Intelligent Translation Memory Matching and Retrieval with Sentence Encoders

Tharindu Ranasinghe[◇], **Constantin Orăsan**[♡] **and Ruslan Mitkov**[◇]
[◇]Research Group in Computational Linguistics, University of Wolverhampton, UK
[♡]Centre for Translation Studies, University of Surrey, UK
{t.d.ranasinghehettiarachchige, r.mitkov}@wlv.ac.uk
c.orasan@surrey.ac.uk

Abstract

Matching and retrieving previously translated segments from a Translation Memory is the key functionality in Translation Memories systems. However this matching and retrieving process is still limited to algorithms based on edit distance which we have identified as a major drawback in Translation Memories systems. In this paper we introduce sentence encoders to improve the matching and retrieving process in Translation Memories systems - an effective and efficient solution to replace edit distance based algorithms.

1 Introduction

Translation Memories (TMs) are "structured archives of past translations" which store pairs of corresponding text segments[1] in source and target languages known as "translation units" (Simard, 2020). TMs are used during the translation process in order to reuse previously translated segments. The original idea of TMs was proposed more than forty years ago when (Arthern, 1979) noticed that the translators working for the European Commission were wasting valuable time by re-translating (parts of) texts that had already been translated before. He proposed the creation of a computerised storage of source and target texts which could easily improve the performance of translators and that could be part of a computer-based terminology system. Based on this idea, many commercial TM systems appeared on the market in the early 1990s. Since then the use of this particular technology has kept growing and recent studies show that it is used on regular basis by a large proportion of translators (Zaretskaya et al., 2018).

Translation Memories systems help translators by continuously trying to provide them with so-called matches, which are translation proposals retrieved from its database. These matches are identified by comparing automatically the segment that has to be translated with all the segments stored in the database. There are three kinds of matches: exact, fuzzy and no matches. Exact matches are found if the segment to be translated is identical to one stored in the TM. Fuzzy matches are used in cases where it is possible to identify a segment which is similar enough to the one to be translated, and therefore, it is assumed that the translator will spend less time editing the translation retrieved from the database than translating the segment from scratch. No matches occur in cases where it is not possible to identify a fuzzy match (i.e. there is no segment similar enough to the one to be translated to be worth using its translation).

TMs distinguish between fuzzy matches and no matches by calculating the similarity between segments using a similarity measure and comparing it to a threshold. Most of the existing TM systems rely on a variant of the edit distance as the similarity measure and consider a fuzzy match when the edit distance score is between 70% and 95%.[2] The main justification for using

[1]Segments are typically sentences, but there are implementations which consider longer or shorter units.

[2]It is unclear the origin for these value, but they are widely used by translators. Most of the tools allow translators to customise the value of this threshold according to their needs. Translators use their experience to decide which value for the

this measure is the fact that edit distance can be easily calculated, is fast, and is largely language independent. However, edit distance is unable to capture correctly the similarity between segments when different wording and syntactic structures are used to express the same idea. As a result, even if the TM contains a semantically similar segment, the retrieval algorithm will not be able to identify it in most of the cases.

Researchers tried to address this shortcoming of the edit distance metric by employing similarity metrics that can identify semantically similar segments even when they are different at token level. Section 2 discusses some of the approaches proposed so far. Recent research on the topic of text similarity employed methods that rely on deep learning and various vector based representations used in this field (Ranasinghe et al., 2019b; Tai et al., 2015; Mueller and Thyagarajan, 2016). One of the reasons for this is that calculating the similarity between vectors is more straightforward than calculating the similarity between texts. It is easy to calculate how close or distant two vectors are by using well understood mathematical distance metrics. In addition, deep learning based methods proved more robust in numerous NLP applications.

In this paper we propose a novel TM matching and retrieval method based on the Universal Sentence Encoder (Cer et al., 2018) which has the capability to capture semantically similar segments in TMs better than methods based on edit distance. We selected the Universal Sentence Encoder as our sentence encoder since it outperforms other sentence encoders like Infersent (Conneau et al., 2017) in many Natural Language Processing tasks including Semantic Retrieval (Cer et al., 2018). Also the recently release of Multilingual Universal Sentence Encoder [3] is available on 16 different languages (Yang et al., 2019). Since we are planning to expand our research to other language pairs than the English - Spanish pair investigated in this paper, the multilingual aspect of the Universal Sentence Encoder can prove very useful.

The rest of the paper is organised as follows. Section 2 briefly describes several approaches used to improve the matching and retrieval in TMs. Section 3 contains information about the settings of the experiments carried out in this paper. It includes the experiments that were done for semantic textual similarity tasks comparing the Universal Sentence Encoder and edit distance. The same section also presents the results of the experiments on real world TMs. Section 4 discusses the results and describes future research directions. The implementation of the methods presented in this paper is available on Github.[4]

2 Related Work

Despite being the most used tools by professional translators, Translation Memories have rarely been criticised because of the quality of the segments they retrieve. Instead, quite often the requests from translators focus on the quality of the user interface, the need to handle different file formats, their speed and possibility of working in the cloud (Zaretskaya et al., 2018). Most of the current work on TMs is focused on the development of addons like terminology managers and plugins which integrate machine translation engines, as well as project management features (Gupta et al., 2016). Even though retrieval of previously translated segments is a key feature in a TM system, this process is still very much limited to edit-distance based measures.

Researchers working on natural language processing have proposed a number of methods which try to improve the existing matching and retrieval approaches used by translation memories. However, the majority of these approaches are not suitable for large TMs, like the ones normally employed by professional translators or were evaluated on very small number of segments. Planas and Furuse (1999) extend the edit distance metric to incorporate lemmas and part-of-speech information when calculating the similarity between two segments, but they test their approach on less than 150 segments from two domains using two translation memories with less than 40,000 segments in total. Lemmas and part-of-speech information is also used in (Hodász and Pohl, 2005) in order to improve matching, especially for morphologically rich languages like Hungarian. They also experiment with sentence skeletons in which NPs are automatically aligned between source and target. Unfortunately, the paper presents only preliminary results. Pekar

threshold is appropriate for a given text.

[3]https://tfhub.dev/google/universal-sentence-encoder-multilingual-large/3

[4]https://github.com/tharindudr/intelligent-translation-memories

and Mitkov (2007) show how it is possible to improve the quality of matching by taking into consideration the syntactic structure of sentences. Unfortunately, the evaluation is carried out on only a handful of carefully selected segments. Another method which performs matching at level of syntactic trees is proposed in (Vanallemeersch and Vandeghinste, 2014). The results presented in their paper are preliminary and the authors notice that tree matching method is "prohibitively slow".

More recent work has focused on incorporating paraphrases into the matching and retrieving algorithm (Utiyama et al., 2011; Gupta and Orasan, 2014; Chatzitheodorou, 2015). Utiyama et al. (2011) proposed a finite transducer which considers paraphrases during the matching. The evaluation shows that the method improves both precision and recall of matching, but it was carried out with only one translator and focused only on segments with exactly the same meaning. Gupta and Orasan (2014) proposed a variant of the edit distance metric which incorporates paraphrases from PPDB[5] using greedy approximation and dynamic programming. Both automatic evaluation and evaluation with translators show the advantages of using this approach (Gupta et al., 2016). Chatzitheodorou (2015) follows a similar approach. They use NooJ[6] to create paraphrases for the verb constructions in all source translation units to expand the fuzzy matching capabilities when searching in the TM. Evaluation with professional translators showed that the proposed method helps and speeds up the translation process.

To best of our knowledge, deep learning methods have not been used successfully in translation memories. Gupta (2016) presents an attempt to use ReVal, an evaluation metric that was successfully applied in the WMT15 metrics task (Gupta et al., 2015). Unfortunately, none of the neural based methods used are able to lead to better results than the standard edit distance.

3 Experiments and Results

As mentioned above, the purpose of this research is to find out whether it is possible to improve the quality of the retrieved segments by using the Universal Sentence Encoder (Cer et al., 2018) released by Google as the sentence encoder for this experiment. It comes with two versions: one trained with a Transformer encoder and the other trained with a Deep Averaging Network (DAN) (Cer et al., 2018). The transformer encoder architecture uses an attention mechanism (Vaswani et al., 2017) to compute context aware representations of words in a sentence and average those representations to calculate the embedding for the sentence. The DAN encoder begins by averaging together word and bi-gram level embeddings. Sentence embeddings are then obtained by passing the averaged representation through a feedforward deep neural network (DNN). The architecture of the DAN encoder is similar to the one proposed in (Iyyer et al., 2015).

The two architectures have a trade-off of accuracy and computational resource requirement. The one that relies on a Transformer encoder has higher accuracy, but is computationally more expensive. In contrast the one with DAN encoding is computationally less expensive, but has a slightly lower accuracy. For the experiments presented in this paper we used both architectures. The trained Universal Sentence Encoder model for English is available on TensorFlow Hub[7].

3.1 Experiments on STS

In order to assess the performance of the two architectures described in the previous section, we applied them on several Semantic Textual Similarity (STS) datasets and compared their results with those obtained when only edit distance is employed. This was done only to find out how well our unsupervised methods capture semantic textual similarity in comparison to a simple edit distance.

In this section we present the datasets that we used, the method and the results.

3.1.1 Dataset

We carried out these experiments using two datasets: the SICK dataset (Bentivogli et al., 2016) and SemEval 2017 Task 1 dataset (Cer et al., 2017) which we will refer to as STS2017 dataset.

The SICK data contains 9,927 sentence pairs with a 5,000/4,927 training/test split. Each pair is annotated with a relatedness score between 1 and 5, corresponding to the average relatedness judged by 10 different individuals. Table 1 shows a few examples from the SICK training dataset.

[5]http://paraphrase.org/
[6]https://nooj4nlp.net.cutestat.com/
[7]https://tfhub.dev/google/universal-sentence-encoder/4

Sentence Pair	Similarity
1. A little girl is looking at a woman in costume. 2. A young girl is looking at a woman in costume.	4.7
1. A person is performing tricks on a motorcycle. 2. The performer is tricking a person on a motorcycle.	2.6
1. Someone is pouring ingredients into a pot. 2. A man is removing vegetables from a pot.	2.8
1. Nobody is pouring ingredients into a pot. 2. Someone is pouring ingredients into a pot.	3.5

Table 1: Example sentence pairs from the SICK training data

The STS2017 test datset has 250 sentence pairs annotated with a relatedness score between [1,5]. As the training data for the competition, participants were encouraged to make use of all existing data sets from prior STS evaluations including all previously released trial, training and evaluation data [8]. Once we combined them all STS2017 had 8527 sentence pairs with a 8227/250 training/test split. Table 2 shows a few examples from the STS2017 dataset.

Sentence Pair	Similarity
1. Two people in snowsuits are lying in the snow and making snow angels. 2. Two angels are making snow on the lying children	2.5
1. A group of men play soccer on the beach. 2. A group of boys are playing soccer on the beach.	3.6
1. One woman is measuring another woman's ankle. 2. A woman measures another woman's ankle.	5.0
1. A man is cutting up a cucumber. 2. A man is slicing a cucumber.	4.2

Table 2: Example sentence pairs from the STS2017 data

3.1.2 Method

We followed a simple approach to calculate the similarity between two sentences. Each sentence was passed through the Universal Sentence Encoder to acquire the corresponding sentence vector for each sentence. The Universal Sentence Encoder uses a 512 dimension vector to represent a sentence. If the two vectors for two sentences X and Y are a and b correspondingly, we calculate the cosine similarity between a and b as of equation 1 and use that value to represent the similarity between the two sentences.

$$\begin{aligned}\cos(\mathbf{a},\mathbf{b}) &= \frac{\mathbf{ab}}{\|\mathbf{a}\|\|\mathbf{b}\|} \\ &= \frac{\sum_{i=1}^{n}\mathbf{a}_i\mathbf{b}_i}{\sqrt{\sum_{i=1}^{n}(\mathbf{a}_i)^2}\sqrt{\sum_{i=1}^{n}(\mathbf{b}_i)^2}}\end{aligned} \tag{1}$$

Simple edit distance between two sentences was used as a baseline. In order to convert it to a similarity metric, we converted the edit distance between two sentences to the negative value and performed a min-max normalisation over the whole dataset to bring it to a value between 0 and 1.

[8] http://alt.qcri.org/semeval2017/task1/

3.1.3 Results

All the results were evaluated using the three evaluation metrics normally employed in STS tasks: Pearson correlation (τ), Spearman correlation (ρ) and Mean Squared Error (MSE). Table 3 contains results for SICK dataset and Table 4 for STS2017 dataset.

Algorithm	τ	ρ	MSE
DAN Encoder	0.761	0.708	0.514
Transformer	0.780	0.721	0.426
Edit Distance	0.321	0.422	3.112

Table 3: Results for SICK dataset

Algorithm	τ	ρ	MSE
DAN Encoder	0.744	0.708	0.612
Transformer	0.723	0.721	0.451
Edit Distance	0.360	0.481	2.331

Table 4: Results for STS2017 dataset

As shown in Tables 3 and 4 both architectures of Universal Sentence Encoder outperform edit distance significantly in all three evaluation metrics for both datasets. This is not surprising given how simple edit distance is, but reinforces our motivation to use better methods to capture semantic similarity in translation memories. Table 5 shows some of the example sentences where Universal Sentence Encoder architectures showed promising results against the baseline - edit distance.

As can be seen in table 5 both architectures of Universal Sentence Encoder handle semantic textual similarity better than edit distance in many cases where the word order is changed in two sentences, but the meaning remains same. This detection of similarity even when the word order is changed will be important in segment matching and retrieval in TMs.

3.2 Experiments on Translation Memories

In this section we present the experiments we conducted on TMs using the Universal Sentence Encoder. First we introduce the dataset that

Sentence 1	Sentence 2	GOLD	ED	Transf.	DAN
Israel expands subsidies to settlements	Israel widens settlement subsidies	1.0000	0.0214	0.8524	0.8231
A man plays the guitar and sings.	A man is singing and playing a guitar.	1.0000	0.0124	0.7143	0.7006
A man with no shirt is holding a football	A football is being held by a man with no shirt	1.0000	0.0037	0.9002	0.8358
EU ministers were invited to the conference but canceled because the union is closing talks on agricultural reform, said Gerry Kiely, a EU agriculture representative in Washington.	Gerry Kiely, a EU agriculture representative in Washington, said EU ministers were invited but canceled because the union is closing talks on agricultural reform.	1.0000	0.1513	0.7589	0.7142

Table 5: Examples of sentence pairs where Universal Sentence Encoder performed significantly better than edit Distance in the STS task. GOLD column shows the score assigned by humans, normalised between 0 and 1. The ED column shows the similarity obtained regarding the edit distance. Transf and DAN columns show the similarity obtained by Transformer and DAN architecture in Universal Sentence Encoder respectively.

we used and then we present the methodology employed and the evaluation results.

3.2.1 Dataset

In order to conduct the experiments, we used DGT-Translation Memory, a translation memory made publicly available by The European Commission's (EC) Directorate General for Translation, together with the EC's Joint Research Centre. It consists of segments and their professionally produced translations covering twenty-two official European Union (EU) languages and their 23 language-pair combinations (Steinberger et al., 2012). It is typically used by researches who work on TMs (Gupta et al., 2016; Baisa et al., 2015).

We used the English - Spanish segment pairs for the experiments, but our approach is easily adoptable to any language pair as long as there are embeddings available for the source language. We used data from the year 2018: *2018 Volume 1* was used as the translation memory and *2018 Volume 3* was used as the input segments. The translation memory we built from *2018 volume 1* had 230,000 segment pairs, whilst the *2018 volume 3* had 66,500 segment pairs which we used as input segments.

3.2.2 Method

We conducted the following steps for both architectures in Universal Sentence Encoder.

1. Calculated the sentence embeddings for each segment in the translation memory (230,000 segments) and stored the vectors in a AquilaDB[9] database. AquilaDB is a Decentralized vector database to store Feature Vectors and perform K Nearest Neighbour retrieval. It is build on top of popular Apache CouchDB[10]. A record of the database has 3 fields: source segment, target segment and source segment vector.

2. Calculated the sentence embedding for one incoming segment.

3. Calculated the cosine similarity of that embedding with each of the embedding in the database using equation 1. We retrieve the embedding that had the highest cosine similarity with the input segment embedding and retrieve the corresponding target segment for the embedding as the translation memory match. We used *'getNearest'* functionality provided by AquilaDB for this step.

The efficiency of the TM matching and retrieval is a key-factor for translators who are using them. Therefore, we first analysed the efficiency of each architecture in Universal Sentence Encoder. The results are shown in table 6. The experiments were carried out on an Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz desktop computer. The performance of the Universal Sentence Encoder will be more

[9]`https://github.com/a-mma/AquilaDB`
[10]`https://github.com/apache/couchdb`

efficient in a GPU (Graphics Processing Unit). Nonetheless we carried our experiments without using a GPU since the translators using translation memory tools would probably not have access to a GPU on daily basis.

Architecture	Step 1	Step 2	Step 3
DAN Encoder	78s	0.77s	0.40s
Transformer	108s	1.23s	0.40s

Table 6: Time efficiency of each architecture in Universal Sentence Encoder

When we calculated the sentence embeddings for the segments in the translation memory in Step 1, we processed the segments in batches of 256 segments. As can be seen in the table 6, DAN Architecture had the maximum efficiency providing sentence embeddings within 78 seconds for 230,000 segments. The Transformer architecture was not too far behind, being able to calculate the embeddings of the 230,000 segments in 108 seconds.

The next column in table 6 reports the time taken from each sentence encoder to embed a single segment. We did not consider input segments as batches as we did earlier for the segments in the translation memory. We assumed that since the translators translate the segments one by one it would not be fair to encode the input segments in batches. In that step too, the DAN Architecture was more efficient than the Transformer Architecture.

The next column is the time taken to retrieve the best match from the translation memory. It includes the time taken to calculate the cosine similarity of the segment embeddings of the segments of the translation memory with the segment embedding of the input segment. Also, it includes the time taken to sort the similarities and get the index of the highest similarity and retrieve the corresponding segment which we considered as the best match for the input segment from the translation memory. As shown in the table 6 both architectures took approximately similar time for this step since the size of the embedding is same for both architectures.

As a whole, time taken to acquire the best match from the translation memory is the combined time taken to step 2 and step 3. Therefore, the time taken by the Transformer Encoder to retrieve a match from the translation memory for one incoming sentence is just 1.6s, which is reasonable. In light of this, we decided to use the Transformer Architecture for future experiments since it is efficient enough and since it was reported that it provides better accuracy in semantic retrieval tasks than the DAN Architecture (Cer et al., 2018).

3.2.3 Results

In order to compare the results obtained by our method with those of an existing translation memory tool we used Okapi which uses simple edit distance to retrieve matches from the translation memory. We calculated the METEOR score (Denkowski and Lavie, 2014) between the actual translation of the incoming segment and the match we retrieved from the translation memory with the transformer architecture of the Universal Sentence Encoder. We repeated the same process with the match we retrieved from Okapi. We used METEOR score since we believed it can capture the semantic similarity between two segments better than the BLEU score (Denkowski and Lavie, 2014).

To understand the performance of our method, we first removed the segments where the match provided by Okapi and the Universal Sentence Encoder was same. Then, to have a better analysis of the results, we divided the results in to 5 partitions. The first partition contained the matches derived from Okapi that had a fuzzy match score between 0.8 and 1. We calculated the average METEOR score for the segments retrieved from Okapi and for the segments retrieved from Universal Sentence Encoder in the particular partition. We performed the same process for all the partitions: fuzzy match score ranges 0.6-0.8, 0.4-0.6, 0.2-0.4 and 0-0.2.

As shown in table 7 Universal Sentence Encoder performs better than Okapi for the fuzzy match scores below 0.8, which means that the Universal Sentence Encoder performs better when Okapi fails to find a significantly similar match in TM. However, this is not a surprise given that METEOR score is largely based on overlapping ngrams, and therefore will reward segments that have a high fuzzy match score.

However, we noticed that in most cases, the difference between the actual translation and the suggested match from either Okapi or Universal Sentence Encoder is just a number, a location, an organisation or a name of a person. We

Fuzzy score	Okapi	USE	Amount
0.8-1.0	**0.931**	0.854	1624
0.6-0.8	0.693	**0.702**	4521
0.4-0.6	0.488	**0.594**	6712
0.2-0.4	0.225	**0.318**	13136
0-0.2	0.011	**0.134**	24612

Table 7: Result comparison between Okapi and the Universal Sentence Encoder for each partition. Fuzzy score column represents the each partition. Okapi column shows the average METEOR score between the matches provided by the Okapi and the actual translations in that partition. USE column shows the average METEOR score between the matches provided by the Universal Sentence Encoder and the actual translations in that partition. Amount column shows the number of sentences in each partition. Bold shows the best result for that partition

Fuzzy score	Okapi	USE	Amount
0.8-1.0	**0.942**	0.889	1512
0.6-0.8	0.705	**0.726**	3864
0.4-0.6	0.496	**0.602**	6538
0.2-0.4	0.228	**0.320**	13128
0-0.2	0.011	**0.134**	24612

Table 8: Result comparison between Okapi and the Universal Sentence Encoder for each partition after performing NER. The Fuzzy score column represents each partition. The Okapi column shows the average METEOR score between the matches provided by the Okapi and the actual translations in that partition. The USE column shows the average METEOR score between the matches provided by the Universal Sentence Encoder and the actual translations in that partition. The Amount column shows the number of sentences in each partition. Bold shows the best result for that partition

thought this might affect the results since we are depending on the Universal Sentence Encoder's ability to retrieve semantically similar segments from the TM. For this reason, we applied a Named Entity Recognition (NER) pipeline on the actual translations, segments retrieved from Okapi and the segments retrieved from Universal Sentence Encoder. Since the target language is Spanish, we used the Spanish NER pipeline provided by Spacy that was trained on the AnCora and WikiNER corpus[11]. We detected locations, organisations and person names with the NER pipeline and replaced them with a placeholder. We also used Añotador [12] to detect dates in the segments and replaced them too with a placeholder. Last, we used a regular expression to detect number sequences in the segments and replaced them too with a place holder. After that we removed the cases where the match provided by Okapi and the Universal Sentence Encoder is same and recalculated the results in table 7 following the same process.

As shown in table 8 for the cases where the fuzzy match score is above 0.8, the segments retrieved by Okapi are still better than the segments retrieved from the Universal Sentence Encoder. However for the cases where the fuzzy match score is below 0.8 the Universal Sentence Encoder seems to be better than Okapi. After performing NER, the results of the Universal Sentence Encoder improved significantly in most of the partitions: specially in 0.6-0.8 partition.

Given the fact that METEOR relies largely on string overlap we assumed that it is unable to capture the fact that the segments retrieved using the Universal Sentence Encoder are semantically equivalent. Therefore, we asked three native Spanish speakers to compare the segments from Okapi and report the sentences where Universal Encoder performed significantly better than Okapi. Due to the time restrictions they did not have time to go through all the segments. But their opinion was generally that the Universal Sentence Encoder was better at identifying semantically similar segments in the TM. Table 9 presents sample segments they provided.

4 Conclusion and Future Work

In this paper we have proposed a new TM matching and retrieval method based on the Universal Sentence Encoder. Our assumption was that by using this representation we will be able to retrieve better segments from a TM than when using a standard edit distance. As shown in 3.2.3 section, the Universal Sentence Encoder performs better than Okapi for fuzzy match scores ranged below 0.8. Therefore, we believe that the sentence encoders can improve the matching and retrieval in TMs and should be explored more. Usually TM matches with lower fuzzy match scores (¡ 0.8) are not used by professional translators, or when used, they lead to a decrease in translation productivity. But our method can provide better matches to sentences below fuzzy match score 0.8, hence will be able to improve the translation productivity. According to the annotation guidelines of (Cer et al., 2017) a semantic textual similarity score of 0.8 means *”The two sentences are mostly*

[11]`https://spacy.io/models/es`
[12]`http://annotador.oeg-upm.net/`

Source segment	Human Translated segment	Universal Sentence Encoder Suggestion	Okapi Suggestion
If applicable	En su caso	si procede	No procede
Date of granting	Fecha de concesión de la subvención	Fecha de autorización	Fecha de la garantía otorgada
This Decision shall be kept under constant review and shall be renewed or amended, as appropriate, if the Council deems that its objectives have not been met.'	La presente Decisión estará sujeta a revisión continua y se prorrogará o modificará, según proceda, si el Consejo estima que no se han cumplido sus objetivos.	Será prorrogada o modificada, según proceda, si el Consejo considera que no se han cumplido sus objetivos.	Se prorrogará o modificará, si procede, en caso de que el Consejo estime que no se han cumplido los objetivos de la misma.
The information shall include:	Esta información incluirá:	Esa información podrá versar sobre lo siguiente:	Los indicadores clave de rendimiento incluirán:
General characteristics of the finished product	Características generales del producto terminado	descripción del producto final,	Características generales del componente de servicios de copernicus
Such reports shall be made publicly available.	Dichos informes se harán públicos.	Sus informes se harán públicos.	Se pondrá a disposición del público un resumen de las evaluaciones.
The Commission decision to initiate the procedure ('the Opening Decision') was published in the Official Journal of the European Union.	La Decisión de la Comisión de incoar el procedimiento (en lo sucesivo, Decisión de incoación) se publicó en el Diario Oficial de la Unión Europea.	La Decisión de la Comisión de incoar el procedimiento (en lo sucesivo, Decisión de incoación) fue publicada en el Diario Oficial de la Unión Europea.	La decisión de la Comisión de incoar el procedimiento se publicó en el Diario Oficial de la Unión Europea.
Chapter 2 is amended as follows:	El capítulo 2 se modifica como sigue:	la parte 2 se modifica como sigue:	la sección 2 queda modificada como sigue:

Table 9: Example segments where Universal Sentence Encoder suggestion was better than the Okapi suggestion

equivalent, but some unimportant details differ" and semantic textual similarity score of 0.6 means *"The two sentences are roughly equivalent, but some important information differs/missing"*. If we further analyse the fuzzy match score range 0.6-0.8, as shown in table 10, the mean semantic textual similarity for the sentences provided by Universal Sentence Encoder is 0.768. Therefore, we assume that the matches retrieved from the Universal Sentence Encoder in the fuzzy match score range 0.6-0.8 will help to improve the translation productivity. However, this is something that we plan to analyse further by carrying out evaluations with professional translators.

In the future, we also plan to experiment with other sentence encoders such as Infersent (Conneau et al., 2017) and SBERT (Reimers and Gurevych, 2019) and with alternative algorithms which are capable to capture semantic textual similarity between two sentences. We will try unsupervised methods like word vector averaging

Fuzzy score	Mean STS score
0.8 - 1.0	0.952
0.6 - 0.8	0.768
0.4 - 0.6	0.642
0.2 - 0.4	0.315
0 - 0.2	0.121

Table 10: Mean STS score for the sentences retrieved by Universal Sentence Encoder for each fuzzy match score. Fuzzy score column shows the fuzzy match score ranges and Mean STS score column shows that mean STS score for the sentence retrieved by Universal Sentence Encoder for that fuzzy match score range.

and word moving distance (Ranasinghe et al., 2019a) as well as supervised algorithms such Siamese neural networks (Ranasinghe et al., 2019b) and transformers (Devlin et al., 2018).

5 Acknowledgment

We would like to acknowledge Rocío Caro Quintana from University of Wolverhampton, Encarnación Núñez Ignacio from University of Wolverhampton and Bellés-Calvera, Lucía from Jaume I University: the team of volunteer annotators that provided their free time and efforts to manually evaluate the results between Universal Sentence Encoder and edit Distance.

Also we would like to acknowledge María Navas-Loro and Pablo Calleja from Polytechnic University of Madrid for providing Añotador for free to detect dates in the Spanish segments.

References

Arthern, Peter J. 1979. Machine translation and computerized terminology systems: A translator's viewpoint. *Translating and the Computer, Proceedings of a Seminar, London 14th November 1978. Amsterdam: North-Holland Publishing Company*, pages 77–108.

Baisa, Vít, Aleš Horák, and Marek Medveď. 2015. Increasing coverage of translation memories with linguistically motivated segment combination methods. In *Proceedings of the Workshop Natural Language Processing for Translation Memories*, pages 31–35, Hissar, Bulgaria, September. Association for Computational Linguistics.

Bentivogli, Luisa, Raffaella Bernardi, Marco Marelli, Stefano Menini, Marco Baroni, and Roberto Zamparelli. 2016. Sick through the semeval glasses. lesson learned from the evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. *Language Resources and Evaluation*, 50:95–124.

Cer, Daniel M., Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *SemEval@ACL*.

Cer, Daniel, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium, November. Association for Computational Linguistics.

Chatzitheodorou, Konstantinos. 2015. Improving translation memory fuzzy matching by paraphrasing. In *Proceedings of the Workshop Natural Language Processing for Translation Memories*, pages 24–30, Hissar, Bulgaria, September. Association for Computational Linguistics.

Conneau, Alexis, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark, September. Association for Computational Linguistics.

Denkowski, Michael and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Gupta, Rohit and Constantin Orasan. 2014. Incorporating paraphrasing in translation memory matching and retrieval. In *Proceedings of the Seventeenth Annual Conference of the European Association for Machine Translation (EAMT2014)*, pages 3–10.

Gupta, Rohit, Constantin Orăsan, and Josef van Genabith. 2015. ReVal: A Simple and Effective Machine Translation Evaluation Metric Based on Recurrent Neural Networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1066–1072, Lisbon, Portugal, September.

Gupta, Rohit, Constantin Orăsan, Marcos Zampieri, Mihaela Vela, Josef van Genabith, and Ruslan Mitkov. 2016. Improving translation memory matching and retrieval using paraphrases. *Machine Translation*, 30(1-2):19–40.

Gupta, Rohit. 2016. *USE OF LANGUAGE TECHNOLOGY TO IMPROVE MATCHING AND*

RETRIEVAL IN TRANSLATION MEMORY. Ph.D. thesis, University of Wolverhampton.

Hodász, Gábor and Gábor Pohl. 2005. MetaMorpho TM: a linguistically enriched translation memory. In *In International Workshop, Modern Approaches in Translation Technologies*.

Iyyer, Mohit, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1681–1691, Beijing, China, July. Association for Computational Linguistics.

Mueller, Jonas and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 2786–2792. AAAI Press.

Pekar, Viktor and Ruslan Mitkov. 2007. New Generation Translation Memory: Content-Sensitive Matching. In *Proceedings of the 40th Anniversary Congress of the Swiss Association of Translators, Terminologists and Interpreters*.

Planas, Emmanuel and Osamu Furuse. 1999. Formalizing Translation Memories. In *Proceedings of the 7th Machine Translation Summit*, pages 331–339.

Ranasinghe, Tharindu, Constantin Orasan, and Ruslan Mitkov. 2019a. Enhancing unsupervised sentence similarity methods with deep contextualised word representations. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 994–1003, Varna, Bulgaria, September. INCOMA Ltd.

Ranasinghe, Tharindu, Constantin Orasan, and Ruslan Mitkov. 2019b. Semantic textual similarity with Siamese neural networks. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 1004–1011, Varna, Bulgaria, September. INCOMA Ltd.

Reimers, Nils and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11.

Simard, Michel. 2020. Building and using parallel text for translation. In O'Hagan, Minako, editor, *The Routledge Handbook of Translation and Technology*, chapter 5, pages 78 —- 90. Routledge.

Steinberger, Ralf, Andreas Eisele, Szymon Klocek, Spyridon Pilos, and Patrick Schlüter. 2012. DGT-TM: A freely available translation memory in 22 languages. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 454–459, Istanbul, Turkey, May. European Language Resources Association (ELRA).

Tai, Kai Sheng, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566, Beijing, China, July. Association for Computational Linguistics.

Utiyama, Masao, Graham Neubig, Takashi Onishi, and Eiichiro Sumita. 2011. Searching Translation Memories for Paraphrases. In *Proceedings of the 13th Machine Translation Summit*, pages 325–331, Xiamen, China, September.

Vanallemeersch, Tom and Vincent Vandeghinste. 2014. Improving fuzzy matching through syntactic knowledge. In *Translating and the Computer 36*, volume 36, pages 90 – 99, London, UK.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Yang, Yinfei, Daniel Matthew Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernández Ábrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2019. Multilingual universal sentence encoder for semantic retrieval. *ArXiv*, abs/1907.04307.

Zaretskaya, Anna, Gloria Corpas Pastor, and Miriam Seghiri. 2018. User Perspective on Translation Tools: Findings of a User Survey. In Corpas Pastor, Gloria and Isabel Duran, editors, *Trends in E-tools and Resources for Translators and Interpreters*, pages 37 – 56. Brill.