consistently outperforms baseline NMT, with the exception of EN-HU. With TER as evaluation criterion, NMT scores better for EN-HU and FR-EN. In terms of METEOR, SMT consistently outperforms baseline NMT. The quality of pure TM is estimated to be the lowest for all language pairs, which is not surprising, since e.g. a q for which $\mathcal{M}_q$ is empty is left untranslated.

Figure 7 presents the performance of the different systems for different subsets defined on the basis of $Sim(q, s_b)$ for one language pair (DE-EN).[16] With $Sim(q, s_b)$ below 70%, M3TRA does not lead to better scores compared to SMT. Pure

[14] We acknowledge that using BLEU is not ideal, especially when comparing SMT and NMT (Shterionov et al., 2017).

[15] θ_{Sim} = 0.2; n-best matches = 15; PSS weights: length = 0; frequency = 0.83; match score = 0.17.

[16] For reasons of space we restrict ourselves to one language pair. For the other languages, similar trends are observed.

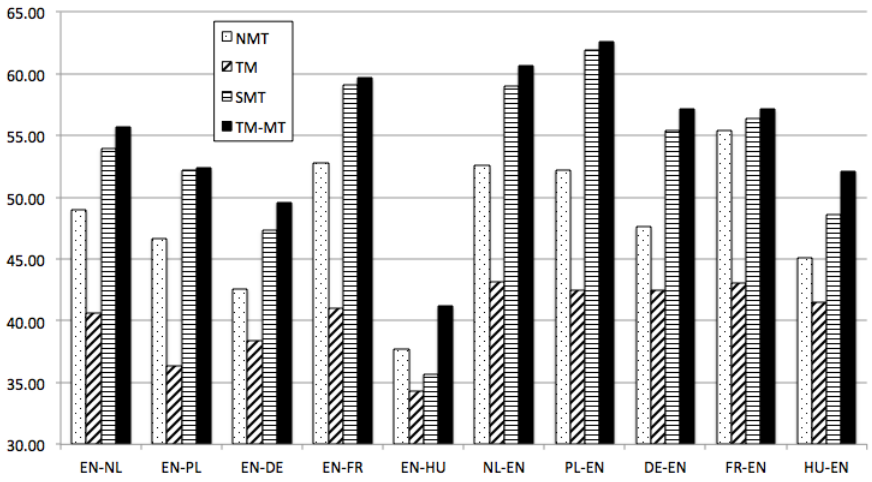


Figure 6: Overview BLEU scores

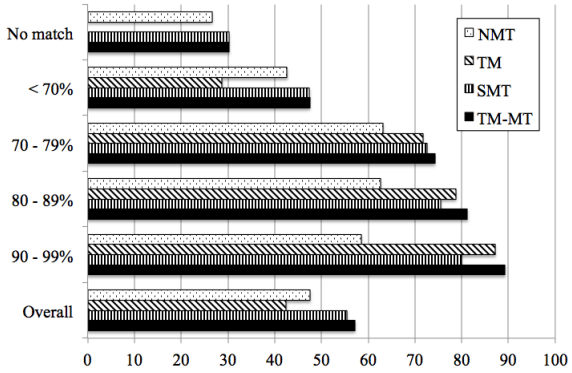


Figure 7: BLEU scores per match range (DE-EN)

TM starts scoring better than SMT in the range 80-89%. Thanks to FR, M3TRA also outperforms pure TM in the two highest match ranges.

6 Discussion

The main novelty of M3TRA is in its adaptable parameters, threshold values and safeguards, as well as in its combination of various features that are present in a number of approaches described in Section 2. Most notably, the use of XML markup to add pretranslation spans to input sentences is also used by He (2011), Koehn and Senellart (2010) and Ma et al. (2011). In M3TRA, Moses is constrained to include these pretranslated spans in the final output (the so-called exclusive mode is used). The fuzzy repair feature is closely related to the work of Ortega et al. (2016). Also the option to simply use TM target matches above a certain match score threshold has been implemented before (Simard and Isabelle, 2009). Moreover, by making use of the information obtained during the alignment process, M3TRA can be adapted easily to provide translators with information on the origin of parts of the proposed translations, possibly indicating which sentences should most likely be post-edited (Espla-Gomis et al., 2015). Finally, the combination of information from different fuzzy matches is also present in previous research (Wang et al., 2013; Li et al., 2016).

The test results show that integrating TM with MT can lead to better MT output, provided that sufficient high-scoring matches are retrieved from the TM. We argue that M3TRA is especially beneficial in a context with enough repetition and where the focus is (at least to a certain extent) on consistency and formulaic language use. Looking at the results for the different language pairs, the potential for improvement is highest for EN-HU and HU-EN,[17] which is most likely due to the (morphological) structure of the Hungarian language and its associated problems for (S)MT. The significant improvements for almost all language combinations indicate that M3TRA potentially works with different language families (Germanic, Romance, Finno-Ugric). The smallest improvement was found for the only Slavic language we tested (Polish).

With regard to the relatively low scores obtained by our NMT baseline, a number of comments are in order. First, we only tested certain standard/recommended settings in OpenNMT. It is likely that higher scores can be reached by tuning other NMT hyperparameters to better fit the dataset used. Second, SMT uses BLEU scores as tuning criterion, whereas in NMT perplexity is

[17] We realise one has to be careful when comparing BLEU scores across (target) languages.

		NMT	TM	SMT	TM-MT	Altered
EN-NL	*BLUE*	49.02	40.66	53.91	55.72**	25.5%
	TER	38.16	56.57	36.90	34.96	
	MET.	67.67	52.37	71.04	72.25	
EN-PL	*BLUE*	46.64	36.31	52.18	52.38	17.87%
	TER	39.57	60.85	37.79	37.54	
	MET.	35.45	26.39	38.67	38.73	
EN-DE	*BLUE*	42.57	38.37	47.32	49.59**	30.50%
	TER	44.81	59.13	44.43	41.95	
	MET.	55.56	45.05	60.11	61.71	
EN-FR	*BLUE*	52.76	41.00	59.08	59.65*	19.15%
	TER	35.79	57.63	32.96	32.22	
	MET.	67.31	50.16	72.97	73.45	
EN-HU	*BLUE*	37.75	34.33	35.71	41.18**	39.16%
	TER	48.01	61.72	55.31	49.98	
	MET.	55.23	45.66	55.67	58.67	
NL-EN	*BLUE*	52.55	43.17	59.00	60.63**	20.95%
	TER	35.11	55.13	32.32	30.56	
	MET.	41.65	30.28	44.95	45.51	
PL-EN	*BLUE*	52.21	42.49	61.95	62.57**	9.17%
	TER	35.28	55.54	29.42	28.86	
	MET.	42.17	29.94	46.60	46.85	
DE-EN	*BLUE*	47.59	42.50	55.44	57.17**	25.69%
	TER	39.90	55.73	36.49	34.67	
	MET.	38.70	30.17	43.05	43.46	
FR-EN	*BLUE*	55.42	43.11	56.39	57.12**	23.57%
	TER	32.42	55.14	35.33	34.23	
	MET.	44.02	30.37	45.81	45.70	
HU-EN	*BLUE*	45.09	41.51	48.62	52.10**	35.11%
	TER	43.35	56.13	44.25	40.37	
	MET.	37.51	29.60	40.10	40.93	
($p < 0.01$; ** $p < 0.001$)*						

Table 3: Results (significance tests for SMT vs TM-MT). Altered: % of sentences affected by TM-MT vs SMT

used to train the system. Third, BLEU evaluation focuses on precision (arguably the strength of SMT), and less on fluency (NMT's forte).[18] Finally, it is possible that SMT is more suited than NMT for contexts in which there is a considerable amount of repetition, and where adequacy and precision are crucial.

This study is limited in a number of ways: (a) the coverage of certain M3TRA components could still be improved, such as fuzzy repair, which could be extended to cover multiple edits per TM match or to also target non-sequential tokens, (b) only one dataset was used for testing, (c) only automatic metrics were used for evaluation, (d) BLEU scores were used for both training and testing, (e) no previously developed TM-MT integration method was used as baseline, and (f) the time spent on developing the NMT baseline was restricted. These limitations can be seen as suggestions for future research. For example, it would be interesting to see how professional translators appreciate M3TRA's output and indications of the origin of proposed translations, and what effect this has on translation efficiency. Some preliminary tests have been carried out (Coppers et al., 2018), but an in-depth study is still lacking. Such a study would also require us to take issues such as the positioning of formatting (and other types of tags) into consideration, which was outside the scope of the current paper. The same holds for a more qualitative evaluation of M3TRA's output (e.g. paying attention to certain morphological features).

[18] It can be argued, however, that BLEU scores are a good evaluation metric in a context in which precision is important.

7 Conclusions

We designed and tested a system for the integration of MT and TM, M3TRA, with a view to increasing the quality of MT output. M3TRA contains two main components, fuzzy repair and span pretranslation, which both make use of a TM with fuzzy matching techniques and an SMT system with related alignment information. The system uses the option to add XML markup to sentences sent to a Moses SMT system. Tests on ten language combinations using the DGT-TM dataset showed that it is clear that this approach has potential. Significantly higher BLEU scores for 9 of the 10 language combinations were observed, and METEOR and TER scores showed comparable patterns. In a next step, M3TRA has to be evaluated in an actual translation environment involving professional translators.

Acknowledgements

This research was done in the context of the SCATE project, funded by the Flemish Agency for Innovation and Entrepreneurship (IWT project 13007).

References

Biçici, E. and M. Dymetman. 2008. Dynamic translation memory: using statistical machine translation to improve translation memory fuzzy matches. *International Conference on Intelligent Text Processing and Computational Linguistics*, 454–465.

Bisazza, A. and M. Federico. 2016. A survey of word reordering in statistical machine translation: Computational models and language phenomena. *Computational Linguistics, 42*(2), 163-205.

Coppers, S., J. Van den Bergh, K. Luyten, I. van der Lek-Ciudin, T. Vanallemeersch and V. Vandeghinste. 2018. Intellingo: An Intelligible Translation Environment. *ACM conference on Human Factors in Computing Systems*, 1–13.

Espla-Gomis, M., F. Sánchez-Martínez and M.L. Forcada. 2015. Using machine translation to provide target-language edit hints in computer aided translation based on translation memories. *Journal of Artificial Intelligence Research, 53*(1), 169–222.

Feng, Y., S. Zhang, A. Zhang, D. Wang and A. Abel. 2017. Memory-augmented Neural Machine Translation. *arXiv preprint arXiv:1708.02005*.

Gu, J., Y. Wang, K. Cho and V.O. Li. 2017. Search Engine Guided Non-Parametric Neural Machine Translation. *arXiv preprint arXiv:1705.07267*.

He, Y. 2011. *The Integration of Machine Translation and Translation Memory*. Doctoral dissertation. Dublin City University.

Hewavitharana, S., S. Vogel and A. Waibel. 2005. Augmenting a statistical translation system with a translation memory. *10th Annual Conference of the European Association for Machine Translation*, 126–132.

Hokamp, C. and Q. Liu. 2017. Lexically Constrained Decoding for Sequence Generation Using Grid Beam Search. *arXiv preprint arXiv:1704.07138*.

Klein, G., Y. Kim, Y. Deng, J. Senellart and A.M. Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. *arXiv preprint arXiv:1701.02810*.

Koehn, P. 2004. Statistical significance tests for machine translation evaluation. *Proceedings of EMNLP 2004*, 388-395.

Koehn, P. 2009. *Statistical machine translation*. Cambridge: Cambridge University Press.

Koehn, P., H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, ... and C. Dyer. 2007. Moses: Open source toolkit for statistical machine translation. *45th annual meeting of the Association of Computational Linguistics*, 177–180.

Koehn, P. and J. Senellart. 2010. Convergence of translation memory and statistical machine translation. *2nd Joint EM+/CNGL Workshop Bringing MT to the User: Research on Integrating MT in the Translation Industry*, 21–31.

Lavie, A. and A. Agarwal. 2002. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. *2nd Workshop on Statistical Machine Translation*, 228–231.

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

Li, L., C.P. Escartín, A. Way and Q. Liu. 2017. Combining translation memories and statistical machine translation using sparse features. *Machine Translation, 30*(3), 183–202.

Li, L., A. Way and Q. Liu. 2016. Phrase-level combination of SMT and TM using constrained word lattice. *54th Annual Meeting of the Association for Computational Linguistics*, 275–280.

Ma, Y., Y. He, A. Way and J. van Genabith. 2011. Consistent translation using discriminative learning - A translation memory-inspired approach. *49th Annual Meeting of the Association for Computational Linguistics*, 1239-1248.

Ortega, J.E., F. Sánchez-Martínez, and M.L. Forcada. 2016. Fuzzy-match repair using black-box machine translation systems: what can be expected? *12th Biennial Conference of the Association for Machine Translation in the Americas*, Vol. 1, 27–39.

Papineni, K., S. Roukos, T. Ward and W.J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. *40th Annual Meeting of the Association for Computational Linguistics*, 311–318.

Shterionov, D., P. Nagle, L. Casanellas, R. Superbo and T. ODowd. 2017. Empirical evaluation of NMT and PBSMT quality for large-scale translation production. *20th Annual Conference of the European Association for Machine Translation*, 74–79.

Simard, M. and P. Isabelle. 2009. Phrase-based machine translation in a computer-assisted translation environment. *Machine Translation Summit XII*, 120–127.

Simard, M. and P. Langlais. 2001. Sub-sentential exploitation of translation memories. *Machine Translation Summit VIII*, 335–339.

Snover, M., B. Dorr, R. Schwartz, L. Micciulla and J. Makhoul. 2006. A study of translation edit rate with targeted human annotation. *Proceedings of the Association for Machine Translation in the Americas*, Vol. 200, No. 6.

Steinberger, R., A. Eisele, S. Klocek, S. Pilos and P. Schlüter. 2013. DGT-TM: A freely available translation memory in 22 languages. *arXiv preprint arXiv:1309.5226*.

Sudoh, K., K. Duh, H. Tsukada, T. Hirao and M. Nagata. 2010. Divide and translate: improving long distance reordering in statistical machine translation. *Joint Fifth Workshop on Statistical Machine Translation and Metrics*, 418–427.

Wang, K., C. Zong and K.Y. Su. 2013. Integrating translation memory into phrase-based machine translation during decoding. *51st Annual Meeting of the Association for Computational Linguistics*, 11–21.

Zhechev, V. and J. Van Genabith. 2010. Seeding statistical machine translation with translation memory output through tree-based structural alignment. *4th Workshop on Syntax and Structure in Statistical Translation*, 43–51.