2nd Workshop on Natural Language Processing for Translation Memories (NLP4TM 2016)

Workshop Programme

9:00 – 10:30: Session 1: Invited talks

9:00 – 9:30
Marcello Federico, *Machine translation adaptation from translation memories in ModernMT*

9:30 – 10:00
Núria Bel, *Data fever in the 21st century. Where to mine Language Resources*

10:00 – 10:30
Samuel Läubli, *Data, not Systems: a Better Way to Conduct the Business of Translation*

10:30 – 11:00 Coffee break

11:00 – 12:00: Session 2: Research papers

11:00 – 11:20
A. Bellandi, G. Benotto, G. Di Segni, E. Giovannetti, *Investigating the Application and Evaluation of Distributional Semantics in the Translation of Humanistic Texts: a Case Study.*

11:20 – 11:40
Tapas Nayak, Santanu Pal, Sudip Kumar Naskar, Sivaji Bandyopadhyay, Josef van Genabith, *Beyond Translation Memories: Generating Translation Suggestions based on Parsing and POS Tagging*

11:40 – 12:00
Friedel Wolff, Laurette Pretorius, Loïc Dugast, Paul Buitelaar, *Methodological pitfalls in automated translation memory evaluation*

12:00 – 13:30: Session 3: Cleaning of translation memories shared task

12:00 – 12:20
Eduard Barbu, Carla Parra Escartín, Luisa Bentivogli, Matteo Negri, Marco Turchi, Marcello Federico, Luca Mastrostefano, Constantin Orasan, *1st Shared Task on Automatic Translation Memory Cleaning Preparation and Lessons Learned*

12:20 – 13:00
Presentations of the systems that took part in the shared task

13:00 – 13:30
Round table

Editors

Constantin Orasan	University of Wolverhampton, UK
Carla Parra	Hermes, Spain
Eduard Barbu	Translated, Italy
Marcello Federico	FBK, Italy

Workshop Organizers/Organizing Committee

Constantin Orasan	University of Wolverhampton, UK
Carla Parra	Hermes, Spain
Eduard Barbu	Translated, Italy
Marcello Federico	FBK, Italy

Workshop Programme Committee

Juanjo Arevalillo	Hermes, Spain
Yves Champollion	WordFast, France
Gloria Corpas	University of Malaga, Spain
Maud Ehrmann	EPFL, Switzerland
Kevin Flanagan	Swansea University, UK
Corina Forascu	University “Al. I. Cuza”, Romania
Gabriela Gonzalez	eTrad, Argentina
Rohit Gupta	University of Wolverhampton, UK
Manuel Herranz	Pangeanic, Spain
Samuel Läubli	Autodesk, Switzerland
Liangyou Li	DCU, Ireland
Qun Liu	DCU, Ireland
Ruslan Mitkov	University of Wolverhampton, UK
Aleksandros Poulis	Lionbridge, Sweden
Gabor Proszeky	Morphologic, Hungary
Uwe Reinke	Flensburg University of Applied Sciences, Germany
Michel Simard	NRC, Canada
Mark Shuttleworth	UCL, UK
Masao Utiyama	NICT, Japan
Mihaela Vela	Saarland University, Germany
Andy Way	DCU, Ireland
Joern Wuebker	Lilt, United States
Marcos Zampieri	Saarland University and DFKI, Germany

Table of contents

Author Index

Preface

Translation Memories (TM) are amongst the most used tools by professional translators, if not the most used. The underlying idea of TMs is that a translator should benefit as much as possible from previous translations by being able to retrieve how a similar sentence was translated before. Moreover, the usage of TMs aims at guaranteeing that new translations follow the client's specified style and terminology. Despite the fact that the core idea of these systems relies on comparing segments (typically of sentence length) from the document to be translated with segments from previous translations, most of the existing TM systems hardly use any language processing for this. Instead of addressing this issue, most of the work on translation memories focused on improving the user experience by allowing processing of a variety of document formats, intuitive user interfaces, etc.

The term second generation translation memories has been around for more than ten years and it promises translation memory software that integrates linguistic processing in order to improve the translation process. This linguistic processing can involve matching of subsentential chunks, edit distance operations between syntactic trees, incorporation of semantic and discourse information in the matching process. Terminologies, glossaries and ontologies are also very useful for translation memories, by facilitating the task of the translator and ensuring a consistent translation. The field of Natural Language Processing (NLP) has proposed numerous methods for terminology extraction and ontology extraction. The building of translation memories from corpora is another field where methods from NLP can contribute to improving the translation process.

We are happy we could include in the workshop programme four contributions dealing with the aforementioned issues. In addition, the programme of the workshop in complemented by the presentations of three well-known researchers.

The first edition of this workshop organised at RANLP 2015 confirmed the fact that there is interest in the research community for the topics proposed. In addition, it highlighted the need for automatic methods for cleaning translation memories. For this reason, the second edition of the NLP4TM workshop also organises a shared task on cleaning translation memories in an attempt to make the creation of resources for translation memories easier.

The Organising Committee would like to thank the Programme Committee, who responded with very fast but also substantial reviews for the workshop programme. This workshop would not have been possible without the support received from the EXPERT project (FP7/2007-2013 under REA grant agreement no. 317471, http://expert-itn.eu).

1st Shared Task on Automatic Translation Memory Cleaning Preparation and Lessons Learned

Eduard Barbu[1], Carla Parra Escartín[2], Luisa Bentivogli[3],
Matteo Negri[3], Marco Turchi[3], Marcello Federico[3],
Luca Mastrostefano[1], Constantin Orasan[4]

[1]Translated, Italy; [2]Hermes Traducciones, Spain; [3]FBK Trento, Italy;
[4]University of Wolverhampton, United Kingdom

{eduard,luca}@translated.net, carla.parra@hermestrans.com,
{bentivo,negri,turchi,federico}@fbk.eu, C.Orasan@wlv.ac.uk

Abstract

This paper summarizes the work done to prepare the first shared task on automatic translation memory cleaning. This shared task aims at finding automatic ways of cleaning TMs that, for some reason, have not been properly curated and include wrong translations. Participants in this task are required to take pairs of source and target segments from TMs and decide whether they are right translations. For this first task three language pairs have been prepared: English → Spanish, English → Italian, and English → German. In this paper, we report on how the shared task was prepared and explain the process of data selection and data annotation, the building of the training and test sets and the implemented baselines for automatic classifiers comparison.

Keywords: Translation Memories, data selection, data annotation

1. Introduction

Translation Memories (TMs) are among the most used tools by professional translators, if not the most used. The underlying idea of TMs is that a translator should benefit as much as possible from previous translations by being able to retrieve how a similar sentence was translated before. Moreover, the usage of TMs aims at guaranteeing that new translations follow the client's specified style and terminology. However, in order to ensure that professional translators can benefit from the contents already stored in a TM, this must be properly maintained and clean.

The first edition of the Natural Language Processing for Translation Memories (NLP4TM 2015) workshop organized at RANLP 2015 (Orasan and Gupta, 2015) highlighted the need for automatic methods for cleaning TMs. For this reason, in the second edition of the NLP4TM workshop (NLP4TM 2016)[1] a shared task on cleaning translation memories has been organized in an attempt to make the creation of resources for TMs easier as well as to enhance TM curation. This paper summarizes how the data for the shared task has been created and how the shared task has been organized.

The remainder of this paper is organized as follows: Section 2. summarizes the shared task. Section 3. shows how we have selected the data (Subsection 3.1.) to be annotated for three language pairs English-Italian, English-Spanish and English-German. The Subsections 3.2. and 3.3. discuss the annotation of the data and the inter-annotator agreement respectively. Section 4. shows how we have made the training and test sets, Section 5. reports on the baselines we have established to measure the participants' system submissions. The final section 6. summarizes our preparatory work for the shared task.

[1] http://rgcl.wlv.ac.uk/nlp4tm2016/

2. Shared Task

The NLP4TM 2016 shared task on cleaning translation memories aims at finding automatic ways of cleaning TMs that for some reason have not been properly curated and include wrong translations. Participants in this task are required to take pairs of source and target segments from TMs and decide whether they are right translations. For this first task three language pairs have been prepared: English → Spanish, English → Italian, and English → German.

The data was annotated with information on whether the source and target content of each TM segment represent a valid translation. In particular, the following 3 point scale has been applied:

1. The translation is correct (tag "1").
2. The translation is correct, but there are a few orthotypographic mistakes and therefore some minor post-editing is required (tag "2").
3. The translation is not correct (content missing/added, wrong meaning, etc.) (tag "3").

For each language pair, two thirds of the annotated segments are provided for training and one third is provided for testing during the evaluation phase.

Besides choosing the pair of languages with which they want to work, participants can choose to participate in either one or all of the following three tasks:

1. **Binary Classification (I)**: In this task, it is only required to determine whether a segment is right or wrong. For the first binary classification option, only tag ("1") is considered correct because the translators do not need to make any modification, whilst tags ("2") and ("3") are considered wrong translations.

2. **Binary Classification (II)**: As in the first task, in this task it is only required to determine whether the segment is right or wrong. However, in contrast to the first task, a segment is considered correct if it was labeled by annotators as ("1") or ("2"). Segments labeled ("3") are considered wrong because they require major post-editing.
3. **Fine-grained Classification**: In this task, the participating teams have to classify the segments according to the annotation provided in the training data: correct translations ("1"), correct translations with a few orthotypographic errors ("2"), and wrong ("3").

Participants were required to register their intention to participate by filling in an online form. Upon registration, we provided the registered participants with the training set. The test set will be distributed during the evaluation phase and the participating teams will be asked to submit the output of their systems in a format similar to the training set[2]. For evaluation, the standard measures precision, recall and the F1 will be used. In addition, we have foreseen a potential manual error analysis of subsets of the test data. The extent of this analysis will depend on the number of systems submitted. The numbers of runs submitted by participants has not been limited, although the participating teams are required to indicate their primary (and secondary, if relevant) runs.
In order to ensure the reusability and replicability of the shared task results and with the aim of making a real impact in professional translation workflows, all participants have been encouraged to release their systems and make them publicly available for future use. Besides, the development of methods that can be run on large datasets without requiring a lot of computational resources is also fostered. Thus, participans have also been encouraged not to use machine translation as one of the factors used to determine the class of a segment.

3. Data preparation

3.1. Data selection

The data was sampled from the public part of MyMemory (Trombetti, 2009) the biggest translation memory database in the world. The public part of MyMemory is composed of all bi-segments that the translators agreed to make public, from public parallel corpora and glossaries, data crawled from parallel sites on the web and the individual contributions through a collaborative web interface.
Regarding the percentage of errors, the bi-segments coming from the translators have fewer errors, the bi-segments coming from the collaborative web interface have most errors and the bi-segments coming from public parallel corpora or from crawling the web are somewhere in the middle.
In the initial phase we extracted approximately 30K bi-segments for each language pair taking care to sample from all the above mentioned sources. The bi-segments are heterogeneous and belong to different domains ranging from medicine and physics to colloquial conversations. Once we had this first pre-selection, we filtered the extracted bi-segments according to the following criteria:

1. **Minimum length**. The source and target segments should contain at least three words. MyMemory contains a significant number of entries that have only a word or two. However in many cases it is hard to understand if the source is a translation of the target because the context for interpreting the source and target is missing. We decided to avoid this situation for the task and therefore all segments shorter than a 3-word-span were deleted.

2. **No tags**. The extracted bi-segments should not contain tags or strange characters. Even if in the translation memory cleaning task one should consider segments that contain tags or strange characters, their identification is trivial and therefore was excluded from the task.

3. **Appropriate language codes**. The language codes of the source and target segments should coincide with the declared language codes. For example, if the source segment language code is declared as English and the target language code segment is declared as Spanish then the source segment language code should be English and the target segment language code should be Spanish. To check that this is indeed the case we used the high quality automatic language detector Cybozu [3].

4. **One to Many/Many to One**. We only accepted those bi-segments where one source sentence corresponds to at least one target sentence or one target sentence corresponds to at least one source sentence. That is: all bi-segments where many sentences in the source segment corresponded to many sentences in the target sentence were rejected because these bi-segments need realignment.

5. **Uniqueness**. The source and target segments should be unique across the set. We allowed the possibility of having a repeated source segment with multiple corresponding target segments as long as the target segments differed from each other, and viceversa: a unique target segment with diferring source segments[4].

From the bi-segments that met the above criteria we sampled again 10K bi-segments per language pair from which we then manually selected approximately 3K bi-segments per language pair. To facilitate the manual selection of the negative examples, we computed the cosine similarity score between the Machine Translation of the English segment and the target bi-segment. The hypothesis to consider was that low cosine similarity scores can signal bad translations.

[2] Due to time constraints, the testing phase will take place in the last weeks prior to the NLP4TM 2016 workshop and therefore no results can be reported at this time.

[3] https://github.com/shuyo/language-detection

[4] Two segments are different if the segments as character string are different after space normalization.

The manually selected bi-segments do not contain inappropriate language or other errors that cannot be identified automatically.

3.2. Data annotation

The set containing approximately 3K bi-segments per language pair was annotated by two native speakers of each target language. The guidelines for annotating this data set contain annotation instructions and examples[5]. In what follows, we present the annotation guidelines for English–Spanish. Similar annotation guidelines have been produced for the English–German and English–Italian language pairs.

1. You should give the score "1" if the translations can be accepted without editing. That is, the segment in Spanish preserves the meaning of the English segment.

   Example: "This product contains mineral oil."→"Este producto contiene aceite mineral." is a good Spanish translation of the English original segment. You do not need to change anything: punctuation or words.

2. You should give the score "2" when the few operations of editing you perform do not affect the meaning of the phrase. For example you should annotate "2" when:

   - The Spanish segment preserves the meaning of the English segment. However the Spanish segment has very few extra stuff that once deleted makes the translation acceptable:

     Example: "This product contains mineral oil."→"d Este producto contiene aceite mineral.". Deleting the "d" at the beginning makes the translation acceptable (tag "1")

   - The Spanish segment has (or lacks) punctuation that however do not impede understanding the segment. Adding or deleting the extra-punctuation renders the translation acceptable (tag "1"):

     Example: "This product contains mineral oil."→"Este producto contiene aceite mineral". Adding the final dot renders the translation (tag "1").

   - The Spanish segment has very few typos relative to the length of translation. Correcting the typos makes the translation acceptable (tag "1").

     Example: "This product contains mineral oil."→"Este produto contiene aceite mineral.". Correcting produto→producto makes the translation acceptable (tag "1").

3. You should give the score "3" if you need to perform substantial editing or editing that changes the meaning of the Spanish segment.

[5]The reader can consult these annotation guidelines at the web address: http://rgcl.wlv.ac.uk/nlp4tm2016/shared-task/.

Annotator	Annotator 2			
	Category	1	2	3
Annotator 1	1	**1127**	276	281
	2	209	**382**	305
	3	10	9	**360**

Table 1: The agreement for English–Italian

   - *Example*: "This product contains mineral oil."→"Este producto contiene agua mineral.". You need to replace a whole content word that is "agua" (water) with a new word "aceite" (oil) and thus the meaning of the sentence changes.

   - *Example*: "This product contains mineral oil."→"Este produto contiene aceite mineral". In this case, you need to change produto→producto, aciete→aceite and add the final dot to render an acceptable translation. Even if the editing operations do not change the meaning of the Spanish segment the numbers of edits you need to perform is substantial relative to the length of the segment.

The annotation has been performed with the aid of the MT-Equal (Girardi et al., 2014), a toolkit for Human Assessment of Machine Translation Output, developed and maintained by FBK. MT-Equal is an online tool accessible through the Chrome web browser[6]. It defines two types of users: administrators and annotators. While the annotators perform the annotation, the administrators can load data, assign tasks to the annotators, follow the task progress, export the results etc.
Our initial idea was that after the two annotators annotate the 3K they will agree on more than 2K bi-segments. The identical annotated bi-segments would then be used to build the training and test sets. In the next section, we discuss the inter-annotator agreement for each language-pair.

3.3. Inter-annotator agreement

We computed the inter-annotator agreement using the well known Cohen's kappa coefficient (Cohen, 1960). In Table 3.3., we present the agreement for the English–Italian language pair. The main diagonal of the table shows the number of bi-segments where the annotators[7] agree. They agreed for 1869 bi-segments. The number fell short of the 2K bi-segments we were expecting. To reach at least that number we asked an arbiter to annotate the 281 bi-segments that were annotated with tag 1 by annotator 1, and with tag 3 by annotator 2. The arbiter annotated 182 instances with 1, 32 instances as 2 and 67 instances as 3. The final set to be used for training and testing for English–Italian consists of the sum of all agreements and the arbiter resolution (2118 bi-segments). The Cohen's kappa coefficient for the English–Italian annotation task is 0.41.
The initial English–Spanish set had 3012 bi-segments. The first annotator annotated all bi-segments whereas the second annotator annotated 2708 bi-segments. The annotator

[6]http://mtequal.fbk.eu/
[7]labeled Annotator 1 and Annotator 2, respectively

Annotator	Annotator 2			
	Category	1	2	3
Annotator 1	1	**1413**	63	166
	2	203	**193**	107
	3	64	29	**470**

Table 2: The agreement for English–Spanish

Annotator	Annotator 2			
	Category	1	2	3
Annotator 1	1	**1629**	131	13
	2	23	**42**	3
	3	3	10	**15**

Table 3: The agreement for English–German

agreement is calculated for the 2708 common annotations and is reported in table 3.3..

The set to be used for training and testing for English-Spanish consists of the sum of all agreements (2076 bi-segments). The Cohen's kappa coefficient for the English–Spanish annotation task is higher than the same coefficient for English–Italian 0.57.

The initial English–German set had 3016 bi-segments. The first annotator annotated 2509 bi-segments and the second annotator annotated only 2404 bi-segments. However, the annotators chose to work on a different order and while annotator 1 started from the first segment, the second annotator chose to perform the annotation in reverse order, starting by the last bi-segment. The annotator agreement is calculated for the 1869 common annotations in table 3.3..

The Cohen's kappa coefficient for the English–German annotation task is 0.37. Two things can be observed relative to Table 3.3.: the number of bi-segments for which we have agreement is less than 2K (1686), just like in the English–Italian case, and the number of negative bi-segments (annotated with 3 by both annotators) is very low (15). To have a training and test sets comparable with the training and test sets for the other language pairs (English–Italian and English–Spanish), we added noise to the English–German set. We took 410 bi-segments annotated by one of the annotators and not by the other and added noise such as to transform them in 300 bi-segments annotated with 3 and 109 bi-segments annotated with 2. The set to be used for training and testing for English–German consists of the set of all bi-segments where both annotators agreed plus the 410 bi-segments to which we added noise (2096 bi-segments in total).

In conclusion, we selected three sets containing approximately 2K bi-segments where two annotators agreed. According to the interpretation that Landis and Koch (Landis and Koch, 1977) give to Cohen' kappa coefficient, the reported agreement coefficients is borderline between poor and fair. We have not conducted a study to see why the agreement is low. However inspecting a sample of disagreement cases we have noted that the annotators disagree when the translators bring into the translation process background knowledge that is not stated explicitly in the source sentence. For example the word "drug" in the source language can be translated as "the drug for dogs" in the target language when the information that the drug was meant to be for dogs was stated in the context before the segment to be translated.

4. Training and Test Sets

The training and test have been built using stratified sampling. This means that the training and test sets contain the same percentage of bi-segments with the same category label. Table 4. gives the number of bi-segments having the category labels "1", "2" and "3" in the training and test sets for all language pairs. The names of the columns E–G, E–S and E–I stand for English–German (E–G), English–Spanish (E–S) and English–Italian (E–I), respectively.

	Language Pair			Category Label
Training Set	E–I	E–S	E–G	
	872	942	1086	1
	254	128	100	2
	284	313	210	3
Test Set	E-I	E-S	E-G	
	437	471	544	1
	128	65	51	2
	143	157	105	3

Table 4: The size of the training and test sets

5. Baseline systems

To benchmark the results of the classifiers that the participants to the Shared Task will submit we have implemented two baselines. The first baseline generates random labels for the test set with the same distribution of the labels in the training set. The second baseline corrects the results of the first baseline when the Church-Gale (Gale and Church, 1993) score of the source and target segments is above a predefined threshold fixed to 2.5[8]. The idea is that if the difference in length between the source and target segments is too big, then it is likely that the target segment is not the translation of the source. Therefore in these cases we modified the score given by the first baseline to "3". To measure the length of the source and destination segments, we use the modified Church-Gale length difference algorithm (Tiedemann, 2011) presented in Equation 1:

$$CG = \frac{l_s - l_d}{\sqrt{3.4(l_s + l_d)}} \quad (1)$$

The results of the two baselines for all the shared tasks defined in section 2. are presented in table 5..

As stated earlier, we compute Precision, Recall and the F1 score for the two baselines defined before and each sub task defined in the shared task. It is expected that baseline 2 is harder to beat than baseline 1 (baseline 2 gains at most 3 points of F-score over baseline 1). With the exception of the Fine-Grained task, the baselines are not easy to beat, as they reach, in the case of the Binary Classification approximately 0.8 F-score points.

[8]The script that computes the baselines can be downloaded from the URL http://rgcl.wlv.ac.uk/resources/NLP4TM2016/baselines.py.remove

	Language Pair			Measure
	E–I	E–S	E–G	
Baseline 1 Fine-Grained	0.45	0.52	0.63	P
	0.45	0.52	0.63	R
	0.45	0.52	0.63	F1
	E-I	E-S	E-G	
Baseline 2 Fine -Grained	0.47	0.55	0.62	P
	0.47	0.55	0.62	R
	0.47	0.55	0.62	F1
	E-I	E-S	E-G	
Baseline 1 Binary Classification 1	0.61	0.68	0.78	P
	0.62	0.69	0.78	R
	0.61	0.69	0.78	F1
	E-I	E-S	E-G	
Baseline 2 Binary Classification 1	0.62	0.71	0.78	P
	0.62	0.69	0.77	R
	0.62	0.70	0.78	F1
	E-I	E-S	E-G	
Baseline 1 Binary Classification 2	0.8	0.77	0.85	P
	0.79	0.77	0.86	R
	0.79	0.77	0.85	F1
	E-I	E-S	E-G	
Baseline 2 Binary Classification 2	0.82	0.80	0.85	P
	0.79	0.77	0.85	R
	0.80	0.78	0.85	F1

Table 5: Baselines for the shared task

6. Conclusion

In this paper we have presented the methodology for constructing three sets of parallel bi-segments for English–Italian, English–Spanish and English–German sampled from the MyMemory translation memory database. We expected a higher agreement in the annotation task but due to time constraints to release the data for the shared task, we could not assess properly why the level of disagreement was so high. For English–Italian we needed an arbiter to decide a number a cases and thus achieve around 2K annotated examples where at least two annotators agreed in their annotations. The English–German set did not contain enough negative examples , meaning that MyMemory has good quality segments for this language pair. We have created some artificial negative segments by adding noise to the acceptable ones. We have implemented and presented two baselines to be compared against the classification results sent by the participants in the shared task.

7. Acknowledgments

The research reported in this paper is supported by the People Programme (Marie Curie Actions) of the European Union's Framework Programme (FP7/2007-2013) under REA grant agreement n^{o} 317471. Part of the work has been supported by the EC-funded project ModernMT (H2020 grant agreement no. 645487). We are grateful to Translated for giving us access to the MyMemory database. Last but not least we want to thank the 6 annotators who have annotated the data.

8. Bibliographical References

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, April.

Gale, W. A. and Church, K. W. (1993). A program for aligning sentences in bilingual corpora. *COMPUTATIONAL LINGUISTICS*.

Girardi, C., Bentivogli, L., Farajian, M. A., and Federico, M. (2014). Mt-equal: a toolkit for human assessment of machine translation output. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference System Demonstrations, August 23-29, 2014, Dublin, Ireland*, pages 120–123.

Landis, J. and Koch, G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Constantin Orasan et al., editors. (2015). *Proceedings of the First Workshop on Natural Language Processing for Translation Memories (NLP4TM-2015)*, RANLP 2015, Hissar, Bulgaria, September.

Tiedemann, J. (2011). *Bitext Alignment*. Number 14 in Synthesis Lectures on Human Language Technologies. Morgan & Claypool, San Rafael, CA, USA.

Trombetti, M. (2009). Creating the world's largest translation memory.

Investigating the Application and Evaluation of Distributional Semantics in the Translation of Humanistic Texts: a Case Study.

A. Bellandi[1], **G. Benotto**[1], **G. Di Segni**[2], **E. Giovannetti**[1]
[1] Istituto di Linguistica Computazionale "A. Zampolli", C.N.R., Via G. Moruzzi 1, Pisa - Italy
name.surname@ilc.cnr.it
[2] Collegio Rabbinico Italiano, Lungotevere Sanzio 14, Roma - Italy
cri@ucei.it

Abstract

Digital Humanities are persisting ascending and the need for translating humanistic texts using Computer Assisted Translation (CAT) tools demands for a specific investigation both of the available technologies and of the evaluation techniques. Indeed, humanistic texts can present deep differences from texts that are usually translated with CAT tools, due to complex interpretative issues, the request of heavy rephrasing, and the addition of explicative parts in order to make the translation fully comprehensible to readers and, also, stylistically pleasant to read. In addition, these texts are often written in peculiar languages for which no linguistic analysis tool can be available. We faced this situation in the context of the project for the translation of the Babylonian Talmud from Ancient Hebrew and Aramaic into Italian. In this paper we describe a work in progress on the application of distributional semantics to the informing of the Translation Memory, and on the evaluation issues arising from its assessment.

1. Introduction

Computer-Aided Translation (CAT) tools have become an essential component of translators' working environments. They allow to create more consistent translations, reduce repetitiveness, and, in general, increase the translation pace. One of the core components of a CAT tool is the Translation Memory System (TMS). TMSs leverage on a Translation Memory (TM) that is a sentence-pair database which automatically stores all translated text segments together with the source text during the translation process (Reinke, 2013). Basically, the main purpose of a TMS is to allow translators to reuse already done translations. A TMS works best with documents containing a large proportion of repeated or partially repeated text (e.g., formulaic expressions) such as software specifications, instruction and operator manuals, or domain-specific texts, like legal or medical ones. Even if humanistic texts usually present deeply different natures and purposes from those of texts that are usually translated with TMs, they can still share some features that make them suitable for being translated with the aid of a CAT tool. In literature, there are few works describing the application of TMs to humanistic texts (see the following section). However, the "Digital Humanities" are currently relentless ascending and the need for translating humanistic texts using CAT demands for a specific investigation both of the available technologies and of the evaluation techniques. Many humanistic texts (for example ancient writings) pose complex interpretative issues and their translation can require heavy rephrasing together with the addition of explicative parts in order to make the translation fully comprehensible to readers and, also, stylistically pleasant to read. In these cases, translators must: i) fully understand the text they are translating, ii) provide an "explicative translation" that has to be intelligible for readers and, iii) make the translation pleasant and fluent to read (for example by introducing paraphrases and synonyms).

As reported in the following section, some of the approaches aimed at enhancing the TMS adopt linguistic analysis techniques to the source text. However, a remarkable amount of interesting literary texts worth translating are written in languages for which there are no tools or resources available for automatic linguistic analysis. We faced this situation in the context of the Project for the translation of the Babylonian Talmud (BT), as we had to develop a TMS to support the translation of the Talmud from Ancient Hebrew and Aramaic into Italian. The main tool available for the automatic linguistic analysis of Hebrew, (Itai, 2006), has been developed for the analysis of modern Hebrew. To the best of our knowledge, there are no tools for the analysis of ancient Hebrew and Aramaic, thus making the "linguistic informing" of our TM impossible. Moreover, the BT attests to different linguistic stages of Hebrew and Aramaic, all of which are often alternated in the text. In order to find out an alternative strategy, we experimented the application of Distributional Semantics to extract semantically related words from the BT and integrate them, as a kind of "semantic informing", into the TM.

In Section 2, we depict our reference background, briefly illustrating either the TMS researches we took into account and some basic notions about Distributional Semantics. In Section 3 we describe the approach we adopted for semantically informing our TM, the evaluation of which raised some important issues we discuss in Section 4. To conclude, in Section 5 we discuss the weaknesses and the strong points of the proposed approach, and we outline the next steps of this work in progress.

2. Background

In general, a TM consists of, i) a database D containing pairs of segments (s, t), where s is the source language segment of text and t is its translation in the target language, ii) a similarity function Sim, and iii) a threshold σ having a value between 0 and 1. Given a segment s_q to be translated, the TMS returns a translation t_q by searching for the best match in D, i.e., a pair (s, t_q) whose similarity $Sim(s_q) = \sigma$ is maximal, if it exists (Sikes, 2007). The function

Sim measures the similarity between two source-language segments. Typically, it produces a percentage value, where 100% stands for "identical segments", i.e., exact match, and 0% for "completely different segments". Intermediate percentage values are called fuzzy matches; in this case the translator has to edit the proposed suggestions.
The similarity function of most systems is based on variants of the edit distance normalized over the length of the query segment, i.e., the minimum number of edit operations ($Edit(s_1, s_2)$) required to transform the string s_1 into s_2:

$$Sim_{ED}(s_q, s) = (1 - min(1, \frac{Edit(s_q,s)}{max(|s_q|,|s|)})) * 100 \quad (1)$$

In literature, there are two main kinds of approaches for the maximization of the function in Equation 1: approaches leveraging on Machine Translation (MT) confidence measures as in (Simard & Fujita, 2010), and approaches which integrate linguistic and semantic information in TMSs. A common methodology adopted in the first category is to first exploit the (manually made) translations contained in the TM and, then, automatically translate the unmatched lexical items using MT techniques (He et. al, 2010a), (He et. al, 2010b), (Smith & Clark, 2009), (Koehn & Senellart, 2010), (Wang et. al, 2014), (Dong et. al, 2014). Researches belonging to the second category are more similar to ours: they point out the need for similarity computation in TMSs to go beyond the simple surface form comparison. Planas & Osamu (1999) and Hodasz & Gabor (2005) propose to integrate the TM with lemmas and parts of speech along with surface form comparison. Pekar & Ruslan (2007) present an approach based on syntactic transformation rules, where they show how syntactic rules can bring a considerable help in retrieving more useful source segments. Masao et. al (2011), Gupta & Orasan (2014), and Ganitkevitch et.al (2013) propose approaches describing improvements in the retrieval of source segments with paraphrasing techniques. Wolff et. al (2014) proved that a similarity metric based on edit distance is likely to miss several useful suggestions: they found that the largest category of missing suggestions is composed of segments which were orthographically different but semantically similar. A common approach to the automatic extraction of semantically related textual chunks is to use the notion of Distributional Semantics (DS).
The assumption behind all DS models is that the notion of semantic similarity can be defined in terms of linguistic distributions (Miller & Charles, 1991). The distribution of an element can then be inferred from the sum of all its contexts, where a context is the setting of a word among the surrounding words i.e., *co-occurrency window*. One way to collect this information is to provide, for each word, a list of the co-occurrences of the word and the number of times they have co-occurred. The approach described by Schütze (1992,1993) became standard practice for word-space algorithms: data is collected in a matrix of co-occurrence counts (M), called a co-occurrence matrix. A cell m_{ij} of the co-occurrence matrix records the frequency of occurrence of word *i* in the context of word *j*. Such data is then used to build n-dimensional context vector, defined as the rows or columns of the matrix. Context vectors allow us to define (distributional) similarity between words in terms of vector similarity. A convenient way to compute the similarity between the vectors representing such words, is to calculate the cosine of the angles between the two vectors $\vec{w_i}$ and $\vec{w_j}$ representing the words.

3. Description of the Approach

The TM we worked with has been constructed from the collaborative translation carried on, during the last four years, by a group of 40 translators by means of the Traduco[1] web-based system (Albanesi et. al, 2015). The text being translated, the BT, represented a good test for our experimental approach, since, to date, there are no available tools suitable for processing ancient North-western Semitic languages, nor proper bilingual parallel corpora. We started by running the DS algorithm, described in (Baroni & Lenci, 2010), on the whole Talmud, by setting the co-occurrency window to 3[2]. Then we stored the extracted related word pairs and the relative cosine values in a database D_{sem}, to be used by the TMS with an updated similarity measure, that we call Sim_{ED+DS}. This measure is defined to exploit the relatedness between words by refining the weight of the substitution operation in the computation of the Edit() function in Equation 1: the closer the vectors representing the words are in the distributional space (i.e., the higher is the cosine of the angle between them), the lesser is the weight assigned to the substitution. For each word substitution operation found when comparing two source segments, the TMS looks for the relative word pair inside D_{sem} and sets the weight of such operation considering the associated cosine value, as shown in Table 1. Finally, the match between the two segments is scored by the updated Sim_{ED+DS} similarity function.

type of edit operation	weight
insertion: $\perp \rightarrow w_i$	1
substitution: $w_i \rightarrow w_j$	$1 - cos(\vec{w_i}, \vec{w_j})$
deletion: $w_i \rightarrow \perp$	1

Table 1: Weights assigned to the $Edit()$ function

Here is an example of two source segments s_1 =**פְּלָאֵי נִסִּי**, s_2 =**נִסִּים נַעֲשׂוּ לוֹ פְּלָאוֹת נַעֲשׂוּ לוֹ** differing on the words **נִסִּי** and **פְּלָאֵי** (meaning "miracles of" and "marvels of" respectively), and the words **נִסִּים** and **פְּלָאוֹת** (meaning "miracles" and "marvels" respectively), both translated into the segment *"saranno fatti miracoli di miracoli"* (meaning "miracles of miracles will be done"). By detecting the relations between the above words (synonymy and presence of conjunction), the two relative pairs have been scored with the cosine values 0.42 and 0.41, respectively. The similarity value calculated by means of the simple word-based edit distance on these two source segments would be $Sim_{ED}(s_1, s_2)$ = 50.00%, while the DS-boosted measure increases that value to $Sim_{ED+DS}(s_1, s_2)$ = 70.75%.

[1]Test Traduco at http://talmud-dev.ilc.cnr.it:8082/talmud (user&pass:traducodemo)

[2]The window size could vary according to the task and the data to analyze (Kruszewski & Baroni, 2014)

4. Evaluation Issues

To evaluate the contribution yielded by the DS component in the TMS, we initially decided to address the state of the art evaluation techniques, such as those cited in (Whyman and Somers, 1999), (O'Brien, 2012), (Bloodgood and Strauss, 2014), (Snover et. al, 2002), (Papineni et. al, 2002), (Lavie et al., 2004). This preliminary study highlighted how the evaluation of a "semantically informed" TM would not have been a trivial task. As a matter of fact, the reference corpora mentioned in literature appeared very different from the text translated in our case study (the BT), and, in general, from every humanistic texts requiring particular interpretative and explicative translations. To verify our hypothesis and estimate those differences, we selected two reference corpora among those most used to evaluate TMSs in literature and compared them with the BT.

The strategy we adopted to tackle the problem of evaluating the approach was carried out in three steps: i) evaluation of the quality of the related pairs extracted from a text which is considerably smaller than the ones typically used with DS algorithms (section 4.1), ii) evaluation of the impact of the proposed approach in terms of TM retrieval (section 4.2), and iii), evaluation of the translation quality of those segments for which the fuzzy match score was improved with the DS-boosted similarity measure (section 4.3).

4.1. Quality of the Related Word Pairs

Typically, DS models use large text corpora to derive estimates of relatedness between words, and it has been proved that massive quantities are required to match the quality advantage (Sridharan and Murphy, 2012). We anyway think that the (relatively) scarce size of our corpus could be balanced out, as shown further on in Table 3, both from a marked lexical poverty (TTR) and a high repetitivity of the formulaic expressions contained in the text (RR)[3]. In order to verify the quality of the extracted related word pairs, we relied on the expertise of a talmudist who manually analysed a selected set of word pairs SWP we constructed in the following steps: i) we collected all the different source segments translated with the same Italian sentence; ii) we selected those source segments having the same token length to ease the process of evaluation; iii) for each combination of these source segments, the expert identified the pairs of words in which they differed, thus obtaining 150 pairs; iv) to each pair we added the cosine value assigned by the DS algorithm. In addition to the usual lexico-semantic and morphological relations, the DS algorithm was able to detect word pairs involved in other kinds of relations. We identified five classes of relations:

- $idiomatic$: words having the same meaning but written in different languages, typically Hebrew and Aramaic, e.g. the words כְּשֵׁרִין and כְּשֵׁרִים both translated into "$adatti$" ("$suitable$");
- $orthographic$: words appearing in different writings, e.g. the words בַּזּוּגִין and בַּזְּגִין both translated into "$campanelli$" ("$bells$");
- $morphological$: words having the same root but different flexions, e.g. the words יוֹצֵא and יֵצֵא meaning "$exits$", and "$will\ exit$" both translated into "$esce$" ("$exits$");
- $syntactic$: words with or without articles or conjunctions, due to the agglutinating nature of the involved semitic idioms, e.g. the word מָקוֹם meaning "$place$", and the word הַמָּקוֹם meaning "$the\ place$", and words appearing in distinct sentences with a different order, e.g. רַב פַּפָּא אָמַר and אָמַר רַב פַּפָּא meaning "*Rabbi Papá said*" and "*said Rabbi Papá*" respectively;
- $lexico-semantic$: words involved in synonymic or quasi-synonymic relations, e.g. the word הִנִּיחַ meaning "*to put*" and נָתַן meaning "*to place*" or מִשּׁוּם meaning "*on behalf of*" and מְשַׁמֵּיהּ meaning "*as a representative of*".

Finally, the expert rejected 16 word pairs, by validating about 90% of the relations extracted by the DS algorithm.

4.2. Evaluation of the suggestions retrieval

For the comparison between Sim_{ED} and Sim_{ED+DS} we computed the recall of both in terms of source segments having 1 suggestion at least. We created a Talmud Test Set (TTS) by randomly choosing 500 translations from the Talmud corpus, and we considered the rest of the corpus as TM. Finally, we found the best-matching segment from the TM for each TTS segment according to the retrieval metrics Sim_{ED} and Sim_{ED+DS}. Table 2 shows the results.

	100%	**99%-85%**	**84%-70%**	**69%-55%**
Sim_{ED}	184	9	101	226
Sim_{ED+DS}	184	12	121	245
increase(%)	0%	33%	20%	8%

Table 2: Sim_{ED} and Sim_{ED+DS} recall in terms of source segments having 1 suggestion at least.

First of all, we can observe that Sim_{ED+DS} did not produce new exact matches (100%) w.r.t. the baseline. It was expected, since the distributional algorithm did not find pairs with a cosine value of 1. An increment of the exact matches has been achieved in other works, especially when the source languages were linguistically analyzable. An example is documented in (Gupta et. al, 2015) where the authors obtained around 13.76% increase in the retrieval of exact matches thanks to paraphrasing. It is interesting to note that the gains achieved w.r.t. Sim_{ED} are directly proportional to the category of similarity, meaning that Sim_{ED+DS} retrieves more segments in higher categories. This probably happens since many of the sentences in the text containing the contexts shared by the related words are (more or less) the same segments that users translate and that are stored inside the TM. The boosting approach improves the scores by 33% in category 99% - 85%, 20% in category 84% - 70%, 8% in category 69% - 55%.

We submitted the results to the expert talmudist, who confirmed their usefulness. However, a single evaluation is not

[3]For the RR measure we refer to (Bertoldi et. al, 2015).

statistically relevant. We are currently setting up an experiment involving the translators working with the Traduco system on the BT. To do so, as described in the next section, we need to understand to what extent the existing evaluation techniques can be applied to our case study.

4.3. Evaluation of the suggestions quality

As already depicted in section 1, the nature of the translation of a humanistic text like the Talmud is quite different from those typically faced with traditional CAT systems. In order to analyze this difference, we considered two other corpora and involving two language pairs, English-French and German-Italian. The first one is the OpenOffice (OO3) parallel corpus (Tiedemann, 2009), relative to the OO3 software documentation, and the second one is a portion of the EMEA parallel corpus (Tiedemann, 2009) constituted of medical documents from the European Medicines Agency. Table 3 shows that the two source texts of the parallel corpora under examination present a difference in terms of lexical variety and repetitivity w.r.t to Talmud, whose data are shown in Table 3.

Text Corpus (language)	#Tokens	#Types	TTR	RR
Talmud (hebrew/aramaic)	533400	52352	0.098	0.117
OO3 (english)	398646	22074	0.05	0.148
EMEA (german)	1150649	24631	0.021	0.115

Table 3: Texts statistics.

Since it was not possible to linguistically analyse the source language of the BT, we did not stem or lemmatize the source languages of these two corpora either. To calculate the necessary statistics we just removed all punctuation marks and numbers and we lowercased the texts. Table 4 shows that the ratio between the number of target tokens and source tokens is more or less the same for the two reference corpora, while the one relative to the Talmud is almost the double (2.938). Indeed, the translation of the BT is seldom literal, being usually enriched with explicative additions. In particular, the ratio between the target tokens related to the contextual part and the source tokens is 1.20, suggesting that almost the 42% of each source segment translation consists in an explicative part. Since explicative additions can provide to translators an important help in the translation of particularly challenging source segments, they cannot be removed from suggestions, for example, to attempt a direct (word-by-word) translation. For the aforementioned reasons, manually or automatically evaluating the quality of the provided suggestions is a really complicated task. Similarly, evaluations based on post-editing and keystrokes are probably not suitable for our translation context. The translation of a new segment which has been (even partially) already translated, could be significantly different and require, for example, a greater number of words in the target segment to correctly explain the sentence in a different context. For this reason even automatic evaluations, based on assessing how close a translation is w.r.t. to a gold, such as (Snover et. al, 2002), (Papineni et. al, 2002), and (Lavie et al., 2004) would have not been suitable. Furthermore, I_{VAR}[4] shows that the Talmud has a higher variability while translating identical source segments, w.r.t. the other evaluated corpora. For example, similar segments could require really different translation times, for example when one of them contains a peculiar word like *Shemá*, that has to be properly explained using an annotation.

	Talmud TM	OO3 TM	EMEA TM
#segments	124924	72371	106579
#source tokens	533400	398646	1150649
#target tokens	1567645	451286	1445491
target/source ratio	2.938	1.42	1.332
#literal-target tokens	924986	-	-
context-target tokens	642659	-	-
literal-target/source ratio	1.73	-	-
context-target/source ratio	1.20	-	-
I_{VAR}	2.97	1.426	1.147

Table 4: TM comparison.

5. Conclusion

The application of CAT tools to the translation of humanistic texts requires careful reflection on the applicability of the usual techniques, and, in particular, on the ways each research advancement can be evaluated. In this work, still in progress, we have investigated the contribution of a distributional based semantic informing of a TM in the translation of texts i) written in languages not served by linguistic analysis tools and resources, ii) posing particular challenging interpretative issues, and iii) requiring explicative translations to be understood by a non-scholarly audience. The approach has been experimented in the context of the translation of the BT into Italian. The selected portion of the Talmud we used for the experiment, though being relatively small in size, provided significative results. As reported in the relative section, there are a number of evaluation issues arising from the assessment of a TM informing technique applied to humanistic texts. As a matter of fact, apart from documenting what seems to be a promising linguistically agnostic approach for the semantic informing of a TM, the purpose of this work is to provide a starting point of discussion about the application and evaluation of CAT tools for the translation of texts which can be substantially different from those typically used in literature.

6. Acknowledgements

This work has been conducted in the context of the project TALMUD and the scientific partnership between S.c.a r.l. "Progetto Traduzione del Talmud Babilonese" and ILC-CNR and on the basis of the memorandum of understanding between the Italian Presidency of the Council of Ministers, the Italian Ministry of Education, Universities and Research, the Union of Italian Jewish Communities, the Italian Rabbinical College and the Italian CNR (21/01/2011).

[4]The ratio between the number of all the possible translations for the same source entry of the TM and the number of all source entries occurring more than once in the TM.

7. Bibliographical References

Albanesi, D., Bellandi, A., Benotto, G., Di Segni, G. & Giovannetti E. (2015). When Translation Requires Interpretation: Collaborative Computer-Assisted Translation of Ancient Texts. In Proceedings of the 9^{th} Workshop on Language Technology for Cultural Heritage, Social Sciences (ACL Special Interest Group on Language Technologies for the Socio-Economic Sciences and Humanities), pp.84-88.

Baroni, M. & Lenci, A. (2010). Distributional Memory: A General Framework for Corpus-based Semantics. Computational Linguistics, 36(4), pp. 673-721.

Bertoldi, N., Simianer, P., Cettolo, M., Wäschle, K., Federico, M. & Riezler, S. (2014). Online Adaptation to Post-edits for Phrase-based Statistical Machine Translation. Machine Translation, 28(3-4), pp.309-339.

Bloodgood, M. & Strauss, B. (2014). Translation Memory Retrieval Methods. In Proceedings of the 14^{th} Conference of the European Chapter of the Association for Computational Linguistics, (Association for Computational Linguistics), pp. 202-210.

Christodouloupoulos, C. & Steedman, M. (2014). A Massively Parallel Corpus: The Bible in 100 Languages. Language Resources and Evaluation, 49(2), pp 375-395.

Dong M., Cheng Y., Liu Y., Xu J., Sun M., Izuha T., & Hao J. (2014). Query Lattice for Translation Retrieval. In Proceedings of COLING 2014, pp. 2031-2041.

Ganitkevitch, J., Benjamin, V. D. & Callison-Burch, C. (2013). Ppdb: The Paraphrase Database. In Proceedings of NAACL-HLT, pp. 758 - 764.

Gupta, R., Orasan, C., Zampieri, M., Vela, M., & Van Genabith, J. (2015). Can Translation Memories Afford not to Use Paraphrasing?. In Proceedings of EAMT.

Gupta, R.& Orasan, C. (2014). Incorporating Paraphrasing in Translation Memory Matching and Retrieval. In Proceedings of the 17^{th} Annual Conference of EAMT.

Harris, Z. S. (1970). Distributional structure. Springer.

He Y., Ma Y., van Genabith J., & Way A. (2010a). Bridging SMT and TM with Translation Recommendation. In Proceedings of the 48^{th} Annual Meeting of the Association for Computational Linguistics, pp. 622-630.

He Y., Ma Y., van Genabith J., & Way A. (2010b). Integrating n-best SMT Outputs into a TM System. In: Proceedings of the 48^{th} Annual Meeting of the Association for Computational Linguistics, pp. 374-382.

Hodasz, G. & Gabor P. (2005). MetaMorpho TM: a Linguistically Enriched Translation Memory. In International Workshop, Modern Approaches in Translation Technologies (ed. Walter Hahn, John Hutchins, Cristina Vertan), pp. 26-30.

Koehn P. & Senellart J. (2010). Convergence of translation memory and statistical machine translation. In AMTA Workshop on MT Research and the Translation Industry, pp. 21-31.

Kruszewski, G. and Baroni, M. (2014). Dead Parrots Make Bad Pets: Exploring Modifier Effects in Noun Phrases. In Proceedings of *SEM 2014 (Third Joint Conference on Lexical and Computational Semantics), East Stroudsburg PA: ACL, pp. 171-181.

Itai, A. (2006). Knowledge Center for Processing Hebrew. In Proceedings of the workshoptutorial of the 5^{th} International Conference on Language Resources and Evaluation (LREC), pp.34-38.

Landauer, T. K. & Dumais, S. T. (1997). A Solution to Plato's Problem: The Latent Semanctic Analysis Theory of the Acquisition, Induction, and Representation of Knowledge. Psychological Review, 104, pp. 211-140.

Lavie, A., Sagae, K., & Jayaraman, S. (2004). The Significance of Recall in Automatic Metrics for MT Evaluation. In Proceedings of the 6th Conference of the Association for Machine Translation in the Americas (AMTA-2004), pp. 134-143.

Masao, U., Neubig, G., Onishi, T. & Sumita, E. (2011). Searching Translation Memories for Paraphrases. In Machine Translation Summit XIII, pp. 325-331.

Miller, G. A. & Charles, W. G. (1991). Contextual correlates of semantic similarity. Language and Cognitive Processes, 6(1), pp. 1-28.

O'Brien, S. (2012). Towards a Dynamic Quality Evaluation Model for Translation. Journal of Specialized Translation.

Papineni, K., Roukos, S., Ward, T., & Zhu, W. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation. In Proceedings of 40th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 311-318.

Pekar, V. & Ruslan M. (2007). New Generation Translation Memory: Content-Sensivite Matching. In Proceedings of the 40^{th} Anniversary Congress of the Swiss Association of Translators, Terminologists and Interpreters.

Planas, E. & Osamu F. (1999). Formalizing Translation Memories. In Proceedings of the 7^{th} Machine Translation Summit, pp. 331-339.

Reinke, U. (2013). State of the Art in Translation Memory Technology. Translation: Computation, Corpora, Cognition, 3(1) 2013.

Sahlgren, M. (2006). The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces. Ph.D. thesis, Department of Linguistics, Stockholm University ISBN: 91-7155-281-2.

Schütze, H. (1992). Dimensions of Meaning. In Proceedings of the 1992 ACM/IEEE conference on Supercomputing (Supercomputing 92), pp. 787-796.

Schütze, H. (1993). Word space. In Advances in Neural Information Processing Systems 5, pp. 895-902.

Sikes, R. (2007). Fuzzy Matching in Theory and Practice. Multilingual, 18(6), pp. 39-43.

Simard M., Fujitaì, A. (2010). A Poor Man's Translation Memory Using Machine Translation Evaluation Metrics. In Proceedings of the 10^{th} Biennial Conference of the Association for Machine Translation in the Americas (AMTA).

Smith J. & Clark S. (2009). Ebmt for SMT: A New EBMT-SMT Hybrid. In Proceedings of the 3^{rd} Workshop on Example Based Machine Translation, pp. 3-10.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., & Makhoul, J. (2006). A Study of Translation Edit Rate

with Targeted Human Annotation. In Proceedings of the 7th Conference of the Association for Machine Translation in the Americas (AMTA-2006), pp. 223-231.

Sridharan, S., & Murphy, B. (2012). Modeling word meaning: Distributional semantics and the corpus quality-quantity trade-off. In Proceedings of the 3rd Workshop on Cognitive Aspects of the Lexicon COLING, pp. 53-68.

Wang K., Zong C., & Su K.Y. (2014). Dynamically Integrating Cross-domain Translation Memory into Phrase-based Machine Translation During Decoding. In Proceedings of COLING 2014, pp. 398-408

Whyman, E. K., & Somers, H. L. (1999). Evaluation Metrics for a Translation Memory System. Software-Practice and Experience 29(14), pp. 1265-1284.

Wolff, F., Pretorius, L., & Buitelaar, P. (2014). Missed Opportunities in Translation Memory Matching. In Proceedings of the 9^{th} International Conference on Language Resources and Evaluation (LREC), pp. 4401-4406.

Beyond Translation Memories: Generating Translation Suggestions based on Parsing and POS Tagging

Tapas Nayak[1], Santanu Pal[2], Sudip Kumar Naskar[1], Sivaji Bandyopadhyay[1], Josef van Genabith[2,3]
Jadavpur University, India[1]
Saarland University, Germany[2], German Research Center for Artificial Intelligence (DFKI), Germany[3]
tnk02.05@gmail.com, {santanu.pal, josef.vangenabith}@uni-saarland.de,
sudip.naskar@cse.jdvu.ac.in, sivaji_cse_ju@yahoo.com

Abstract

This paper explores how translations of unmatched parts of an input sentence can be discovered and inserted into Translation Memory (TM) suggestions generated by a Computer Aided Translation (CAT) tool using a parse tree and part of speech (POS) tags to form a new translation which is more suitable for post-editing. CATaLog (Nayek et al., 2015) is a CAT tool based on TM and a modified Translation Error Rate (TER) (Snover et al., 2006) metric. Unmatched parts of the sentence to be translated can often be found in some other TM suggestions or in sentences which are not part of TM suggestions. Therefore, we can find the translations of those unmatched parts within the TM database itself. If we can merge the translations of the unmatched parts into one single sentence in a meaningful way, then post-editing effort will be reduced. Inserting the translations for the unmatched parts into TM suggestions may lead to loss of fluency in the generated target sentence. To avoid that, we use parsing and POS tagging together with a back off POS n-gram model to generate new translation suggestions.

Keywords: Computer Aided Translation, Post-editing, Translation Memory based CAT tool

1. Introduction

Computer-aided translation tools (CAT) are widely used by language service providers, freelance translators to improve translation quality and to increase translator's productivity. CATaLog (Nayek et al., 2015) is a CAT tool developed based on Translation Memory (TM). CATaLog uses modified TER (Snover et al., 2006) as the similarity metric. It introduced the concept of color-coding the TM suggestions both in the source language and the target language. Matched and unmatched parts are color coded in green and red, respectively, to facilitate post-editing and to guide the user. The intuition behind the color coding scheme in CATaLog is that the more green color in a TM suggestion, the more matching and hence less post-editing effort is required. It also provides options for length based sentence pruning and indexing techniques to minimize search time. CATaLog has been specifically designed to improve user experience with TM. In this paper, we report additional functionality to the CATaLog tool and TM technology in general.

TM tools traditionally do not generate any translation; instead they present the user with matching sentence pairs that are similar to the sentence being translated. Post-editors, when working with TM tools, seldom find an exact match. Therefore, almost all the times, the TM suggestions contain at least a few unmatched parts. However, it can often be observed that the translation for those unmatched parts are available in other suggestions or in some other sentences in the TM database. If we can extract the translations for those unmatched parts from other sentences and introduce them into the suggested TM translations, then we can generate almost the entire translation for a particular input sentence. While filing those unmatched parts may lead to a loss of fluency in the suggested new translation, often it improves the accuracy of the suggested translation. Thus, it reduces the post-editing cost significantly since the user does not have to type in the entire translation for the unmatched parts. We introduced this new capability to the CATaLog tool which is reported in this paper.

2. Related Work

CAT tools are very popular among professional translators. They use these tools in their translation workflows on a regular basis to reduce translation time and improve productivity (Lagoudaki, 2008). Along with basic research on CAT tools, some researchers tried to fill the gaps for the mismatched parts in the input sentence in different ways. Biçici and Dymetman (2008) combined dynamically extracted source-target phrase pairs from the TM with the phrase table of a phrase-based SMT (PB-SMT) system. Translation for a mismatched part is taken from the SMT phase table and the translation is replaced in the target sentence. Simard and Isabelle (2009) combined TM with PB-SMT to enable PB-SMT to take advantage of exact or fuzzy matching features of TM. They proposed two different strategies: (i) an MT-TM combined system where, above a certain similarity threshold value, the combined system provides the translation from the TM, otherwise it produces the MT output; and, (ii) it allows the PB-SMT system to actively exploit the most similar material identified by the TM, via TM-based feature functions. Zhechev and van Genabith (2010)

explored similar strategies, but use syntactic information during fuzzy matching by applying sub-tree alignment in order to link nodes between the input sentence, TM match and TM translation. Sub-tree based alignments reliably determine the correspondences between an input sentence and a TM suggestion. An SMT system is used to translate the mismatched parts of the input sentence. The complete translation ensures higher quality than the TM suggestions. Koehn and Senellart (2010) proposed a similar method to combine MT with TM. They used fuzzy matching to retrieve similar segments from the TM for each source segment that needs to be translated and identified the mismatched parts using automatic word alignment. Finally, those mismatched parts are replaced by SMT translations. Ma et al. (2011) use support vector machine and discriminative learning methods to identify the matched words to select a translation unit. They addressed several problems in fuzzy matching based translation unit identification. In general, translation units with lower fuzzy match value are thrown away, however, their method considers those units during translation.
Dandapat et al. (2011) also worked on identifying unmatched parts of the input sentence and replacing their translations in the TM candidate translation. From the original translation memory, they first identified sub-segment level translation pairs which form a sub-segment TM. When some unmatched sub-segment is found in the input sentence, they look for its translation in the sub-segment TM. They did not consider the context of the unmatched sub-segments while inserting their translations; they just plugged those sub-segment translations into the target sentence based on how they appear in the input sentence.

3. System Description

CATaLog generates top 5 suggestions based on modified TER. Whenever the post-editor chooses one suggestion for post-editing, the CAT system tries to fill the unmatched parts of that suggestion and presents the user with a new translation suggestion. The system components are detailed in the following subsections.

3.1. Generating Dictionary

In our approach presented in this paper, we try to fill the unmatched parts of a TM suggestion at the word level. Whenever we find some unmatched words in the input sentence to be translated, we need to look for their translation(s) somewhere. One way of achieving this is to keep a bilingual dictionary. However, a dictionary is a costly resource for many language pairs. Therefore, rather than using a built-in dictionary, we generate a dictionary from the background bilingual corpus available with the translation memory. For illustration purposes, all the examples presented in this paper are in English–Bengali obtained from an English–Bengali parallel corpus with 13,000 sentences. English is considered as the source language and Bengali as the target language in the present work. We generate our English to Bengali dictionary from the parallel corpus where English words are stored along with their parts of speech information and their corresponding translations in Bengali. In the present work we used the Stanford POS tagger[1] to generate the POS tags for the source side of the parallel corpus. The GIZA++ (Och and Ney, 2003) implementation of the IBM word alignment model (Brown et al., 1993) is used to produce one to many alignments between source and target language words. From these source–target alignments we find the translation correspondences of each English word available in the parallel corpus. This dictionary is generated offline once and for good and it gets loaded when the TM application is loaded.
The meaning of a polysemous word depends on the context it appears in. In case of translation, a source word can have completely different translations or may have different suffixes attached to it based on its context. Therefore, to determine which translation is more accurate in a particular context, we look at the neighboring context. Instead of considering the lexical context, we take into consideration the POS context in this work. In our current system, we use a trigram back-off model for determining the contextual translation of a source word. We generate three dictionaries: one is trigram context based, the second one is bigram context based and the third one is simply a unigram dictionary. Here context refers to a POS sequence context. In the trigram contextual dictionary, for a particular source word, we store the previous two POS tags, the POS tag of the word under consideration and the next two POS tags. We also store the frequency of this entire context tag sequence (in the training corpus) along with the particular translation for that word. Trigram context based dictionary entries look as follows:

Example 1. *bottle: VBP_DT_NN_IN_NN/2/* এক বোতল*/1; PRP_CD_NN_IN_NN/2/*বোতল*/1;*

The example given above is the dictionary entry corresponding to the word 'bottle'. Here the POS tag sequence is VBP_DT_NN_IN_NN. Number '2' represents the zero based positional index of the POS tag of the word. 'এক বোতল(*botol*)' is the corresponding translation. The number '1' appearing at the end represents the frequency of this translation in the training corpus for the word 'bottle' for this particular POS context.
We follow the same format for all the three dictionaries. The entries in the bigram and unigram context based dictionaries corresponding to the word 'bottle' are given below.

Example 2.

bottle: DT_NN_IN/1/ এক বোতল*/1; CD_NN_IN/1/*বোতল*/1; ...*

*bottle: NN/*বোতল*/7; NN/* এক বোতল*/6; NN/*বোতল খুলে*/1; NN/*বোতল */1; ...*

[1] http://nlp.stanford.edu/software/tagger.shtml

In order to find the translation of a non-matching word in the input sentence, we first look at the trigram dictionary; if no match is found there, we look for a match in the bigram dictionary and finally back-off to the unigram dictionary. If multiple matches are found in any particular dictionary, we choose the most frequent translation from among them. In case of a frequency tie, which is very unlikely, we choose any one of them randomly. While doing the POS context matching, we first try to get an absolute match first. If there is no absolute POS context match, then we look for approximate POS context match. This concept of absolute POS context match and approximate POS context match is explained in next section.

3.2. Grouping Part Of Speech Tags

We used the Stanford POS tagger to generate the POS tag sequence for the input sentence. The use of POS tags is explained in a later section of the paper. The Stanford tagger uses the Penn Treebank tagset[2] which contains 38 different types of POS tags. However, since we intend to generate a translation for use in post-editing, we relax the constraint of exact POS tag match. Therefore, we group together similar POS tags into coarse grained POS categories to get more matches between the input sentence and the TM suggestions. E.g., we group together VB, VBD, VBZ, VBN, VBG, and VBP into a coarse grained group called VB (i.e., verb). Similarly we group JJ, JJR, and JJS into a JJ group. NN, NNS, NNP, NNPS are grouped into the NN group. POS tags like RB, RBS, and RBR are grouped into the RB group. When performing POS tag matching between the input sentence and TM suggestion, first we look for an absolute match, i.e., POS tag match. In case of no absolute match we go for matching at basic POS category level.

3.3. Finding Translations for Unmatched Parts

Since CATaLog uses the TER metric as the measure of similarity between the input sentence and the TM database, as a byproduct, TER also provides the alignment between input sentence and selected TM suggestion sentences. From this alignment we can easily find out which words of the input sentence do not match with the suggested sentence.

The TER alignment is shown in Example 3 where 'M', 'D', 'I' and 'S' correspond to match, deletion, insertion and substitution operations, respectively. 'I' and 'S' editing operations in the TER alignment correspond to the unmatched words between the input sentence and the corresponding TM suggestion. The 'D' editing operation corresponds to the deletion of extra words that are not present in the input sentence and this editing operation is easily catered to by simply deleting the corresponding word(s) in the target identified through word alignments. In the CATaLog tool these words are shown in red color. 'I' signifies that the corresponding input sentence word has to be inserted, while an 'S' represents the substitution of the TM suggestion word for the input sentence word. In either case, the translation of the unmatched input sentence word has to be inserted to generate the translation of the input sentence.

If we do not consider the POS of the unmatched word in the input sentence, we could only use the most frequent translation of that word from the dictionary. In that case we also do not need to store the POS information in the dictionary. However, considering the POS of the unmatched word allows us to be more specific about its translation in the context (e.g., book: NN|বই; VBD|সংরক্ষণ করা). When the unmatched word 'book' in input sentence is identified as an NN, the system provides the translation "বই (*boi*)"; similarly, when the POS is VBD, then the system prefers the translation "সংরক্ষণ করা (*sangrokshon kora*, English gloss: to reserve)". POS based dictionary matching reduces the POS level ambiguity. If the word is not present in the dictionary it remains untranslated.

While matching the POS tag we might not find an exact POS tag match. In that case we try to find an approximate POS match, i.e., at the coarse-grained POS category level. For example, if the 'book|VBD' word|POS combination can not be found in the dictionary, we look for a dictionary entry for the word 'book' together with POS category VB, i.e., any of VB, VBD, VBZ, VBN, VBG, and VBP. If, for example, the 'book|VBZ' combination is found, the corresponding translation is taken into consideration. The idea is that the translation of 'book|VBD' might not be exactly the same as 'book|VBZ'; however, they are derived from the same root verb and thus a little post-editing effort would result in the correct translation.

3.4. Finding Positions to Insert Translations

After obtaining the translations of all the unmatched words, we need to find out where to put the target language words in the selected TM translation. If the target language words are not put in appropriate places, the suggested new translation becomes less fluent and ineligible for post-editing. TM, despite being technologically very simple, has proved itself to be a widely used technology in the localization industry mainly because of its strength that it presents the user with perfectly fluent translation suggestions for post-editing. Thus, presenting the user with a more accurate but less fluent translation suggestion might not be acceptable. Our idea is to use POS tagging and parsing to guide the identification of the proper location for insertion of the translation of the unmatched word. To achieve this we subject both the input sentence and the selected suggestion sentence to POS tagging and parsing. POS tagging and parsing are performed using the Stanford POS tagger[3] and Stanford parser[4],

[2] https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

[3] http://nlp.stanford.edu/software/tagger.shtml

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

respectively.

3.5. Matching using POS n-grams

When we search for the location for inserting the translation of an unmatched word, we first try to find a corresponding word in the TM suggestion sentence that does not match with any word in the input sentence. Successively, we find the words and their positions in the target side of the parallel sentence that the unmatched source word corresponds to. Those positions are the potential positions where the translation of the unmatched word can be put. The corresponding word in the TM suggestion sentence can be found using the POS tag of the unmatched word. We try to find the same POS tag in source suggestion sentences and the corresponding word should not match with any other word in input sentence. Once we find such a corresponding word, we mark it so that the next time when we try to find another corresponding word for another unmatched word we do not consider it again. While matching the POS tag, first we consider the complete POS tag match. If we do not find any such match, we go for a POS tag which belongs to same POS category as described in the previous section.

We have used a back off POS trigram model for searching the location of the corresponding word. In this model, an n-gram represents a sequence of three consecutive POS tags. Since we consider trigram POS sequences, we have to take into consideration three different trigrams. We test each of the three POS trigrams individually. If none of them matches with any POS trigram sequence in the selected suggestion translation, we back off to POS bigram matching. If multiple matches are found, we resolve the ambiguity using parse tree information of the input sentence to determine which trigram sequence is more suitable. This parse tree matching process is detailed in the following subsection. If we do not find any higher order n-gram match, we fall back to unigrams. If the system fails to find even a unigram POS tag match (i.e., word|POS), then the unmatched word in the input sentence remains un-translated. For such words, the system disregards the POS of the word and provides a drop down list of probable translations which become available on right click of the mouse from which the user can directly choose a translation (as opposed to typing) by left click of the mouse and can put the target word in a proper place.

3.6. Parse Tree Matching

When multiple POS n-gram matches are found, the system resolves this ambiguity using the parse tree of the input sentence. For all the higher order POS n-gram matches, we determine the lowest common ancestor node in the parse tree. The n-gram POS sequence choice for which the depth of the common ancestor node is maximum is considered as the winner. If there is a tie, the system chooses one among them randomly. The idea behind choosing the lowest common (i.e., maximum depth) ancestor is that the lower the common ancestor in the parse tree, the more syntactically coherent they are. If the lowest common ancestor is located at the top of the sub-tree, the words considered in the n-gram sequence are unrelated and hence they should be ignored. This motivates our philosophy behind using the lowest common ancestor.

After we have found the location of the corresponding word in the selected TM suggestion, we determine the positions of translation of that word in the translation of that TM suggestion using the alignment generated by GIZA++. These positions in the TM suggestion translation are the potential positions where the translation of the unmatched word could be put. Since GIZA++ generates one to many alignments from source to target, three situations can arise here. Let w_1 be the unmatched source word in a TM suggestion translation and w_2 be the unmatched source word in the input sentence.

The length of the translation of w_2 could be equal to, shorter, or longer than the length of the translation of w_1.

We define the length in terms of number of words. The potential positions for inserting the translation of w_2 may be continuous or discontinuous in the TM suggestion translation. If the translation of w_1 has the same length or is longer than the translation of w_2, then we just replace the translation of w_2 in those positions. We replace one word in translation of w_1 with one word from the translation of w_2. Therefore, our system will work properly even if the potential positions of insertion are not continuous. Some positions will not be replaced in case the translation of w_1 is longer than the translation of w_2. However, if the translation of w_1 is shorter, we merge the extra words of the translation of w_2 with the last word of the translation of w_2 (separated by space) and put this merged word in the last position of translation of w_1. In this way we place the translation of unmatched word(s) of the input sentence in the TM translation suggestion.

The process is illustrated below with two examples. In the following two examples, for the sake of simplicity, we just make use of the unigram dictionary to get the translation for the unmatched words. However, in the actual system, a trigram back-off model is used.

Example 3.

Input sentence: *you gave me wrong number .*

TM Match: *you gave me right number.*

Translation: আপনি আমাকে ঠিক নম্বর দিয়েছেন .

Translation gloss: *Apni amake thik nombor diyechen.*

POS tag sequence for input sentence: *you/PRP gave/VBD me/PRP wrong/JJ number/NN ./.*

POS tag sequence for suggestion: *you/PRP gave/VBD me/PRP right/JJ number/NN ./.*

Here the unmatched word in the input sentence is: 'wrong'. The unigram dictionary entry for 'wrong' is: wrong: JJ|ভুল (vul) ; JJ|গন্ডগোল (gondogol)

The system looks up in the dictionary the translation of 'wrong' with POS tag JJ and finds the two translations as shown above. The system can choose anyone since it has no other knowledge to resolve the ambiguity. Let us consider that the system chooses the first translation, i.e.,JJ|ভুল.
The three trigram POS sequences considered for matching are: 'gave/VBD me/PRP wrong/JJ'; 'me/PRP wrong/JJ number/NN'; 'wrong/JJ number/NN ./.'
Two bigram POS sequences: 'me/PRP wrong/JJ'; 'wrong/JJ number/NN'
Unigram POS sequence: 'wrong/JJ'
The system gets a matching POS sequence in the TM suggestion sentence for all the three trigram sequences involving the unmatched word 'wrong/JJ'. So, it uses the parse tree information to resolve this ambiguity. The parse tree of the input sentence is shown in Figure 1.

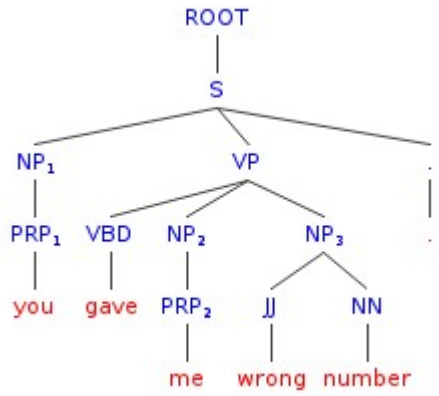

Figure 1: Parse tree of " you gave me wrong number."

ROOT is at depth 0. Here the lowest common ancestor (LCA) for POS sequence "gave/VBD me/PRP wrong/JJ" is VP which is at depth 2. LCA for POS sequence "me/PRP wrong/JJ number/NN" is also at depth 2. LCA for POS sequence "wrong/JJ number/NN ./." is at depth 1 (e.g., in figure 1, the LCS of NP_3 and '.' is S, which is depth 1). Therefore, the third POS sequence can be ignored and we can choose anyone from the first two.
If we consider the bigram sequence 'me/PRP wrong/JJ', then the lowest common ancestor is at depth 2, but for the sequence 'wrong/JJ number/NN' the lowest common ancestor is at depth 3. Therefore, we should choose the second bigram sequence for better output. However, for this particular input sentence the system will not go for a bigram match as it already found a trigram match.
The GIZA++ alignment between the TM match and the corresponding translation (cf. Example 4) is as given below.

1-1, 2-5, 3-2, 4-3, 5-4, 6-6

Thus, the translation of 'right' is at position 3 of target translation. Therefore, at position 3 of the target translation the system puts the translation ('ভুল') of the unmatched word and produces the following translation.

আপনি আমাকে ভুল নম্বর দিয়েছেন .

Let us consider another example.

Example 4.

Input sentence: *i would prefer something in a middle price range .*

TM suggestion: *i would prefer to sit in the back part of the plane .*

TER alignment: *MMMDSMDDSSSSM*

TM suggestion translation: আমি বিমানের পিছনের অংশে বসতে পছন্দ করব .

POS sequence for the input sentence: *i/FW would/MD prefer/VB something/NN in/IN a/DT middle/JJ price/NN range/NN ./.*

POS sequence for TM suggestion : *i/FW would/MD prefer/VB to/TO sit/VB in/IN the/DT back/JJ part/NN of/IN the/DT plane/NN ./.*

Table 1 shows the TER alignment between the TM source suggestion and the input sentence along with the edit operations required to turn the TM source suggestion into the input sentence. Table 1 also shows the word alignment information between the source and target sides of the TM suggestion.
The unmatched words in the input sentence in this case are 'something', 'a', 'middle', 'price', and 'range'
Unigram dictionary entries for the unmatched words are :

something: NN|একটা কিছু ; NN|কিছু ; NN|কোন কিছু ; NN|কিছু একটা

a: DT|একটা ; DT|কোন ; DT|এক

middle: JJ|মাঝারি আকারের ; JJ|মাঝের

price: NN|দাম ; NN|দামটা ; NN|মূল্য

range: VBP|দেড়শ এর মধ্যে বদলাতে থাকে|

The CATaLog system searches for matching translation examples for those unmatched words in the same context as they appear in the input sentence. Here that sequence is 'something, 'a', 'middle', 'price', and 'range'. For the word 'something/NN', the 3 trigram sequences used for search are: 'would/MD prefer/VB something/NN'; 'prefer/VB something/NN in/IN'; 'something/NN in/IN a/DT'. The system found a match with the third trigram sequence in TM suggestion. The match sequence is 'part/NN of/IN the/DT' where the word 'part/NN' does not match with any word of input sentence. So system does not go for bigram or unigram matching search.
Now the system goes for searching the position where the translation of 'part' is located in TM translation.

TM Target Suggestion	TM Source Suggestion	Input Sentence	Edit Operation
আমি	i	i	M
-	would	would	M
পছন্দ করব	prefer	prefer	M
-	to	-	D
বসতে	sit	something	S
-	in	in	M
-	the	-	D
পিছনের	back	-	D
অংশে	part	a	S
-	of	middle	S
-	the	price	S
বিমানের	plane	range	S
.	.	.	M

Table 1: TM source–target Alignment and TM source–Input sentence Alignment

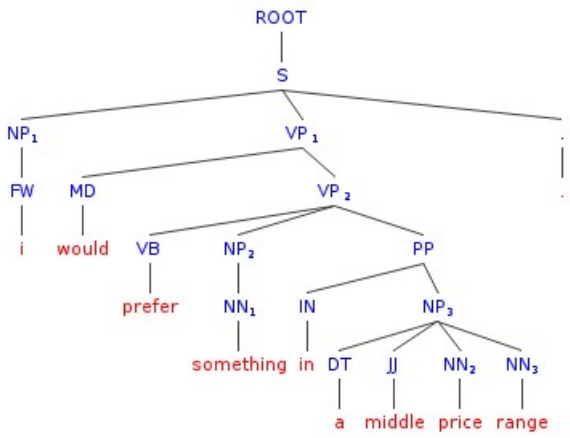

Figure 2: Parse tree of "i would prefer something in a middle price range."

For that it uses GIZA++ alignment. Below is the GIZA++ alignment for the TM suggestion.

1-1, 3-6, 3-7, 5-5, 8-3, 9-4, 12-2, 13-8

Here the position index before the hyphen (-) is the word position in the TM source suggestion and the position index after hyphen (-) is the word position in the TM suggestion translation. 'part' is the 9th word in the TM source suggestion and according to the GIZA++ alignment its translation is the 4th word in the translation, i.e., 'অংশে (*angse*)'. The target language word 'অংশে' is replaced by 'একটা কিছু (*ekta kichu*)', the translation of the unmatched input sentence word 'something'. Therefore the suggestion translation is modified as follows.

আমি বিমানের পিছনের একটা কিছু বসতে পছন্দ করব .

Next, tries to find a match for 'a/DT'. The corresponding POS trigrams are 'something/NN in/IN a/DT', 'in/IN a/DT middle/JJ' and 'a/DT middle/JJ price/NN'. Here the 'part/NN of/IN the/DT' sequence starting with part/NN has already matched with 'something'; therefore, this match is not considered again. However, the system gets a match for the other two trigrams - 'in/IN the/DT back/JJ' and 'the/DT back/JJ part/NN' where 'the/DT' has not matched with any word of the input sentence. To resolve the ambiguity we consider the parse tree of the input sentence. The parse tree of the input sentence is shown in Figure 2.

The trigram 'In/IN a/DT middle/JJ' has the lowest common ancestor at depth 4 whereas 'a/DT middle/JJ price/NN' has the lowest common ancestor at depth 5. We consider the trigram which has the lowest common ancestor at a higher depth (i.e., lower level). Therefore, in this case, the trigram 'a/DT middle/JJ price/NN' is considered and the corresponding matched sequence is 'the/DT back/JJ part/NN'. Subsequently the system looks for the translation of the 7th word 'the' of the TM suggestion. However, since there is no alignment corresponding to the 7th source word in the GIZA++ alignment, the translation of 'a' is not placed in the TM suggestion translation. Afterwards the system searches for 'middle/JJ'. The corresponding POS trigrams are 'in/IN a/DT middle/JJ', 'a/DT middle/JJ price/NN' and 'middle/JJ price/NN range/NN'. The first two trigrams match with 'in/IN the/DT back/JJ' and 'the/DT back/JJ part/NN' where 'back/JJ' does not match with any word of the input sentence. The third trigram does not match with any POS sequence in the TM suggestion. To resolve the ambiguity we need to consult the parse tree again. The POS trigram 'in/IN a/DT middle/JJ' has the lowest common ancestor at depth 4 while the POS trigram 'a/DT middle/JJ price/NN' has the lowest common ancestor at depth 5. Therefore, the latter trigram is considered and the corresponding word for 'middle/JJ' is 'back/JJ'. 'back/JJ' is located at position 8 of the TM suggestion and its translation is 'পিছনের' which is located at position 3 of the TM suggestion translation. Therefore the translation of 'middle/JJ', 'মাঝারি আকারের', is replaced by the word 'পিছনের' in the TM suggestion translation. Thus the modified translation is formed as follows.

আমি বিমানের মাঝারি আকারের একটা কিছু বসতে পছন্দ করব .

The system next searches for 'price/NN' whose translation to be used here is 'দাম'. The three POS trigrams to be considered are 'a/DT middle/JJ price/NN', 'middle/JJ price/NN range/NN', and 'price/NN range/NN ./.'. Here 'a/DT middle/JJ price/NN' gets a match with 'the/DT back/JJ part/NN', where 'part/NN' is the corresponding word for 'price/NN'. However, 'part/NN' has already been used earlier; therefore, the system ignores this match. The other two trigrams do not match with any POS trigram of the TM suggestion. Two POS bigrams considered for 'price/NN' are 'middle/JJ price/NN' and 'price/NN range/NN'. 'middle/JJ price/NN' matches with 'back/JJ part/NN'; however, it is ignored since the translation position of 'part/NN' has already

been replaced. The other bigram does not match either. Therefore the system falls back to the unigram match for 'price/NN'. It matches with 'part/NN' and 'plane/NN'. Since 'part/NN' has already been used, the system considers 'plane/NN' which is at position 12 of the TM suggestion and its translation, 'বিমানের', is at position 2 of the suggestion translation. Therefore, 'বিমানের' is replaced by 'দাম' and the suggestion translation is modified as given below.

আমি দাম মাঝারি আকারের একটা কিছু বসতে পছন্দ করব .

The system tries to find a match for 'range/NN' later on. However, its trigram, bigram, and unigram POS sequences are either being used already or do not match. Therefore, its translation is not put in the suggested translation. Finally the word 'বসতে' which is the translation of 'sit' is deleted since 'sit' does not match with any word of the input sentence. Thus, the final translation suggestion is produces as given below.

আমি দাম মাঝারি আকারের একটা কিছু পছন্দ করব .

Since the translations of 'a/DT' and 'range/NN' are not placed in the translation suggestion, their translations 'একটা' and `দেড়'শ এর মধ্যে বদলাতে থাকে', respectively, are added to a list and are shown to the post-editor as suggestions. The post-editor can directly use those translations without typing them and can put them in the proper place. In this way the system modifies the TM translation suggestion to generate more appropriate translation candidates. These translation candidates can be post-edited with less post-editing effort.

3.7. Length Based Pruning

The POS tag and parse tree based process of filling up of the translation of the unmatched word in the TM translation suggestion to make it more suitable for post-editing works well when the input sentence and the suggestion translations are of similar lengths. In case of the input sentence and suggestion sentence have completely different length, their parse tree will be completely different. In such cases looking for a POS sequence match involving the unmatched word in the suggestion sentence can give us wrong results, which will eventually lead to loss of fluency in the target translation. Therefore, we consider only those sentences for translation suggestion whose lengths are within a specified limit of the length of the input sentence. The pruning strategy deals with sentence length. TM retrieved sentences that are either too short or too long with respect to the input sentence are discarded. The system considers only those sentences whose lengths lie in a certain range which is calculated based on the length of the input sentence. Users can set the range as per his choice. This reduces the time to generate initial translation suggestion.

3.8. Re-ranking of TM Suggestions

After producing the translations corresponding to the TM suggestions, the first option chosen by the TM module might not remain the best translation option anymore. This is also evident from the experimental results obtained (cf. 'first' vs. 'best' in Table 2, Section 4.). This motivated us to perform re-ranking of the produced translation suggestions to bring the most suitable translation in the top suggestion. Re-ranking deals with various features including language model probability, length of the input sentence, length of source side TM suggestions, number of unmatched words for which translations are successfully inserted into the corresponding TM translation suggestion, and the original similarity score produced by the CATaLog system.

3.8.1. Similarity Score of CATaLog

CATaLog calculates similarity scores on the basis of TER alignment. Let us consider, match_reward=0.80, deletion_cost=0.20, insertion_cost=0.50, substitution_cost=0.70, and suppose TER alignment between an input segment and TM source segment is "MMDIMISMM". Therefore, the corresponding original TM match score (OTMS) is calculated as follows.

$$OTMS = 0.80 \times 5 - 0.20 - 0.50 \times 2 - 0.70 = 2.1$$

Now, let us consider that CATaLog_TS has successfully inserted the translation of two words represented as 'I' in the TER alignment. Therefore, two additional match_reward scores are added with $OTMS$ to arrive at the new TM match score (NTMS).

$$NTMS = OTMS + 2 \times 0.80 = 3.7$$

We estimate the fluency score of the translation suggestion on the target side using a language model and the estimated length of the actual translation of the input sentence. We used a 5-gram language model trained on the Bengali side of the TM corpus. We used SRILM toolkit (Stolcke, 2002) for language modelling with a back-off model. To generate the language model score of each translation suggestion we sum up all n-gram match logarithmic scores in a linear way. The resulting sum is a negative value (say, $-lm$). We take the absolute value of this score and normalize it with the length of the translation suggestion (TL), which gives us a normalized score, $P = lm/TL$. Then we take the inverse of it ($1/P$) to arrive at the corresponding LM score ($LMS = 1/P$) for this translation.
We also used the concept of brevity penalty to penalize a translation if its length is much smaller or longer than the estimated reference translation. Since no reference translation is available for the input sentence, we estimate the length of the translation of the input sentence on the basis of its length. Let length of the input sentence be SL and length of translation suggestion be TL. We assume that the reference translation length ($RefLen$) will be in the range between $0.8 \times SL$ and $1.2 \times SL$. If the candidate translation length lies

in that range, then we do not penalize it. However, when the length of the translation is out of that range, we assign it a length penalty based on Algorithm 1.

Algorithm 1 Length Based Penalty

procedure CALCULATE_PENALTY((SL, TL)
  $LP \leftarrow 0$
  $minRefLen \leftarrow 0.8 \times SL$
  $maxRefLen \leftarrow 1.2 \times SL$
  $diff \leftarrow 0$
  if $TL \geq minRefLen$ **AND** $TL \leq maxRefLen$ **then**
    $LP = 1.0$
    return LP
  end if
  if $TL \leq minRefLen$ **then**
    $diff = minRefLen - TL$
  end if
  if $TL \geq maxRefLen$ **then**
    $diff = TL - maxRefLen$
  end if
  $LP = e^{(\frac{-diff}{SL})}$
  return LP
end procedure

We calculate a fluency score (cf. Equation 1) using the language model score LMS and length based penalty (LP).

$$smoothness\_score = LMS \times LP \quad (1)$$

Finally, we re-rank the top suggestions based on the new score in Equation 2.

$$Final\_score = smoothness\_score \times NTMS \quad (2)$$

4. Experiments and Results

The effectiveness of the proposed system (CATaLog_TS) is demonstrated by comparing it against CATaLog (Nayek et al., 2015) and the Moses (Koehn et al., 2007) implementation of the PB-SMT model. We used an English–Bengali parallel corpus which contains 13,000 sentences. This parallel corpus serves as our TM on which we train the CATaLog and CATaLog_TS system. The baseline PB-SMT system is also trained on the same parallel corpus. For building the PB-SMT system, we set the maximum phrase length to 7 and a 5-gram language model is trained using KenLM (Heafield, 2011). Parameter tuning was carried out using Minimum Error Rate Training (MERT) (Och, 2003) on a held-out development set containing 500 sentences. Two different testsets were used for evaluation: **Testset1** contained 100 sentences and **Testset2** contained 500 sentences. We evaluated our system using two well known automatic MT evaluation metrics: BLEU (Papineni et al., 2002) and TER (Snover et al., 2006).

CATaLog_TS provides five translation suggestions based on the top five close matches retrieved from the TM by CATaLog. The term 'First' in Table 2 represents the first (i.e. the top ranked) translation suggestion provided by the CATaLog or CATaLog_TS system. The 'Best' translation suggestion among the five translation suggestions is chosen according to S-BLEU.

Table 2 shows that, as far as the 'First' translation suggestion is concerned, CATaLog_TS provides 2.13 and 2.03 BLEU points absolute (22.4% and 19.2% relative) improvements over CATaLog for testset1 and testset2 respectively. The respective improvements are 8.21 and 9.64 points (12.8% and 14.6% relative) for TER. Similarly, for the 'Best' translation suggestion, the improvements provided by CATaLog_TS over CATaLog for testset1 and testset2 are 3.59 and 1.91 BLEU points (29.8% and 14.5% relative) and 10.99 and 6.24 TER points (17.1% and 10.3% relative) respectively.

More importantly, for testset1, CATaLog_TS 'Best' performs better than the state-of-the-art PB-SMT system in both BLEU and TER. However, in case of testset2, CATaLog_TS 'Best' performs better according to TER while Moses fares better according to BLEU. This is probably due to the fact that the Moses system was tuned with the BLEU evaluation metric.

From Table 2, we can conclude that CATaLog_TS always performs better than CATaLog. The TER score for CATaLog_TS is much lower than CATaLog for both 'First' and 'Best' translation suggestions. BLEU scores also reflect the same trend. Comparison with Moses system reveals that CATaLog_TS provides the lowest TER scores for both the testsets, even if we just consider the 'first' translation suggestion. However, Moses is ahead on testset2 while CATaLog_TS fares better on testset1 according to BLEU.

Table 2 also shows that after re-ranking the top suggestions, CATaLog_TS_ReRank system provides much higher BLEU score and lower TER score compared to Moses for testset1. Although, in case of testset2, the BLEU score of the CATaLog_TS_ReRank system is better than the best option of the CATaLog_TS system, but lower than that of Moses. However, for either case (i.e, testset1 and testset2), the TER score of CATaLog_TS_ReRank system is considerably better than the other systems. It is to be noted that the CATaLog_TS 'Best' system was decided on the basis of S-BLEU score, while for the actual evaluation purpose we use BLEU. BLEU being a system level score does not perform well at sentence level evaluation; hence the BLEU and TER scores provided by CATaLog_TS_ReRank system are better than those provided by CATaLog_TS 'Best' system.

5. Conclusions and Future Work

The CATaLog tool is specifically targeted towards improving the user experience with TM. It does so by color coding the TM suggestions. In this paper we reported the introduction of another important functionality in TM, that of proposing a new translation. Traditionally, TMs do not generate any translation; so we present a step beyond traditional TM. Besides, this improves HCI issues with TM since this new functionality generates a new translation based on the translation template chosen by the user.

Testset	System		Performance	
			TER	BLEU
Set1	CATaLog	First	64.10	9.49
		Best	64.41	12.03
	Moses		57.12	14.57
	CATaLog_TS	First	55.89	11.62
		Best	53.42	15.62
	CATaLog_TS_ReRank		**48.49**	**18.07**
Set2	CATaLog	First	65.98	10.58
		Best	60.82	13.15
	Moses		58.44	**18.34**
	CATaLog_TS	First	56.34	12.61
		Best	54.58	15.06
	CATaLog_TS_ReRank		**53.83**	15.68

Table 2: Systematic comparison between CATaLog, CATaLog_TS, CATaLog_TS_ReRank and Moses

In future we will replace the existing bilingual dictionary with a probabilistic bilingual dictionary. We would like to conduct a user evaluation in real world experimental settings with human translators to measure productivity gain yielded by the tool. colorredWe would also like carry out an evaluation to compare our system against other CAT systems available.

Acknowledgements

We would like to thank the anonymous reviewers for their feedback. Santanu Pal is supported by the People Programme (Marie Curie Actions) of the European Union's Framework Programme (FP7/2007-2013) under REA grant agreement no 317471.

Bibliographical References

Biçici, E. and Dymetman, M. (2008). Dynamic translation memory: using statistical machine translation to improve translation memory fuzzy matches. In *Computational Linguistics and Intelligent Text Processing*, pages 454–465. Springer.

Brown, P. F., A. Della-Pietra, S., J. Della-Pietra, V., and L. Mercer, R. (1993). The Mathematics of Statistical Machine Translation. *Computational Linguistics*, pages 263–313.

Dandapat, S., Morrissey, S., Way, A., and Forcada, M. L. (2011). Using Example-Based MT to Support Statistical MT when Translating Homogeneous Data in a Resource-Poor Setting. In *Proceedings of the 15th Annual Conference of the European Association for Machine Translation*, pages 201–208. European Association for Machine Translation.

Heafield, K. (2011). KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197.

Koehn, P. and Senellart, J. (2010). Convergence of translation memory and statistical machine translation. In *Proceedings of AMTA Workshop on MT Research and the Translation Industry*, pages 21–31.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180.

Lagoudaki, E. (2008). The value of machine translation for the professional translator. In *Proceedings of the 8th Conference of the Association for Machine Translation in the Americas*, pages 262–269, Waikiki, Hawaii.

Ma, Y., He, Y., Way, A., and van Genabith, J. (2011). Consistent translation using discriminative learning - a translation memory-inspired approach. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1239–1248, June.

Nayek, T., Naskar, S. K., Pal, S., Zampieri, M., Vela, M., and van Genabith, J. (2015). Catalog: New approaches to tm and post editing interfaces. In *Proceedings of the NLP4TM Workshop*.

Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51.

Och, F. J. (2003). Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, pages 160–167.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Simard, M. and Isabelle, P. (2009). Phrase-based machine translation in a computer-assisted translation environment. *Proceedings of the Twelfth Machine Translation Summit (MT Summit XII)*, pages 120–127.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas*, AMTA.

Stolcke, A. (2002). Srilm - an extensible language modeling toolkit. pages 901–904.

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

Methodological pitfalls in automated translation memory evaluation

Friedel Wolff[1], Laurette Pretorius[1], Loïc Dugast[1], Paul Buitelaar[2]
[1]College of Graduate Studies
University of South Africa
[2]INSIGHT Center for Data Analytics
National University of Ireland, Galway
{wolfff,pretol,dugasl}@unisa.ac.za, paul.buitelaar@insight-centre.org

Abstract

A translation memory system attempts to retrieve useful suggestions from previous translations to assist a translator in a new translation task. While assisting the translator with a specific segment, some similarity metric is usually employed to select the best matches from previously translated segments to present to a translator. Automated methods for evaluating a translation memory system usually use reference translations and also use some similarity metric. Such evaluation methods might be expected to assist in choosing between competing systems. No single evaluation method has gained widespread use; additionally the similarity metric used in each of these methods are not standardised either. This paper investigates the choice of fuzzy threshold during evaluation, and the consequences of different choices of similarity metric in such an evaluation method. Important considerations for automated evaluation of translation memory systems are presented.

Keywords: translation memory, evaluation, text similarity

1. Introduction

Evaluation is important in natural language processing. Amongst others it is essential for measuring progress and to provide a way to distinguish between competing systems in a comparison (Mapelli et al., 2008).[1]

A translation memory (TM) is a database containing records of associated source and target text. It is common for records to store mutual translations of sentences, but other granularities are possible, such as paragraphs or terms. In its normal operation, a TM system is queried for a new source text to be translated which is compared to all source texts currently stored in the TM. If an identical source text has been translated before, it is a trivial lookup procedure to return this previous record to present it to the user. If this specific source text has not been translated before, we see the more interesting, general case where some kind of fuzzy matching is required. The record with the most similar source text is retrieved according to some measure of similarity between the new source text to be translated and the previously translated source text stored in the TM. The record with the previously translated source text and its associated target text is returned as a suggestion for the translator to consider. Translation memory systems are often user configurable with a similarity threshold f and only provide suggestions with a similarity higher than or equal to this threshold value (often expressed as a percentage such as 70% or a fraction 0.7), depending on the content of the translation memory. Therefore, unlike the usual case in machine translation systems, a translation memory system does not necessarily provide suggestion(s) for each input segment. Considering each translation memory suggestion causes cognitive effort for the translator (O'Brien, 2007), and the threshold is a mechanism to limit the number of unhelpful suggestions that are provided to the translator for consideration.

Evaluation in translation memory systems should ideally provide a way to evaluate TM systems and their operation, including any parameters, as well as to evaluate datasets, and to compare their relative performance. We might therefore look for a way of evaluating the choice of the similarity threshold f (or whatever mechanism is used to decide if any suggestion should be presented at all), and some measure that provides an estimation or proxy for the value of the suggestions to the translator. It should be noted that the kind of evaluation as discussed in this paper ignores matters of system performance such as requirements in run time or memory use—although these could also be important in their own right.

No existing method for the evaluation of TM systems has been widely accepted in scientific literature. Section 2 provides an overview of existing methods for evaluating translation memory systems. This paper does not attempt to do a meta-evaluation to choose the best amongst them, but discusses the underlying similarity metrics that are employed by them. These metrics are not unique to the field of TM systems—many of these are used in other fields such as machine translation, speech recognition and bio-informatics. An interesting aspect of automated TM evaluation is that similarity metrics are employed not only during the normal operation of the TM system (retrieval), but also often as part of automated evaluation methods. Section 3 discusses two classes of these similarity metrics, with implementation details in section 4. Section 5 contains our core contribution: we investigate two methodological pitfalls that affect the automatic evaluation of translation memory evaluation, namely the fuzzy threshold and the choice of similarity metric during evaluation. In section 6 we discuss and contextualise our results. Section 7 concludes while also suggesting some further areas for investigation.

2. Existing evaluation methods

Some previous work in evaluating translation memory systems are discussed in this section.

Manual evaluation with human judges is possible (Bloodgood and Strauss, 2014). In (Gow, 2003) an evaluation

[1]Also see http://hlt-evaluation.org

method was designed that allows a comparison between systems that use different approaches to search and retrieval, especially with regards to segmentation (or the absence thereof). Initial work on evaluation with eye-tracking equipment was reported in (O'Brien, 2007). Since these methods rely on human judges, they are not usable as automated methods for the evaluation of translation memories or translation memory systems.

Contrary to the abovementioned methods, our focus in this paper is on fully automated evaluation methods. Such methods have been invaluable in many fields, such as machine translation (Machácek and Bojar, 2014) where the annual workshop on statistical machine translation also includes a shared task devoted to evaluation metrics. Such automated methods are valuable since they allow repeated experiments without involving human judges in each experiment.

Next we present a summary of previous efforts in fully automatic evaluation of TM systems.

2.1. Methods based on information retrieval evaluation

Some evaluation methods for the evaluation of translation memory systems have taken their inspiration from the evaluation of information retrieval systems.

In (Whyman and Somers, 1999) the authors considered a black box evaluation method for a translation memory system—it only considers the input and output of the system, not the contents of the TM. Their method offers a way to determine a good value for the fuzzy match threshold f to set in a translation memory system. This method defines both a precision-like and recall-like metric, both of which are calculated using a keystrokes estimation as similarity metric. By repeatedly evaluating for these two measures over a wide range of values of f, it provides an opportunity to select an optimum value for f (even within a black box evaluation setup) by optimising for the required combination of precision and recall.

Another evaluation method was proposed in (Baldwin, 2009). Different similarity metrics were evaluated for use during retrieval. The method uses 10-fold cross-validation over the whole test set, each time exhaustively using all segments from one tenth of the dataset as queries against the remaining 90% which acts as the TM database. The top result returned is classified as a hit or a miss, depending on whether or not it has the most useful target language suggestion. It therefore uses full knowledge of the target segments in the translation memory to identify the most useful target suggestion, regardless of why it would be a useful suggestion. This is also a slight point of criticism, since suggestions could be useful for a number of reasons, including completely unexpected ones (Wolff et al., 2014)—sometimes an impossible feat for a retrieval method to model. This method employs two similarity metrics during evaluation to determine the most useful target language suggestion: the 3-operation edit similarity over bigrams (see section 3.2) and the WSC metric (Baldwin and Tanaka, 2000).

2.2. Correlation with reference metric

Recent work suggested a method to compare the performance of similarity metrics (Vanallemeersch and Vandeghinste, 2015). For each segment in the evaluation set it calculates two quantities: (1) the similarity of the source query to the *source* text of the suggestions returned as calculated by the retrieval metric, and (2) the evaluation score under a metric Sim_{TER} (based on TER (Snover et al., 2006)) for the *target* suggestion compared to the reference translation from the evaluation set. The correlation between these two quantities is used to score the performance of different similarity metrics. In that sense Sim_{TER} is used as a reference metric. In this evaluation, the effect of the similarity threshold is not considered.

This part of their evaluation method therefore evaluates the accuracy with which a similarity metric predicts the value of its suggestions, rather than trying to evaluate the inherent value of the suggestions themselves.

Additionally the authors investigated the effect on mean evaluation score with suggestions retrieved for a greater or lesser number of the test segments. Evaluating the mean evaluation score for N suggestions entails choosing the N segments with the best-scoring suggestions according to the similarity metric used for retrieval. This is similar to evaluating the effect of the similarity threshold (see section 5.1) in the sense that the number of segments in the test set with the best scoring suggestions is a proxy for the similarity threshold by which the system determines if a specific entry in the TM is suggested or not. From this investigation, the relative increase in mean evaluation score between similarity metrics is investigated as N is increased. Since the mean evaluation score is mostly monotonically increasing, it seems as if such an evaluation might propose an arbitrarily low similarity threshold to be able to propose as many suggestions as possible. It therefore seems as if this does not provide a meaningful way to choose a value for the similarity threshold f.

2.3. Methods based on machine translation evaluation

A translation memory system can be evaluated as a machine translation system (Simard and Fujita, 2012), using well-known methods such as BLEU, NIST, METEOR, and WER. Since these evaluation methods rely on the fact that an output is generated for each segment, such experiments have to be conducted without filtering with a similarity threshold. These methods for the evaluation of machine translation systems are therefore not considered further in this paper, as they can not evaluate the functioning of a TM system as it is used in practice—they were simply designed with another application in mind. However, some findings of this paper are relevant to our investigation here, and we will return to it in our discussion in section 6.

Although some of the machine translation evaluation methods operate on the level of a whole test set, several of them can be viewed as similarity metrics in themselves (as was done with TER in (Vanallemeersch and Vandeghinste, 2015)), analogous to the metrics discussed in the next section.

3. Similarity metrics

When investigating translation memory systems, we see similarity metrics used in two roles: (1) While retrieving

suggestions for a translator, a TM system uses a similarity metric to compare source texts. (2) As seen above, evaluation methods use similarity metrics on the suggestions provided in the target language.

String similarity metrics are often based on some kind of edit distance, although other formulations are also possible. A *distance function* (such as the well-known Levenshtein distance) measures dissimilarity between two strings. Identical strings have a distance of 0. The less similar two strings are, the greater the distance between them. *Similarity metrics* provide a similarity score, usually presented as a fraction between 0.0 and 1.0, or as a percentage between 0% and 100%. Identical strings have a similarity of 1.0 or 100%. To convert a distance d into a similarity score, some *normalisation constant*, say l is used to constrain the distance to the range $[0, 1]$. The similarity score is then defined as $1 - d/l$. To ensure that the minimum value of the similarity score is 0, the value of l has to be the maximum possible value of d. Using such a normalised metric allows comparison of the extent of similarity between different string pairs, regardless of whether they are short or long. Without such normalisation, a distance of 6 could indicate two very similar strings, or two very different strings, depending on their lengths. During TM evaluation, the similarity metrics are used in aggregate form (say as an average over a test collection). In such cases, an unnormalised score could cause the scores on long strings to overshadow the overall result, possibly skewing the results and complicating interpretation thereof. Because the similarity metrics and their normalisation work differently, there could be a difference between the similarity scores for a string pair as measured between two different similarity metrics. The different similarity metrics therefore not only affect the ranking of suggestions, but also whether any suggestions are presented at all when constrained with a particular similarity threshold.

This section describes several metrics operating on a pair of symbol strings. It is easiest to consider the character as the lowest atom for consideration. Another possibility is to operate on word or token level. Another further possibility is n-grams of characters or indeed of words. These possibilities were considered by (Baldwin, 2009): unigram, bigram and mixed unigram/bigram combinations of both character and word based indexing. We perform matching on both character and word level in the later sections of this paper.

Many other similarity metrics could be discussed. The limited selection below is chosen based on performance in previous literature and existing software.

In the descriptions of the similarity metrics below, the following notation will be used: A and B refer to the two strings being compared (the query and a candidate string, respectively), $|A|$ indicates the length of the string A in terms of the granularity considered (e.g. character length when working on character level).

3.1. 4-operation edit similarity

This metric is based on possibly the best known string distance function, usually called the Levenshtein distance (Levenshtein, 1966). The Levenshtein distance is the number of insertions, deletions and substitutions required to transform the one string into the other. It is also called "4-operation edit distance" with the understanding that the fourth operation is the identity operation (no change).[2]

The maximum distance between two strings is the length of the longest string, and occurs in the case of a comparison with the empty string, or when the two strings have no characters in common. The similarity metric is therefore defined as

$$sim_{4ops}(A, B) = 1 - \frac{edit_{4ops}(A, B)}{max(|A|, |B|)}$$

Although the implementation details of proprietary tools for computer assisted translation is hidden, an informal investigation of some Free and Open Source Software indicates that this metric is used by some TM system implementations: OmegaT,[3] Okapi Framework,[4] Virtaal[5] and the Amagama TM server[6]. This suggests that this metric enjoys some respect as a similarity metric in existing tools.

Here are a few examples operating over different granularities:

- characters: $sim_{4ops}(\textit{"metaphor"}, \textit{"metamorphosis"}) = 1 - \frac{6}{13} = 0.538$
- words: $sim_{4ops}(\textit{"metaphor"}, \textit{"metamorphosis"}) = 1 - \frac{1}{1} = 0$
- characters: $sim_{4ops}(\textit{"A number"}, \textit{"A number of tests"}) = 1 - \frac{9}{17} = 0.471$
- words: $sim_{4ops}(\textit{"A number"}, \textit{"A number of tests"}) = 1 - \frac{2}{4} = 0.5$

3.2. 3-operation edit similarity

This metric is similar to the **4-operation edit similarity**, but the underlying distance function does not consider substitution as one of the basic operations. A substitution is therefore modelled as an insertion and a deletion (two operations). The identity operation is the third operation. It is also known as the *indel* distance (*in*sert and *del*ete distance) or the *longest common subsequence distance*.

The maximum distance between two strings is the sum of the lengths of the two strings, and occurs when the two strings have no characters in common. The similarity metric is therefore defined as

$$sim_{3ops}(A, B) = 1 - \frac{edit_{3ops}(A, B)}{|A| + |B|}$$

In the method by Baldwin (Baldwin, 2009) this metric was often the best performing one in his evaluation.

Here are a few examples operating over different granularities:

- characters: $sim_{3ops}(\textit{"metaphor"}, \textit{"metamorphosis"}) = 1 - \frac{7}{21} = 0.667$

[2] This definition assumes unit cost, in other words a distance of 1 for each of the operations except identity. Different weights can be assigned to different operations, but is not considered further in this paper.

[3] http://omegat.org/

[4] http://okapi.opentag.com/

[5] http://virtaal.translatehouse.org/

[6] http://amagama.translatehouse.org/

- words: $sim_{3ops}(\text{“metaphor”}, \text{“metamorphosis”}) = 1 - \frac{2}{2} = 0$
- characters: $sim_{3ops}(\text{“A number”}, \text{“A number of tests”}) = 1 - \frac{9}{25} = 0.64$
- words: $sim_{3ops}(\text{“A number”}, \text{“A number of tests”}) = 1 - \frac{2}{6} = 0.667$

4. Implementation details

In this section we provide details on several implementation aspects that would be required for duplicating our results.

For all the word-based metrics we use the BreakIterator from the ICU project[7] for tokenisation.

Our experiments were conducted using the Open Source Amagama TM server.[8] One particular implementation detail that is relevant, is the way in which candidate suggestions are retrieved from the database: a full-text index over the source segments is kept that allows for selecting an initial candidate list with at least *some* minimal token overlap. Tokens are lowercased, stemmed words with no stopwords removed. This is a very loose, initial filter that still allows candidates with very low similarities to be subjected to full consideration by the similarity metric in use. The similarity scores given to each candidate is completely determined by the retrieval metric in the experiment—the full-text engine has no influence in this regard since it is not used again after the initial filtering. The use of a full-text index is merely an optimisation to eliminate the big part of the TM for consideration within each query, similar in spirit to the *approximate query coverage* (AQC) (Vanallemeersch and Vandeghinste, 2015).

5. Experimental setup

In trying to point out methodological pitfalls, we configure a TM evaluation setup based mostly on the method of Whyman and Somers (Whyman and Somers, 1999), while substituting their keystroke estimation metric with another in each case. Since this method provides both a precision-like and recall-like measure, it allows us to investigate the effect on both measures, as well as combining it through the F_1-score. The method was further enhanced with 10-fold cross-validation as done in (Baldwin, 2009). Since true stratification is not possible in this kind of dataset, we instead ensure that the distribution of source text lengths over the 10 folds are consistent by checking that the average length in each fold is close to the average of the whole dataset. This serves as “semi-stratification” in the same spirit as (Baldwin, 2009).

Within each dataset, each part of the experiment is specified with a pair of similarity metrics: one used for retrieval, and one used for evaluation (after results are obtained). The experiment therefore performs over the Cartesian product of the following similarity metrics (with shorthand indicated in brackets):

- 4-operation edit similarity over characters and words (edit4, edit4word)
- 3-operation edit similarity over characters and words (edit3, edit3word)

When using a metric M during evaluation, we say that the results are evaluated “under” the similarity metric M.

Two datasets in two linguistically unrelated languages were used: The 2004_1 subset of the DGT-TM Release 2011 (Steinberger et al., 2012) and the GNOME 3.8 user interface translations,[9] specifically for the language pairs English–French (`en-fr`) and English–Hungarian (`en-hu`).

Whereas the DGT corpus contains legislative texts, the GNOME corpus contains user interface translations of the GNOME desktop environment version 3.8. The GNOME texts are also different in that it can contain XML markup, placeholders and more non-textual elements as is often found in software localisation texts. See table 1 for an overview of the corpus statistics. The standard deviations for the average character lengths indicate the deviation of the averages over the 10 folds, not the standard deviation on segment level. This indicates the very small variance in average segment length between the 10 folds, and serves to show that our “semi-stratification” as mentioned above is reasonably fair.[10]

Corpus	Unique segments	Avg. characters
DGT (fr)	71033	126.75 ± 1.225
DGT (hu)	45964	125.61 ± 1.186
GNOME (fr)	36493	27.51 ± 0.357
GNOME (hu)	36008	27.65 ± 0.431

Table 1: Corpus statistics

For each of the folds in the cross-validation, we try to retrieve a TM suggestion for each of the source text segments in the testing data from the remaining (“training”) data. Only the highest-ranking suggestion according to the similarity metric used for retrieval is considered (ties are broken randomly). A result set for each similarity metric is created this way and submitted to evaluation. One difference in our evaluation method here is that we compare the *target text* of the suggestions with the reference translations from the testing data, whereas the original method compared source texts. Indeed, the authors remark (Whyman and Somers, 1999):

> For this measure to be strictly accurate, the keystroke count should of course be carried out on the target-language text segments associated with the matches, as compared to the desired target translation.

Their own concern for not doing so is based on a concern for subjectivity in choosing a reference translation. However, since their research was published, the use of such reference translations has become commonly accepted in the machine translation community, and we therefore proceed in using

[7] http://site.icu-project.org/

[8] http://amagama.translatehouse.org/

[9] Available from https://l10n.gnome.org/releases/gnome-3-8/

[10] Although a random division into 10 folds will produce different partitions on each occasion, statistics similar to the ones reported are always obtained.

it as such. We believe that the 10-fold cross-validation and bigger datasets should eliminate major concerns of this nature. It is also worth noting that using the source texts would simply duplicate the calculation done while retrieving suggestions, and would therefore provide no way of evaluating the intended function of the TM system—to retrieve a target text that assists the translator in the current translation task.

In the evaluation of machine translation experiments, multiple reference translations are recommended. It is a luxury we do not have in this case.

A central aspect of the evaluation is to define a measure of "usefulness" for each suggestion, according to the similarity metric used during evaluation. This allows for the computation of a weighted "hit value" over all suggestions (defined below), rather than merely counting "hits" and "misses" among the suggestions.

During evaluation we calculate the weighted precision P_f^w and the weighted true efficiency F_f^w as defined in (Whyman and Somers, 1999), for each of the similarity metrics in the evaluation method. These two measures represent precision-like and recall-like quantities respectively. They are calculated as follows:

$$P_f^w = \frac{h_f^w}{m_f}$$
$$F_f^w = \frac{h_f^w}{n}$$

where n is the number of segments evaluated, m_f is the number of these for which a suggestion (match) is retrieved at similarity threshold f, and h_f^w is the weighted "hit value" at similarity threshold f. The weighting is very similar in form to the generalised similarity score discussed in section 3, that is $1 - d/l$. The difference is that whereas the similarity metrics have a range of $[0, 1]$, this weighting for evaluation has a range of $[-1, 1]$, which gives rise to an important property namely that an unhelpful suggestion (with very low similarity to the reference translation) receives a penalty and therefore contributes a negative score to h_f^w rather than merely a small positive score.

A generalisation of the weighting scheme in (Whyman and Somers, 1999) is presented as follows:

$$\hat{W}_l = 2 \times sim(A, B) - 1$$

with $sim(A, B)$ the similarity metric in use in each case. In the original paper, only a keystrokes estimation was used. h_f^w is then the sum of all these $\hat{W}_l$ weights measured over individual string pairs. The $n - m_f$ segments with no suggestion at f are weighted with neutral "usefulness", i.e. 0—they did not contribute any value to the translator nor demanded any cognitive effort to consider.

This evaluation method allows many types of investigation, since it models the effect of the similarity threshold f, and the variables P_f^w and F_f^w can be studied independently. Since this is not the main focus of this paper, we consider the balanced F_1-score a sufficient instrument to indicate the methodological pitfall we will present. For certain applications, other evaluation outcomes might be more important and could be used instead.

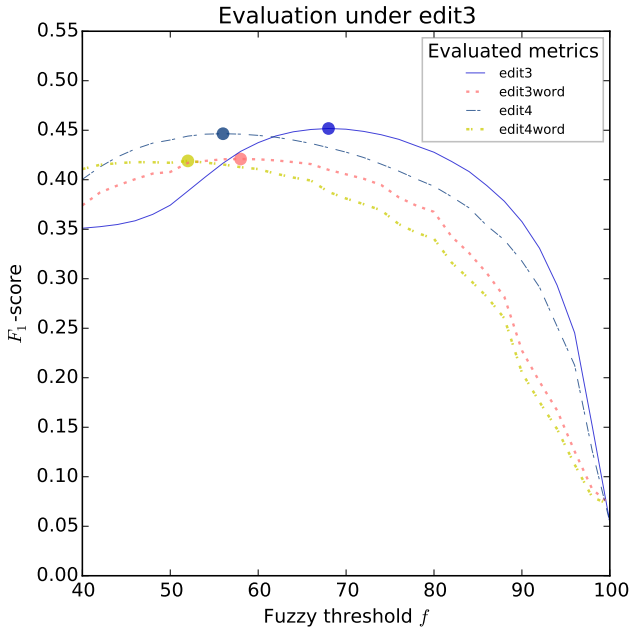

Figure 1: Evaluating all retrieval metrics on the DGT `en-fr` test set under the **edit3** metric.

5.1. The influence of the similarity threshold

It is tempting to perform each of the experiments with a fixed similarity threshold to simplify the experiment. Although no specific value is an obvious choice, a choice such as 70% would probably not raise eyebrows, as this is the default value in some tools for computer assisted translation. Such a threshold would appear to simulate the behaviour of such a translation tool. However, in this section we investigate the effect of the similarity threshold, and see that it has to be investigated as a variable in its own right, since the TM performance of similarity metrics are highly dependent on the value of the similarity threshold.

To fully see the effect of changes in the similarity threshold, we filter retrieved suggestions with a very low similarity threshold (say 40%), but iteratively perform the evaluation from that value increasing it in increments of 2%, each time eliminating some suggestions that do not meet the threshold any more and recalculating our evaluation scores. This allows us to inspect the change in evaluation outcomes over a large range of the similarity threshold as if many individual experiments were performed. This could be used to graph precision against recall to providing additional insights into the relationship between precision, recall and the similarity threshold (not shown here).

In figures 1 and 2 we see the F_1-scores graphed against the similarity threshold f. As expected, with a very high threshold ($f > 90\%$) the F_1-score is much lower for all metrics which reflects the fact that very few suggestions are provided that fulfil such a strict requirement for similarity. This represents a high-precision, low-recall configuration for the TM system. As the graph is considered leftwards towards lower values of f, we see how the performance of different metrics change in relation to changes in f.

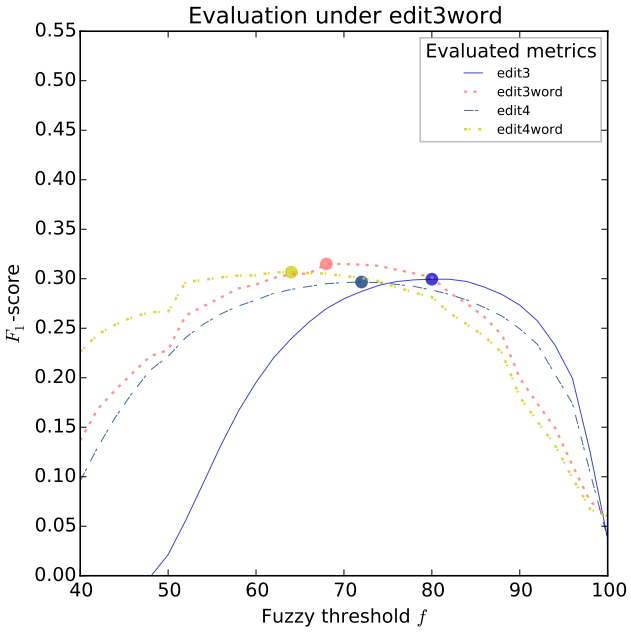

Figure 2: Evaluating all retrieval metrics on the DGT `en-fr` test set under the **edit3word** metric.

We can see the F_1-score of 3-operation edit similarity operating over characters (**edit3**) reaching its optimum at higher values of f. On the other hand **edit4word**'s optimum is usually at much lower values of f than the other metrics. This happens consistently and regardless of whether or not the metric is best performing in that part of the experiment.

Since each similarity metric reaches its optimum at a different value of f (sometimes far apart), it is important to note that choosing any single fixed value for f for comparing all the similarity metrics would not compare each metric at its strongest point. If choosing a different (fixed) similarity threshold for an evaluation experiment could alter the outcome of an evaluation, it would invalidate the experimental results. Therefore we need to take that into account for the design of evaluation methods. However, it remains important to evaluate the system with a fixed value of f tuned for each metric, since this is the way that TM systems work in practice. The performance of a system at values of f far away from the optimal configuration is inconsequential to a system that can be tuned to perform in an optimal configuration (in as much as the tuning is relevant to the task). The shape of the graph should therefore be irrelevant.

We see therefore that it is crucial to optimise the desired score (F_1-score in our case) separately for every similarity metric over the range of the variable f by tuning the variable f on a separate dataset before evaluation proper. In each of the 10 folds of the final experiment, we use 80% of the dataset as TM database, 10% for tuning the variable f, and the remaining 10% for calculating the evaluation score. The average of these evaluation scores from each of the 10 folds is the overall score reported in the next section.

5.2. Results

In all cases considered here we found the F_1-score for each metric reaching a clear maximum within the given range of f—the point where the similarity threshold f would be optimal for the relevant retrieval metric under the evaluation metric employed. We proceed by considering only this optimum value for each metric. In the two evaluation sets of the DGT corpus, each similarity metric selects itself as the best performing metric.

In the case of the GNOME corpus, we find a similar situation—in the `en-fr` dataset each of the metrics selects itself as winner. In the `en-hu` dataset, two of the four metrics (**edit3word** and **edit4**) do not select themselves, but the F_1-score for these three metrics are very close to the winning metric. In the light of the consistency of the bias in the corpora, as well as the very small difference between the cases where the bias was not outright, we assume that the four cases mentioned do not invalidate our results. The full results for the four datasets are shown in tables 2–5. Rows correspond to the output of the TM system as retrieved with each of the similarity metrics, and each column indicates the evaluation using a single similarity metric. Cases where a metric did not select itself are indicated in italics. The best performing metric in a column is indicated in bold, and a bold entry on the diagonal indicates a metric selecting itself.

	Evaluation metrics			
Retrieval metrics	*edit3*	*edit3word*	*edit4*	*edit4word*
edit3	**0.400**	0.258	0.312	0.213
edit3word	0.374	**0.282**	0.290	0.229
edit4	0.397	0.258	**0.315**	0.216
edit4word	0.374	0.281	0.293	**0.237**

Table 2: Evaluation results: DGT (`en-hu`)

	Evaluation metrics			
Retrieval metrics	*edit3*	*edit3word*	*edit4*	*edit4word*
edit3	**0.449**	0.324	0.380	0.290
edit3word	0.419	**0.349**	0.351	0.307
edit4	0.445	0.322	**0.383**	0.291
edit4word	0.418	0.343	0.354	**0.310**

Table 3: Evaluation results: DGT (`en-fr`)

Apart from the self-selection bias, while evaluating under one of the character-based metrics there is usually a difference in performance between the word-based metrics and the character-based metrics. It is interesting to see this difference being less obvious when evaluating under one of the word-based metrics. It seems as if there is at least some bias between the word-based metrics and the character-based metrics.

	Evaluation metrics			
Retrieval metrics	*edit3*	*edit3word*	*edit4*	*edit4word*
edit3	**0.332**	0.061	**0.236**	0.035
edit3word	0.239	*0.142*	0.145	0.102
edit4	0.322	0.057	*0.234*	0.040
edit4word	0.243	**0.144**	0.157	**0.119**

Table 4: Evaluation results: GNOME (`en-hu`)

	Evaluation metrics			
Retrieval metrics	*edit3*	*edit3word*	*edit4*	*edit4word*
edit3	**0.372**	0.084	0.262	0.048
edit3word	0.274	**0.172**	0.169	0.120
edit4	0.367	0.085	**0.266**	0.054
edit4word	0.283	0.171	0.189	**0.132**

Table 5: Evaluation results: GNOME (`en-fr`)

6. Discussion

Automated evaluation applied through k-fold cross-validation is supposed to eliminate some sources of bias in the test data. We also performed our investigation over two very different datasets, and repeated it for language pairs with two linguistically unrelated target languages. Although there is some variance in the results between the datasets, in all four datasets we see the same result almost without failure: that each similarity metric has a bias for itself. In the 2 out of 16 cases where the bias was not outright, the difference with the winning metric was very small.

The specific values of the F_1-scores obtained seem rather meaningless on their own, since the range of values for the same system output vary substantially depending on the similarity metric used during evaluation. It is therefore not meaningful to read anything in particular into the specific values, especially when comparing results obtained under evaluation experiments using different similarity metrics. Although we used the balanced F_1-score in all results discussed above, we saw the same bias when using F_β-scores with $\beta = 0.5$ (favouring weighted precision) and $\beta = 2$ (favouring weighted true efficiency).

(Baldwin, 2009) mentioned a concern for such a bias briefly, and opted to use the average accuracy score from separate evaluations with two similarity metrics (variations on edit3 over bigrams and WSC) in an attempt to remove bias. It is not clear if there is any scientific basis for believing that such an average is substantially less biased, or merely contains the bias of the constituents of the average in equal measure. Keeping the results from our study here in mind, the good performance of character-based metrics over word-based metrics in that study raises questions about whether it was caused by an evaluation method that employed two character-based similarity metrics.

The results of this paper confirm the suspicions of the authors in a study using an evaluation metric based on TER that found the metrics based on TER often outperforming others (Vanallemeersch and Vandeghinste, 2015).

A similar bias was observed when using machine translation evaluation metrics both for retrieval and evaluation in a TM system (Simard and Fujita, 2012), although such an experiment would not be comparable to our work here (see section 2.3). A related issue was observed when optimising for different metrics in a machine translation system during minimum error-rate training (MERT) (Servan and Schwenk, 2011). While their finding is arguably intuitive, as the tuning directly affects the evaluated output, a surprising result in our case is that the bias of the method used during retrieval is transferred over the language boundary from the source language to the target language where evaluation is performed.

7. Conclusion

We proposed an automated TM evaluation method based on the strong points of two previous approaches (Whyman and Somers, 1999; Baldwin, 2009). This provides both a precision-like and recall-like metric, a way of penalising less useful suggestions, and k-fold cross-validation reduces problems due to overfitting and local artefacts in the data.

We showed that it is important to model the similarity threshold as part of the evaluation experiment to mimic the way tools for computer aided translation work. We found that it is crucial to not use a fixed similarity threshold over the whole experiment, since this could favour certain metrics over others—each metric should be evaluated at the point where its performance is maximised.

Within our evaluation method, we used several similarity metrics for retrieving suggestions, and used the same similarity metrics again during evaluation. We considered all possible combinations of a set of four similarity metrics. We considered it over two very different datasets, in two language pairs. Experiments were conducted at character and word level. The limited selection of metrics investigated in this paper should be extended in future work. Amongst others, non-commutative metrics were not considered here (where $sim(A, B)$ is not necessarily equal to $sim(B, A)$).

We exposed the methodological pitfall of TM evaluation methods employing string similarity metrics to evaluate the performance of the same string similarity metrics as techniques for matching and ranking TM suggestions during retrieval.

Each similarity metric shows a bias during evaluation for itself. The findings in this paper suggest that no similarity metric is inherently more fair as part of an evaluation method without further evidence.

8. Acknowledgement

This research was supported in part by funding from the Science Foundation Ireland under Grant Number SFI/12/RC/2289 (Insight) and the Academy of African Languages and Science Strategic Project of the University of South Africa.

9. Bibliographical References

Baldwin, T. and Tanaka, H. (2000). The effects of word order and segmentation on translation retrieval performance. In *Proceedings of the 18th Conference on Computational Linguistics - Volume 1*, COLING '00, pages 35–41, Stroudsburg, PA, USA. Association for Computational Linguistics.

Baldwin, T. (2009). The hare and the tortoise: speed and accuracy in translation retrieval. *Machine Translation*, 23:195–240.

Bloodgood, M. and Strauss, B. (2014). Translation memory retrieval methods. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 202–210, Gothenburg, Sweden, April. Association for Computational Linguistics.

Gow, F. (2003). *Metrics for Evaluating Translation Memory Software*. Ph.D. thesis, University of Ottawa.

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

Machácek, M. and Bojar, O., (2014). *Proceedings of the Ninth Workshop on Statistical Machine Translation*, chapter Results of the WMT14 Metrics Shared Task, pages 293–301. Association for Computational Linguistics.

Mapelli, V., Arranz, V., Mazo, H., and Choukri, K. (2008). Latest developments in ELRA's services. In Nicoletta Calzolari (Conference Chair), et al., editors, *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2008/.

O'Brien, S. (2007). Eye-tracking and translation memory matches. *Perspectives: Studies in translatology*, 14(3):185–205.

Servan, C. and Schwenk, H. (2011). Optimising multiple metrics with MERT. *The Prague Bulletin of Mathematical Linguistics (PBML)*.

Simard, M. and Fujita, A. (2012). A poor man's translation memory using machine translation evaluation metrics. In *Proceedings of the Tenth Conference of the Association for Machine Translation in the Americas*.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231. Citeseer.

Steinberger, R., Eisele, A., Klocek, S., Pilos, S., and Schlüter, P. (2012). DGT-TM: A freely available translation memory in 22 languages. In Nicoletta Calzolari (Conference Chair), et al., editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, May. European Language Resources Association (ELRA).

Vanallemeersch, T. and Vandeghinste, V. (2015). Assessing linguistically aware fuzzy matching in translation memories. In *Proceedings of the 18th Annual Conference of the European Association for Machine Translation*, EAMT, Antalya, Turkey.

Whyman, E. and Somers, H. (1999). Evaluation metrics for a translation memory system. *Software-Practice and Experience*, 29(14):1265–84.

Wolff, F., Pretorius, L., and Buitelaar, P. (2014). Missed opportunities in translation memory matching. In Nicoletta Calzolari (Conference Chair), et al., editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, may. European Language Resources Association (ELRA).