View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Ghent University Academic Bibliography

Neural Fuzzy Repair: Integrating Fuzzy Matches into Neural Machine Translation

Bram Bulté
Centre for Computational
Linguistics (CCL)
KU Leuven
bram.bulte@ccl.kuleuven.be

Arda Tezcan
Language and Translation
Technology Team (LT3)
Ghent University
arda.tezcan@ugent.be

Abstract

We present a simple yet powerful data augmentation method for boosting Neural Machine Translation (NMT) performance by leveraging information retrieved from a Translation Memory (TM). We propose and test two methods for augmenting NMT training data with fuzzy TM matches. Tests on the DGT-TM data set for two language pairs show consistent and substantial improvements over a range of baseline systems. The results suggest that this method is promising for any translation environment in which a sizeable TM is available and a certain amount of repetition across translations is to be expected, especially considering its ease of implementation.

1 Introduction

Even though Machine Translation (MT) quality may have increased considerably over the past years, most notably with advances in the field of Neural Machine Translation (NMT), Translation Memories (TMs) still offer some advantages over MT systems. They are not only able to translate previously seen sentences 'perfectly' but they also offer 'near perfect' translation quality when highly similar source sentences are retrieved from the TM. As a result, in Computer-Assisted Translation (CAT) workflows, the MT system is often used as a backoff mechanism when the TM fails to retrieve high fuzzy matches above a certain threshold (Rossi and Chevrot, 2019; Federico et al., 2012), even though it has been shown that this basic integration method is not always the most optimal TM-MT combination strategy (Simard and Isabelle, 2009).

Our aim in this paper is to integrate the advantages of TMs into NMT systems in order to improve MT quality by utilizing existing translations for highly similar source sentences in a given TM. We propose a simple method for TM-NMT integration that is based on augmenting the source data with retrieved fuzzy TM targets by means of concatenation. We train both *dedicated* Neural Fuzzy Repair (NFR) systems that deal specifically with query sentences for which a (sufficiently high-scoring) match is found in the TM as well as *unified* systems capable of translating any query sentence. Several configurations are tested on the DGT-TM data set (Steinberger et al., 2013) for the language directions English into Dutch (EN→NL) and English into Hungarian (EN→HU).

In the next section, we provide an overview of previous research on TM-MT integration. Section 3 details the approach proposed in this paper. The experimental setup is presented in section 4, and the results in section 5. This is followed by the discussion (section 6) and conclusion (section 7).

2 Research background

The idea to combine the advantages of TM and MT is certainly not new. Early TM-MT integration approaches made use of example-based MT systems (Simard and Langlais, 2001) or focused on editing high-scoring TM matches (Hewavitharana et al., 2005). Editing TM matches (or *fuzzy repair*) proved to be beneficial for the quality of MT output, as demonstrated in later studies that also implemented such an approach (Ortega et al., 2016). Alternatively, phrase-based statistical MT (PBSMT) systems have been augmented with TM information by constraining the output to contain (parts of) retrieved TM matches (Koehn and Senellart, 2010a), by enriching the system's phrase table (Biçici and Dymetman, 2008; Simard and Isabelle, 2009), or by adapting the PBSMT system itself (Wang et al., 2013), all leading to significantly better performance.

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1800–1809
Florence, Italy, July 28 - August 2, 2019. ©2019 Association for Computational Linguistics

More recently, with the rise of NMT, researchers focused on ways to incorporate TM information in neural MT architectures. For example, this has been attempted by means of a lexical memory added to the NMT system (Feng et al., 2017), lexical constraints imposed on the NMT search algorithms (Hokamp and Liu, 2017), *rewards* attached to retrieved and matched *translation pieces* that guide the NMT output (Zhang et al., 2018), by explicitly providing the NMT system with access to a list of retrieved TM matches during decoding (Gu et al., 2018), or by adding an extra encoder for retrieved TM matches (Cao and Xiong, 2018). In all cases, this resulted in impressive gains in estimated translation quality.

All of these TM-NMT integration approaches either alter the search algorithms at decoding or change the architecture of the NMT system by combining information from multiple encoders. Our method is different in that it only involves a change in data preprocessing, without altering the NMT system itself. The proposed change at preprocessing is inspired by research on Automatic Post-Editing (APE) of MT output as well as multi-source machine translation. In the context of APE, NMT engines have been trained with a concatenation of source sentence and MT output at the source side, with a specific *break* token separating the two strings (Hokamp, 2017). A similar simple concatenation approach has also been used to take advantage of multiple source languages to increase the quality of NMT output (Dabre et al., 2017). In both cases, the NMT systems managed to process these augmented inputs successfully.

In the next section, we describe the TM-NMT integration approach followed in this paper.

3 Neural Fuzzy Repair

We present a simple approach to TM-NMT integration, based on augmenting source sentences with fuzzy matches retrieved from a TM, and training *dedicated* or *unified* NMT systems. First, we present the TM system and method for fuzzy match retrieval. We then describe how we augment the input that is used to train an NMT system, which is presented next.

3.1 TM and Fuzzy match retrieval

Our TM consists of any set $\mathcal{M}$ of source and target sentence pairs (S, T); the same sentences that would be used as training data for an MT system. Each source sentence $s_i \in S$ is compared to all other source sentences $s_j \in S$ using a similarity metric Sim. The fuzzy source sentences $S'_i \in S$ that match a given source sentence s_i with a similarity score higher than the specified threshold λ are stored in the set $\mathcal{F}_{s_i}$ together with their corresponding target sentences $T'_i \in T$ ($Sim(s_i, s_j) \geq \lambda$). Perfect matches ($Sim(s_i, s_j) = 1$) are excluded from $\mathcal{F}_{s_i}$.

We use token-based edit distance (Levenshtein, 1966) as primary match metric for the tests in this paper[1], based on the work of Hyyrö (2001). Since extracting fuzzy matches from a large TM using edit distance is computationally costly[2], we attempt to speed up this process in three ways. First, for each source sentence we extract candidates using the *SetSimilaritySearch*[3] library for Python and calculate *editdistance* only on the extracted candidates (*sss+ed*). *SetSimilaritySearch* offers a vector similarity search algorithm based on indexing and optimization strategies that does not rely on approximation methods, and offers performance gains over a number of inverted list-based approaches and signature-based methods (Bayardo et al., 2007). To extract candidates for high fuzzy matches with *SetSimilaritySearch*, we use the similarity measure $containment_{max}$, which is defined as follows:

$$containment_{max}(v_i, v_j) = \frac{\|v_i \cap v_j\|}{max(\|v_i\|, v_j\|\|)}$$

where v_i and v_j are two vectors consisting of unique tokens obtained from two sentences s_i and s_j, respectively. Second, we only calculate the *editdistance* score for the *n-best* candidates extracted by *SetSimilaritySearch* (*sss_nbest+ed*). Finally, we use multi-threading (*sss_nbest+ed(mt)*).

In Section 5.1 we evaluate what impact these three techniques have on the speed of retrieval and the number of matches retrieved.

3.2 Source augmentation

For each source sentence s_i for which at least one sufficiently high-scoring match is found in the TM (i.e. $\mathcal{F}_{s_i} \neq \emptyset$), an augmented source x_i is generated according to one of the following formats,

[1] https://github.com/aflc/editdistance. This metric can be replaced by other alternatives in the literature (Bloodgood and Strauss, 2015).

[2] Extracting fuzzy matches for all source sentences in a data set consisting of 20K sentences took roughly 1 hour (3996 seconds) on a 2.50GHz Intel Xeon E5 core.

[3] https://github.com/ardate/SetSimilaritySearch

while preserving the original target sentence t_i:

- format 1: $x_i : s_i$ @@@ t'_1
- format 2: $x_i : s_i$ @@@ t'_1 @@@ t'_2
- format 3: $x_i : s_i$ @@@ t'_1 @@@ t'_2 @@@ t'_3

where t'_1 represents the target side of the highest scoring match s'_1 in $\mathcal{F}_{s_i}$, and t'_2 and t'_3 the target side of the second and third highest scoring matches s'_2 and s'_3, respectively. We use '@@@' as *break* token marking the boundary between two sentences.

For formats 2 and 3, in case $\mathcal{F}_{s_i}$ does not contain at least either 2 or 3 elements, the corresponding empty slots are left blank. Each augmented source x_i, coupled with its original target sentence t_i taken from $\mathcal{M}$, is stored in the new set $\mathcal{M}' = (X, T)$.

In addition to using format 1 as described above, we also test an alternative configuration 'format 1 n-best', in which we include augmented-source/target pairs (X^n, T) in $\mathcal{M}'$ by utilizing the n-best matches for a given s_i. For example, with this alternative configuration, when $n = 3$, $\mathcal{X}^n$ contains the following augmented source for each s_i, which are paired with the original target sentence t_i.:

- format 1 n-best:

  $x_i^1 : s_i$ @@@ t'_1

  $x_i^2 : s_i$ @@@ t'_2

  $x_i^3 : s_i$ @@@ t'_3

This alternative configuration only affects the training set $\mathcal{M}'$ and does not change the way test sentences are handled. For all different values of n, the source sentences in the test set are augmented with the translation of the best possible fuzzy match t'_1.

The different data augmentation strategies described above potentially lead to different sizes of training data sets (see Section 5.2).

3.3 NMT system

We use OpenNMT (Klein et al., 2017) with close to standard settings to train our NFR systems. For example, we kept the default optimizer (sgd), learning rate (1.0), word embedding size (500 for source and target), batch size (64) and dropout probability (0.3). We did, however, change a number of parameters related to data preprocessing and training. The maximum source and target length at preprocessing are set to 300 and 100, respectively, and the source vocabulary size is doubled to 100K (since the augmented source input $\mathcal{X}$ are bilingual). We train seq2seq bidirectional RNN models with global attention, and increased the hidden LSTM layer nodes to 750 (from 500), training steps to 200K (from 100K) and learning rate decay to 0.8 (from 0.5).

3.4 Integration

Two methods for integrating the augmented training set $\mathcal{M}'$ in the NMT workflow are tested based on the different formats described in Section 3.2. We create:

- two separate NMT systems, a *backoff* NMT system with $\mathcal{M}$ as training data and a *dedicated* NFR system with only $\mathcal{M}'$ as training data, or
- one *unified* NFR system that uses the union of sets $\mathcal{M}$ and $\mathcal{M}'$ as training data.

We retrieve fuzzy matches for each query sentence q_i in the test set Q, by comparing them to each s_j in the training set $\mathcal{M}$ in line with the method described under 3.1. In case at least one match is found for which $Sim(q_i, s_j) \geq \lambda$, an augmented query input y is generated according to the method described under 3.2. As the *dedicated* system is only capable of translating y, it is combined with a *backoff* system capable of translating q, in order to translate all source sentences in a given test set. On the other hand, the *unified* system, which can be considered a simpler alternative to the *backoff* integration method, can translate both q and y.

4 Experimental setup

In this section we describe the baseline systems our NFR systems are compared with, the data, and evaluation.

4.1 Baseline systems

We compare the NFR systems to five baselines: (a) a standard NMT model, (b) a phrase-based SMT system, (c) TM matching, (d) a previously developed hybrid TM-SMT system (Bulté et al., 2018), and (e) Google Translate[4].

The baseline NMT system is the *backoff* NMT system with $\mathcal{M}$ as training data as described in

[4] February, 2019.

Section 3.4. As SMT baseline we train a Moses engine (Koehn et al., 2007) with the sentence pairs in $\mathcal{M}$, using standard settings[5]. TM matching simply means selecting the highest scoring TM target t'_1 for each query sentence q_i. Finally, we include Google Translate as an example of a widely used NMT system, which is not trained with domain-specific data, unlike the other baseline systems.

4.2 Data

We use the TM of the Directorate-General for Translation of the European Commission (Steinberger et al., 2013) for two language pairs: English into Dutch and English into Hungarian. All sentences were tokenized using the Moses toolkit as well as lowercased prior to training. We randomly divide the data into a training set (approx. 2.4M sentence pairs), two development sets (3000 sentence pairs each) and a test set (3207 sentences). The first development set is used for validation during training of the NMT systems and for tuning the SMT systems; the second development set is used to test the performance of different NFR configurations. Test sentences for which a perfect match was found in either the training or one of the development sets were removed. We ensured that the source side for all data sets was identical for both language pairs.

We use pure token-based *editdistance* to extract fuzzy matches for the source sentences in the two development sets and the test set, considering their relatively small size. We use *editdistance* with candidate selection using *SetSimilaritySearch* to extract matches in the training set (see Section 3.1). Table 1 shows the percentage of query sentences in the test set for which fuzzy matches are found in different match ranges (i.e. <50, 50-59 ... 90-99). Since the source sentences in the test and training sets are the same for both language pairs, the values apply to both EN-NL and EN-HU.

< 50	50-59	60-69	70-79	80-89	90-99
41.3%	11.4%	10.3%	8.8%	14.2%	14.0%

Table 1: Percentage of test sentences per fuzzy match range (n=3207).

For 58.7% of the sentences in the test set a match of 50% or higher was found in the TM, with proportionally most matches occurring in the highest match ranges.

< 50	50-59	60-69	70-79	80-89	90-99
32.9	21.0	21.1	24.4	23.4	33.1

Table 2: Average number of source tokens per sentence, per fuzzy match range.

Table 2 shows the average number of source tokens per sentence for each fuzzy match range. On average, the longest sentences are found at both ends of the fuzzy match scale, i.e. the highest match range and the subset of sentences without fuzzy match higher than 50%, with approximately 33 tokens per sentence. In the other match ranges, sentences are around 10 tokens shorter.

4.3 Evaluation

Three automated evaluation metrics are used: BLEU[6] (Papineni et al., 2002), TER[7] (Snover et al., 2006), and METEOR[8] (Lavie and Agarwal, 2007). There is one reference translation per test sentence. BLEU scores are used as the primary evaluation metric, and the significance of performance differences in terms of BLEU scores between systems is tested using bootstrap resampling (Koehn, 2004). All evaluations are carried out on tokenized data.

5 Results

In this section we describe the impact of our fuzzy matching technique on the speed of retrieval and the quantity of retrieved matches (5.1), the outcome of the NFR system selection (5.2), the final results on the test set (5.3), as well as the effect of the size of the TM on the performance of the NFR system (5.4).

5.1 Fuzzy match retrieval

Table 3 shows the fuzzy match extraction time for four different approaches, as defined in Section 3.1, on three different sizes of data sets. To analyze the fuzzy matching speed of these different approaches, we extracted a maximum of 5 fuzzy matches for each source sentence and used $\lambda = 0.5$ as threshold for both *editdistance* and *SetSimilaritySearch*. Relatively small subsets (randomly extracted 5K, 10K and 20K sentence pairs) of the original training data were used for these tests. The table also shows the relative fuzzy matching

[5] 5-gram KenLM, distortion limit = 6, max. phrase length = 7.

[6] Moses *multi-bleu.perl* script.

[7] Version 0.7.25: https://github.com/snover/terp

[8] Version 1.5: https://www.cs.cmu.edu/∼alavie/METEOR/

speed of the three different methods compared to *editdistance* alone, on the data set containing 20K sentence pairs (%20K).

Method	5K	10K	20K	%20K
ed	303	1071	3996	100%
sss+ed	15	54	158	3,95%
sss_n20+ed	7	27	100	2,50%
sss_n20+ed(16t)	1	3	10	0,25%

Table 3: Fuzzy matching speed (seconds) on 5, 10 and 20 thousand sentence pairs using four different methods. *n20* refers to 20-best candidates and *16t* to multi-threading with 16 threads.

By using the three techniques described in Section 3.1, we reduced the fuzzy matching time on the training set to 0,25% of the time it takes to extract matches using only *editdistance* on the 20K data set. Using the *sss_nbest+ed(mt)* method, we extracted all fuzzy matches for all source sentences per training set described in Section 4.2 in approximately 24 hours[9].

While taking *n-best* match candidates reduces the number of *editdistance* calculations, depending on the value of *n*, it also potentially leads to a loss of training data. Table 4 provides the percentage of source sentences for which no fuzzy matches are found above the *editdistance* threshold of 0.5 using three different matching methods.

Method	5K	10K	20K
ed	78,86%	75,88%	71,56%
sss+ed	78,86%	75,88%	71,56%
sss_n20+ed	78,92%	75,97%	71,73%

Table 4: Percentage of source sentences without fuzzy matches above the *editdistance* score of 0.5, in sets of 5, 10 and 20 thousand sentence pairs.

The results in Table 4 indicate that calculating *editdistance* only on the candidates extracted by *SetSimilaritySearch* does not lead to data loss in these three data sets. Limiting the candidate list to 20-best candidates, however, slightly increases the number of sentences for which no fuzzy matches are found. Even though the increase seems minimal for these three relatively small data sets (i.e. 0,06%, 0,09% and 0,17% for 5, 10 and 20 thousand sentence pairs respectively), there is an increasing trend with increasing data size.

5.2 NFR system selection

We use the second development set to test different NFR configurations. For the sake of these tests, we fix the minimum fuzzy-match threshold λ to 0.5. Six different dedicated NFR systems and three unified systems are compared. We test two parameters: the augmented input format (F1-F3), and the n-best matches included per source sentence using format 1 (F1 n-best 1-3), as described in Section 3.2. The best-scoring NFR systems are selected for the final evaluation on the basis of the test set.

Table 5 provides the results of the evaluation on the second development set for the baseline systems and the dedicated and unified NFR systems for both language pairs. Here we only consider the subset of sentences for which a match was found in the TM with a match score higher than 0.5 (2266 sentences), and only look at BLEU scores. Table 5 also shows the size of the training set for each system configuration, given that the different configurations lead to training data sets of varying sizes (see Section 3.2).

	BLEU scores		Train
System	EN-NL	EN-HU	set
Baseline NMT	64.16	53.52	2.4M
Baseline SMT	68.99	46.41	2.4M
Baseline TM	69.92	60.12	-
Google Translate	49.84	39.55	N/A
Dedicated F1 1-best	79.22	68.35	1.8M
Dedicated F1 2-best	78.95	68.25	3.2M
Dedicated F1 3-best	78.70	**68.77**	4.5M
Dedicated F2	79.31	68.69	1.8M
Dedicated F3	**79.33**	68.45	1.8M
Unified F1 1-best	78.59	67.35	4.2M
Unified F2	78.96	67.56	4.2M
Unified F3	79.06	67.65	4.2M

Table 5: BLEU scores on the development set for sentences with at least one fuzzy match above the threshold of 0.5, and size of training data set, per system.

For EN-NL, all NFR systems score between 8.35 and 9.41 BLEU points higher than the best baseline system (TM) for this subset of sentences. Only 0.74 BLEU points separate the worst and the best performing NFR system. *Dedicated F3* obtained the best BLEU score, closely followed by *Dedicated F2*. *Unified F3* also slightly outperforms the other unified systems trained with the second and the first data format.

Also for EN-HU there is only 1.42 BLEU points difference between the worst and best scoring NFR system. Here, the best NFR system out-

[9] Using 2000-best candidates and 16 threads.

performs the best baseline (TM) by 8.65 BLEU points. We note that the TM baseline in itself scores 6.6 BLEU points higher than the best MT baseline (NMT). The dedicated NFR system *F1 3-best* attains the highest BLEU score.

5.3 Test set evaluation

Table 6 contains the results for EN-NL for the entire test set (3207 sentences). The dedicated NFR + NMT backoff approach outperforms all baseline systems, scoring +3.19 BLEU, -3.6 TER and +1.87 METEOR points compared to the best baseline (TM-SMT). Compared to the NMT baseline, the difference is 7.46 BLEU points. The best unified NFR system (*Unified F3*) scores only slightly worse than the approach with a dedicated NFR system and NMT backoff. Both NFR systems score significantly higher than the best baseline in terms of BLEU ($p < 0.001$). We note that the baseline SMT outperforms the baseline NMT, which in turn obtains better scores than Google Translate on this data set.

System	**BLEU**	**TER**	**MET.**
Baseline NMT	51.45	36.21	69.83
Baseline SMT	54.21	35.99	71.28
Baseline TM-SMT	55.72	34.96	72.25
Google Translate	44.37	41.51	65.07
Best NFR + NMT backoff	**58.91**	**31.36**	**74.12**
Best NFR unified	58.60	31.57	73.96

Table 6: Test results EN-NL (all sentences).

The results for EN-HU (Table 7) show a similar overall picture, with an even clearer advantage for the NFR systems. The best dedicated NFR system with NMT backoff (*Dedicated F1 3-best*) scores 7.06 BLEU points more than the best baseline (TM-SMT), and also yields considerable improvements in terms of TER (-5.34) and METEOR (+4.46). The unified NFR system scores only 0.41 BLEU points lower than the dedicated NFR+backoff system. Also for this language pair the differences in BLEU scores between both NFR systems and the best baseline system are statistically significant ($p < 0.001$). The TM-SMT system is the best baseline in terms of BLEU and METEOR (but not in terms of TER, with the baseline NMT system scoring over 4.5 points better). In contrast to the EN-NL tests, where the SMT system scored better than the NMT system, the baseline NMT for EN-HU obtains a higher translation quality than the SMT baseline. Moreover, Google Translate gives comparable results to those of the baseline SMT system (better in terms of TER but worse in terms of BLEU and METEOR).

System	**BLEU**	**TER**	**MET.**
Baseline NMT	40.47	45.45	57.68
Baseline SMT	33.65	54.76	53.96
Baseline TM-SMT	41.18	49.98	58.67
Google Translate	32.11	52.99	51.40
Best NFR + NMT backoff	**48.24**	**40.11**	**63.13**
Best NFR unified	47.83	40.14	62.77

Table 7: Test results EN-HU (all sentences).

Next we look at the performance of the different systems on different subsets of the test set classified according to the best fuzzy match score (Table 8). For EN-NL, both NFR systems outperform all baselines in all match ranges from 0.6 onward. In the match range 0.5-0.59, the SMT and TM-SMT baselines obtain higher BLEU scores than both NFR systems. For EN-HU, the NFR systems outperform all baselines in all match ranges except for *No match*. The scores of both NFR systems for both language pairs consistently increase across increasing match ranges, a pattern which is also followed by the TM baseline. We note that the NFR systems, also in the highest match range, clearly outperform the TM baselines for both language pairs.

If we disregard the TM and TM-SMT baselines and only look at the 'pure' MT baselines, the difference between the NFR systems and the MT baselines consistently becomes larger with increasing fuzzy match score, for both language pairs. In the highest match range (i.e. 0.9 - 0.99), the increase in BLEU scores compared to the NMT baseline is 21.95 points for EN-NL and 22.76 points for EN-HU. In the range 0.8 - 0.89 this is 15.68 and 17.7 BLEU points respectively. As Table 2 showed, there is no correlation between fuzzy match range and average sentence length, which means that decreasing average sentence length is not an explanation for the increasing performance of the NFR systems with increasing fuzzy match scores. The results suggest that from a fuzzy match score of between 0.5 and 0.6 onward, it becomes advantageous to use an NFR system using the data sets in this study.

For those sentences in the test set for which no match higher than the given threshold ($\lambda \geq 0.5$) was found in the training set (*No match*), the unified NFR system performs slightly worse than the best baselines for translation into both Dutch (-0.34 BLEU) and Hungarian (-0.74 BLEU). Note

	System	EN-NL	EN-HU
No match	Baseline NMT	40.77	**29.76**
	Baseline SMT	**40.87**	23.21
	Baseline TM-SMT	39.71	23.49
	Google Translate	39.02	27.3
	Best NFR unified	40.53	29.02
0.5-0.59	Baseline NMT	51.14	39.23
	Baseline SMT	**54.11**	33.50
	Baseline TM	34.23	28.38
	Baseline TM-SMT	53.86	34.67
	Google Translate	43.24	32.16
	Best NFR dedicated	50.21	40.28
	Best NFR unified	51.55	**42.61**
0.6-0.69	Baseline NMT	56.72	44.73
	Baseline SMT	61.86	39.07
	Baseline TM	49.56	40.67
	Baseline TM-SMT	61.75	41.81
	Google Translate	49.82	35.32
	Best NFR dedicated	**65.31**	52.13
	Best NFR unified	63.76	**53.14**
0.7-0.79	Baseline NMT	57.59	45.75
	Baseline SMT	64.84	40.54
	Baseline TM	61.52	49.39
	Baseline TM-SMT	66.22	48.82
	Google Translate	46.79	36.32
	Best NFR dedicated	**73.12**	**59.29**
	Best NFR unified	72.78	57.73
0.8-0.89	Baseline NMT	67.01	55.91
	Baseline SMT	71.14	47.69
	Baseline TM	69.66	61.89
	Baseline TM-SMT	70.28	60.81
	Google Translate	52.89	38.54
	Best NFR dedicated	**82.69**	73.27
	Best NFR unified	82.09	**73.61**
0.9-0.99	Baseline NMT	65.95	56.16
	Baseline SMT	71.49	47.12
	Baseline TM	83.77	74.67
	Baseline TM-SMT	83.49	75.24
	Google Translate	50.92	38.29
	Best NFR dedicated	**87.90**	**78.92**
	Best NFR unified	87.41	77.59
All ≥ 0.5	Baseline NMT	61.28	50.28
	Baseline SMT	66.27	43.09
	Baseline TM	64.63	55.94
	Baseline TM-SMT	70.19	56.89
	Google Translate	49.38	36.66
	Best NFR dedicated	**75.31**	**64.85**
	Best NFR unified	74.96	64.78

Table 8: Test results (BLEU scores, different match ranges).

that for this subset of test sentences the performance of the different MT systems is highly comparable for EN-NL. In this match range, for example, also Google Translate scores only 1.85 BLEU points lower than the best-scoring system (SMT). For EN-HU, SMT is clearly outperformed by both NMT and Google Translate in the *No match* range.

5.4 Effect of TM size

Considering that the success of the NFR systems depends on the amount of highly-similar matches retrieved from the TM, we examine the effect of different TM sizes by evaluating the performance of baseline NMT and the best unified NFR system for increasingly smaller subsets of our original EN-NL data set. Figure 1 shows the translation quality for the baseline NMT and the best unified NFR system (*Unified F3*) for five different TM sizes, which are indicated as percentages of the original TM size (i.e. approx. 2.4M sentence pairs)[10], as well as the percentage of source sentences in the test set for which similar sentences are retrieved above the similarity threshold ($\lambda \geq 0.5$).

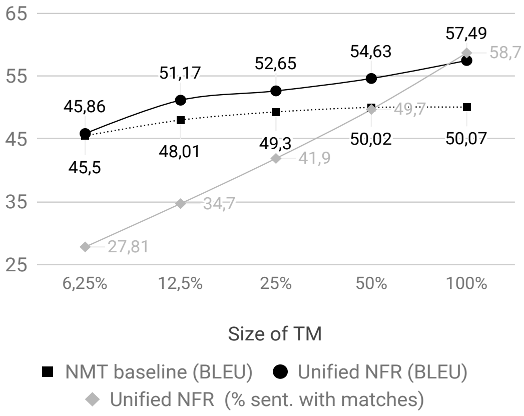

Figure 1: Effect of TM size on translation quality (BLEU) and number of 'similar' matches retrieved from TM.

The NFR system outperforms the baseline NMT system for all TM sizes. The difference in BLEU scores between the two systems becomes more outspoken starting from 12.5% of the original TM size (i.e. approx. 300K sentence pairs), when for 35% of the sentences in the test set a similar match is retrieved from the TM. We note that the NFR system built with 12.5% of the original TM size yields higher BLEU scores than the baseline NMT system trained with the full TM (51.17 vs. 50.07).

6 Discussion

The results of this study confirm that integrating TM information in NMT systems can result in significantly better translation quality, as demonstrated in a number of previous studies (Cao and

[10] In this experiment we used 100K steps (instead of 200K steps) to speed up training, which led to a slight decrease in BLEU scores for the systems built using the original TM (100%).

Xiong, 2018; Hokamp and Liu, 2017; Gu et al., 2018; Zhang et al., 2018). The main novelty of our approach is that it only involves data preprocessing, without altering the architecture (e.g. by adding additional encoders) or algorithms of the NMT system. This makes our method easy to implement, since it is compatible with any 'standard' or out-of-the-box NMT system. This should allow for a smoother implementation and wider adoption.

The NFR systems proposed in this study not only outperform all MT baselines, they also obtain better scores than the TM baseline in all fuzzy match ranges (including the highest ones). This shows that the NFR systems not only successfully exploit the information from TM matches, but go beyond this and effectively succeed in 'repairing' the fuzzy matches, at least to a certain extent. We argue that, for this reason, NFR systems (or, more generally speaking, systems offering NMT-TM integration) might gradually replace TM retrieval in CAT workflows in the future, where MT is currently still often used as a backoff option (Rossi and Chevrot, 2019; Federico et al., 2012). The fact that the MT baselines in our study do not obtain better scores than 'pure' TM retrieval in the higher match ranges (i.e. 0.8-0.99 for EN-NL and 0.7-0.99 for EN-HU) appears to confirm why this is still the case. Moreover, it is possible that NFR systems help to lower the resistance some translators have to adopting MT (Cadwell et al., 2018), especially when the TM-origins of parts of the MT output are marked by using automatic word-alignment methods (Bulté et al., 2018), since this could potentially increase translators' confidence in the quality of automatically generated translations.

Even though we only performed a limited number of tests on one data set, the results show that the NFR system is successful for two language pairs, EN-NL and EN-HU, in spite of the typological differences between the two target languages. Moreover, the results of the system selection procedure reveal that the NFR system is rather robust, in that different configurations yield comparable results, and all lead to significant improvements in estimated translation quality. While combining the dedicated NFR with the baseline NMT systems yielded the best results for both language pairs, the unified NFR systems achieve comparable BLEU gains over the baseline NMT systems. As a result, the unified NFR systems offer yet a simpler alternative to the baseline NMT systems due to their ability to translate all source sentences.

The analyses per match range reveal that using an NFR system starts being advantageous with fuzzy match scores between 0.5 and 0.6. It seems logical that any TM-based method is only suited for contexts with a sizeable TM and with a certain expected degree of repetition and overlap in the data. The tests related to training data size show, however, that with smaller TMs this method is still beneficial. For example, an NFR system built only with 1/8th of the original data set still achieved higher BLEU scores than the baseline NMT system trained on the full data set. We can argue that the most important factor for the NFR systems proposed in this study is the amount of overlap between the training and query sentences.

Looking at the performance of the baseline MT systems, and in particular the relationship between SMT and NMT, there is a clear difference between the two target languages. The EN-NL SMT outperforms NMT by almost 3 BLEU points when evaluating the complete test set and obtains better scores in each of the match ranges (Table 6). The opposite is true for EN-HU, for which the NMT baseline outperforms the SMT baseline by almost 7 BLEU points on the whole test set (Table 7), a trend which is also visible in all match ranges (Table 8). Our findings are in line with those of Koehn et al. (2009), who compare the SMT quality of 462 language pairs and report generally lower SMT quality when translating into morphologically rich languages, such as Hungarian, Finnish and Estonian. The poorer translation quality of the EN-HU SMT in this study can potentially be attributed to the fact that a rich morphology (involving inflections and derivations) leads to an increase in vocabulary size and an overall data sparsity problem, which brings about additional challenges to the'standard' phrase-based SMT systems that rely on explicit phrase alignments on surface forms (Koehn, 2009). Instead of relying on surface forms, NMT systems utilize distributed, abstract word representations that can capture syntactic and semantic relationship between words, which could (partly) explain their relative success on the EN-HU language pair.

In relation to the speed of fuzzy match retrieval, which can be an issue when matches have to be retrieved for all source sentences in a TM, the results suggest that *SetSimilaritySearch* can be used as a fast proxy to *editdistance*. However, in this context it is important to strike the right balance between processing time and loss of training data by using different values for minimum similarity score and n-best candidates for *SetSimilaritySearch*. It still needs to be tested how well the NFR system works with other fuzzy matching metrics (Vanallemeersch and Vandeghinste, 2015), and how fast fuzzy matches can be retrieved from a TM with alternative methods, such as using the off-the-shelf search engine Apache Lucene (Gu et al., 2018; Zhang et al., 2018) or other approximate string matching methods (Koehn and Senellart, 2010b; Navarro, 2001).

7 Conclusion

The TM-NMT integration approach presented in this paper, Neural Fuzzy Repair, makes use of data augmentation to help improve machine translation quality using information retrieved from a translation memory. Compared to previous approaches to incorporate TM information into MT systems, NFR does not require different NMT architectures or algorithms, but relies solely on input preprocessing, and can thus be used in combination with any existing NMT system or toolkit. Tests on two language pairs (EN-NL and EN-HU) showed that this method can achieve substantial gains in estimated translation quality compared to a range of baseline systems, even for relatively small training set sizes. We believe that the ease of implementation of NFR could lead to the wider adoption of TM-NMT integration.

In a next step, we plan to compare the performance of NFR to other approaches to TM-NMT integration, for example by carrying out evaluations on the JRC-Acquis corpus (Gu et al., 2018; Koehn and Senellart, 2010a; Zhang et al., 2018). The approach also needs to be tested on data sets with a lower frequency of repeated sentences, other language pairs as well as different domains, ultimately also involving human evaluation (both in term of perceived quality and post-editing time). In addition, it would be informative to carry out a qualitative analysis of the NFR output in terms of how and to what extent the information contained in the fuzzy matches is used in the final translation, in comparison with the NMT baseline. We also intend to carry out further tests to potentially improve the quality of the output, for example by testing different match metrics and retrieval methods, NMT architectures (e.g. *transformer*), ways to include alignment information and by applying additional morphological preprocessing.

References

Roberto J. Bayardo, Yiming Ma, and Ramakrishnan Srikant. 2007. Scaling up all pairs similarity search. In *Proceedings of the 16th International Conference on World Wide Web*, pages 131–140.

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

Michael Bloodgood and Benjamin Strauss. 2015. Translation memory retrieval methods. *Computing Research Repository*, arXiv:1505.05841.

Bram Bulté, Tom Vanallemeersch, and Vincent Vandeghinste. 2018. M3TRA: integrating TM and MT for professional translators. In *Proceedings of the 21st Annual Conference of the European Association for Machine Translation*, pages 69–78.

Patrick Cadwell, Sharon O'Brien, and Carlos S. C. Teixeira. 2018. Resistance and accommodation: factors for the (non-) adoption of machine translation among professional translators. *Perspectives*, 26(3):301–321.

Qian Cao and Deyi Xiong. 2018. Encoding gated translation memory into neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3042–3047.

Raj Dabre, Fabien Cromieres, and Sadao Kurohashi. 2017. Enabling multi-source neural machine translation by concatenating source sentences in multiple languages. *Computing Research Repository*, arXiv:1702.06135.

Marcello Federico, Alessandro Cattelan, and Marco Trombetti. 2012. Measuring user productivity in machine translation enhanced Computer Assisted Translation. In *Proceedings of the 2012 Conference of the Association for Machine Translation in the Americas*, pages 44–56.

Yang Feng, Shiyue Zhang, Andi Zhang, Dong Wang, and Andrew Abel. 2017. Memory-augmented neural machine translation. *Computing Research Repository*, arXiv:1708.02005.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O. K. Li. 2018. Search engine guided neural machine translation. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pages 5133–5140.

Sanjika Hewavitharana, Stephan Vogel, and Alex Waibel. 2005. Augmenting a statistical translation system with a translation memory. In *Proceedings of the 10th Annual Conference of the European Association for Machine Translation*, pages 126–132.

Chris Hokamp. 2017. Ensembling factored neural machine translation models for automatic post-editing and quality estimation. *Computing Research Repository*, arXiv:1706.05083.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. *Computing Research Repository*, arXiv:1704.07138.

Heikki Hyyrö. 2001. Explaining and extending the bit-parallel approximate string matching algorithm of Myers. Technical report, Dept. of Computer and Information Sciences, University of Tampere.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. *Computing Research Repository*, arXiv:1701.02810.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*.

Philipp Koehn. 2009. *Statistical Machine Translation*. Cambridge University Press.

Philipp Koehn, Alexandra Birch, and Ralf Steinberger. 2009. 462 machine translation systems for Europe. *Proceedings of MT Summit XII*, pages 65–72.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 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*, pages 177–180.

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

Philipp Koehn and Jean Senellart. 2010b. Fast approximate string matching with suffix arrays and a* parsing. In *Proceedings of the ninth Conference of the Association for Machine Translation in the Americas*.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231.

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

Gonzalo Navarro. 2001. A guided tour to approximate string matching. *ACM computing surveys (CSUR)*, 33(1):31–88.

John Ortega, Felipe Sánchez-Martınez, and Mikel Forcada. 2016. Fuzzy-match repair using black-box machine translation systems: what can be expected? In *Proceedings of the 2016 Conference of the Association for Machine Translation in the Americas*, pages 27–39.

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

Caroline Rossi and Jean-Pierre Chevrot. 2019. Uses and perceptions of machine translation at the European Commission. *Journal of Specialised Translation*, 31:177–200.

Michel Simard and Pierre Isabelle. 2009. Phrase-based machine translation in a computer-assisted translation environment. In *Proceedings of MT Summit XII*, pages 120–127.

Michel Simard and Philippe Langlais. 2001. Sub-sentential exploitation of translation memories. In *Machine Translation Summit 8*, pages 335–339.

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 the 2006 Conference of the Association for Machine Translation in the Americas*, pages 223–231.

Ralf Steinberger, Andreas Eisele, Szymon Klocek, Spyridon Pilos, and Patrick Schlüter. 2013. DGT-TM: A freely available translation memory in 22 languages. *Computing Research Repository*, arXiv:1309.5226.

Tom Vanallemeersch and Vincent Vandeghinste. 2015. Assessing linguistically aware fuzzy matching in translation memories. In *Proceedings of the 18th Annual Conference of the European Association for Machine Translation*, pages 153–160.

Kun Wang, Chengqing Zong, and Keh-Yih Su. 2013. Integrating translation memory into phrase-based machine translation during decoding. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long papers)*, pages 11–21.

Jingyi Zhang, Masao Utiyama, Eiichro Sumita, Graham Neubig, and Satoshi Nakamura. 2018. Guiding neural machine translation with retrieved translation pieces. *Computing Research Repository*, arXiv:1804.02559.