PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 98 OCTOBER 2012 121–131

DELiC4MT: A Tool for Diagnostic MT Evaluation over User-defined Linguistic Phenomena

Antonio Toral, Sudip Kumar Naskar, Federico Gaspari, Declan Groves

School of Computing, Dublin City University

Abstract

This paper demonstrates DELiC4MT, a piece of software that allows the user to perform diagnostic evaluation of machine translation systems over linguistic checkpoints, i.e., source-language lexical elements and grammatical constructions specified by the user. Our integrated tool builds upon best practices, software components and formats developed under different projects and initiatives, focusing on enabling easy adaptation to any language pair and linguistic phenomenon. We provide a description of the different modules that make up the tool, introduce a web demo and present a step-by-step case study of how it can be applied to a specific language pair and linguistic phenomenon.

1. Introduction

DELiC4MT[1][2] is an open-source tool for diagnostic evaluation of Machine Translation (MT) which has been developed as part of the FP7 CoSyne project.[3] In contrast to automatic MT evaluation metrics, which are only effective at carrying out overall evaluations of MT systems (either at sentence or document level), this tool allows the evaluation of MT systems over linguistic phenomena specified by the user.

Most of the software tools for MT evaluation developed during the last decade belong to the category of automatic metrics. These are programs that, given the

[1] http://www.computing.dcu.ie/~atoral/delic4mt/

[2] https://github.com/antot/DELiC4MT

[3] http://cosyne.eu

 Corresponding author: atoral@computing.dcu.ie
Cite as: Antonio Toral, Sudip Kumar Naskar, Federico Gaspari, Declan Groves. DELiC4MT: A Tool for Diagnostic MT Evaluation over User-defined Linguistic Phenomena. The Prague Bulletin of Mathematical Linguistics No. 98, 2012, pp. 121–131. doi: 10.2478/v10108-012-0014-9.

output of an MT system and reference translation(s), apply different (primarily *n*-gram based) algorithms that provide a score by comparing the output of the system with the reference(s). Such automatic metrics include BLEU (Papineni et al., 2002), TER (Snover et al., 2006), METEOR (Banerjee and Lavie, 2005), etc. Asiya (Giménez and Màrquez, 2010) is a toolkit that provides a common interface to a rich set of metrics, thus making it easier for MT users to make use of these metrics. Asiya also incorporates schemes for metric combination.

Woodpecker (Zhou et al., 2008) is a piece of software that can evaluate MT systems over specific linguistic phenomena, also known as linguistic checkpoints (linguistically-motivated units e.g. an ambiguous word, a noun phrase, etc.), providing more fine-grained linguistically-motivated evaluation than the aforementioned 'traditional' metrics. The tool presented in the current paper builds on the paradigm introduced by Woodpecker and overcomes two of its limitations: (i) its implementation is language-independent (while Woodpecker had language-dependent data for English–Chinese hardcoded), and (ii) the license of the tool presented here allows anyone to work on it and release modifications, while conversely, Woodpecker's license, MSR-LA,[4] is quite restrictive in this regard. Moreover, the current tool provides additional functionalities such as checkpoint filtering based on PoS tags and statistical significance testing.

Evaluations of different MT systems for a range of linguistic checkpoints have been carried out for English–Chinese (Zhou et al., 2008), Italian–English, German–English and Dutch–English (Naskar et al., 2011).

The rest of the paper is structured as follows. Section 2 introduces the software architecture of the tool. This is followed by an illustrative case study in which the software is applied to a specific language pair and linguistic checkpoint. Finally we draw some conclusions and outline directions for future work.

2. Architecture

The aim of DELiC4MT is to provide the required functionality to perform diagnostic evaluation on a set of linguistic checkpoints. This is done by extracting checkpoint instances from text using PoS tagging (applied only to the source and reference translations) and word alignment and then evaluating these instances. The main focus during its development has been to allow for easy adaptation to any language pair and linguistic phenomenon. In fact, it has been applied successfully to evaluate a set of MT systems over a set of language directions involving German, Italian, Dutch and English, where the set of checkpoints is different for each language (Naskar et al., 2011). The work presented in this paper extends the work previously described in (Naskar et al., 2011) by incorporating a length-based penalty to penalize longer candidate translations as in (Zhou et al., 2008) and filtering of noisy checkpoint instances.

[4] https://research.microsoft.com/en-us/projects/pex/msr-la.txt

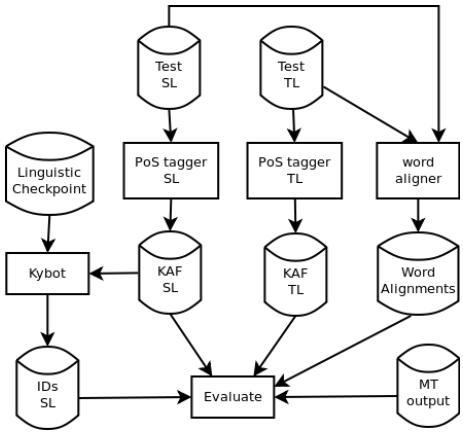

Figure 1. Architecture for linguistic checkpoints-based diagnostic evaluation of machine translation

This paper also provides additional technical details regarding the architecture and implementation of the tool.

The tool makes use of open-source software components and representation standards developed by the research community in recent years. It uses:

- State-of-the-art PoS taggers and word aligners. Treetagger[5] and GIZA++ (Och and Ney, 2003),[6] respectively, are used in the current version, although any similar tool could be used.
- KAF (Bosma et al., 2009), established in the FP7 KYOTO project,[7] for representing textual analysis. KAF is a unique format for representing all the levels of linguistic analysis based on ISO standards for each of those levels (i.e., MAF for morphology, SynAF for syntax and SemAF for semantics). Scripts to convert the output of several state-of-the-art tools are available (e.g. TreeTagger).
- Kybots (Vossen et al., 2010),[8] also developed within KYOTO, to define the linguistic phenomena to be evaluated. A Kybot profile can be thought of as a regular expression over elements and attributes in KAF documents.

Figure 1 presents the architecture of the tool. The source- and target-language sides of the gold standard (test set) are processed by PoS taggers and converted into KAF. The test set is also word aligned, and the identifiers of the aligned tokens are

[5] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/

[6] https://code.google.com/p/giza-pp/

[7] http://www.kyoto-project.eu/

[8] https://kyoto.let.vu.nl/svn/kyoto/trunk/modules/mining_module/

stored. Kybot profiles covering the different evaluation targets (linguistic checkpoints) are run on the source KAF text, and the identifiers of the matched terms are stored. Finally, the evaluation module takes as input the identifiers from the Kybot output, the KAF annotated test sets, the word alignments and the output of the MT system,[9] and calculates the performance of the MT system over the linguistic checkpoint(s) considered.

3. Case study

This section takes a closer look at the different modules of DELiC4MT by presenting a case study over a specific language pair and linguistic checkpoint. The case study demonstrates the use of the tool step-by-step.[10] A broader evaluation over a set of language pairs and checkpoints can be found in (Naskar et al., 2011). A web interface for the tool has also been developed, screenshots of which can be found at `http://www.computing.dcu.ie/~atoral/delic4mt/webdemo`. All of the scripts for running DELiC4MT are packaged within a single wrapper script (`delic4mt.sh`), facilitating ease-of-use.

3.1. Preparing the test data to be evaluated

We use the Italian–English test data of (Toral et al., 2011), consisting of source file *en-it.it* and target *en-it.en*, for illustration purposes.

The test set is PoS tagged using TreeTagger which also performs sentence-splitting and tokenisation. Since the tokens and sentences need to correspond to those in the alignment, we needed to alter TreeTagger's behaviour. A script has been developed for that reason (`treetagger_preserving_tokens_and_lines.pl`); it receives as input the text tokenised and applies TreeTagger to each sentence. The output of TreeTagger is post-processed overwriting any end-of-sentence (`SENT`) PoS tag by `OTHER`. Finally the tag of the last token of the sentence is overwritten to `SENT`. The output of this procedure is processed by a script that converts it to KAF (`treetagger2kaf.pl`). The following pipeline PoS tags the test set:

```
cat en-it.it | tokenizer.perl | treetagger_preserving_tokens_and_lines.pl \
italian | treetagger2kaf.pl -ri > en-it.it.kaf

cat en-it.en | tokenizer.perl | treetagger_preserving_tokens_and_lines.pl \
english | treetagger2kaf.pl -ri > en-it.en.kaf
```

The following is a sample of the KAF files produced for the Italian–English sentence pair 62 ("[...] la carne americana [...]", "[...] American meat [...]") [11]:

[9] The MT output is taken as is; no processing is required as for the test set.

[10] A more technical tutorial is included with the software. The tool also provides a wrapper that encapsulates all the functionalities in a single command.

[11] "carne" is the Italian word for "meat", and "americana" is the translation of "American" (inflected for feminine singular, to agree grammatically with "carne"). Note that in Italian the attributive adjective

```
<text>[...]
 <wf wid="w62_4" sent="62">la</wf>
 <wf wid="w62_5" sent="62">carne</wf>
 <wf wid="w62_6" sent="62">americana</wf>
[...]</text>
<terms>[...]
 <term tid="t62_5" type="open" lemma="carne" pos="NOM">
  <span><target id="w62_5"/></span>
 </term>
 <term tid="t62_6" type="open" lemma="americano" pos="ADJ">
  <span><target id="w62_6"/></span>
 </term>
[...]</terms>

<text>[...]
 <wf wid="w62_3" sent="62">American</wf>
 <wf wid="w62_4" sent="62" para="1">meat</wf>
[...]</text>
<terms>[...]
 <term tid="t62_3" type="open" lemma="American" pos="JJ">
  <span><target id="w62_3"/></span>
 </term>
 <term tid="t62_4" type="open" lemma="meat" pos="NN">
  <span><target id="w62_4"/></span>
 </term>
[...]</terms>
```

The test set needs to be aligned at word level so that target equivalents of the source-language checkpoints can be identified. This is done by appending them to a bigger parallel corpus, e.g., Europarl,[12] in order to help ensure accurate word alignments and avoid data sparseness. Of course, as with all alignment approaches, using in-domain parallel data, if available, would help ensure the accuracy of the word alignments, but for our work we make use of freely available data resources. The additional checkpoint filtering step (cf. Section 3.3) helps to circumvent any potential noisy alignments. The text is preprocessed with the Europarl tokeniser; then GIZA++ is applied, returning word alignments between the source and target sentences that make up the test set. The word alignments for the Italian–English sentence pair presented earlier are shown below:[13]

```
... 4-3 5-2 ...
```

3.2. Creating a linguistic checkpoint

Kybots are used to define linguistic phenomena (and extract their instances) that are to be evaluated. A Kybot profile specifies which information to extract from the KAF documents. For example the Kybot profile presented below extracts under the element "event" the term identifiers of those nouns that are immediately followed by

normally (though not necessarily) follows the noun which it modifies, whereas in English the standard order is to have the adjective first, followed by the noun.

[12] http://www.statmt.org/europarl/

[13] Note that identifiers in the alignment start from 0 while in the KAF files they start from 1.

an adjective in the Italian side of the test set. Target equivalents of the tokens identified by the Kybots in the source are obtained using the word alignments.

```
<Kybot id="kybot_n_a_it">
  <variables>
    <var name="X" type="term" pos="NOM*"/>
    <var name="Y" type="term" pos="ADJ*"/>
  </variables>
  <relations>
    <root span="X"/>
    <rel span="Y" pivot="X" direction="following" immediate="true"/>
  </relations>
  <events>
    <event eid="" target="$X/@tid" lemma="$X/@lemma" pos="$X/@pos"/>
    <role rid="" event="" target="$Y/@tid" lemma="$Y/@lemma" pos="$Y/@pos" rtype="follows"/>
  </events>
</Kybot>
```

For the sake of clarity, the case study presents a rather simple checkpoint. However, the expressive power of the Kybot engine allows the representation of more complex linguistic phenomena. As an example, we used a checkpoint for Italian→English that extracts sequences consisting of a noun followed by the preposition "di" followed by another noun, as there are a range of possible translations of this construction into English that are acceptable (at least in principle), e.g. keeping the preposition "of" between the two English nouns, using the genitive/possessive, or simply juxtaposing the two nouns in the target language. In order to define this checkpoint, one needs to select terms according to different fields, i.e., the first and third according to the PoS tag while the second according to both the lemma and the PoS tag.

The following commands load the Italian test file in KAF and the Kybot profile:

```
doc_load.pl --container-name docs_it en-it.it.kaf
kybot_load.pl --container-name kybots_it kybot_n_a_it.xml
```

Then the Kybot profile can be applied on the KAF document, and the matching terms are output:

```
kybot_run.pl --dry-run --profile-from-db --container-name docs_it --kybot-container-name \
kybots_it kybot_n_a_it.xml > out_n_a_it.xml
```

The following sample of the output shows the term "carne americana", terms 5 and 6 extracted from sentence 62, as it is a noun adjective sequence:

```
<kybotOut>
  <doc shortname="en-it.it.kaf">
   [...]
   <event eid="e66" target="t62_5" lemma="carne" pos="NOM"/>
   <role rid="r66" event="e66" target="t62_6" lemma="americano" pos="ADJ" rtype="follows"/>
   [...]
  </doc>
</kybotOut>
```

3.3. Filtering checkpoint instances

Optionally the tool can filter checkpoints based on corresponding PoS tags (recommended as it alleviates word alignment errors). This is done by establishing constraints based on PoS tags mappings between checkpoints extracted and the equivalent tokens in the target language (e.g., `NOM*=N*`, indicating that the equivalent token

in the target language of a token in the source with PoS tag NOM* must have the PoS tag N*). If a constraint is not fulfilled the corresponding instance of the checkpoint is dropped. Consider the following two constraints for Italian→English:

```
NOM* = N*
ADJ* = JJ*
```

The following sample of the Kybot output shows a term made up of tokens 24 (index 23 in word alignment) and 25 (index 24 in word alignment) from sentence 1:

```
<event eid="e1" target="t1_24" lemma="sinodo" pos="NOM"/>
<role rid="r1" event="e1" target="t1_25" lemma="patriarcale" pos="ADJ" rtype="follows"/>
```

Consider then the word alignments of the sentence.

```
... 23-22 22-23 21-24 26-24 24-25 ...
```

The tokens of the checkpoint instance (source tokens 23 and 24) get aligned to target tokens 22 and 25 (23 and 26 in the KAF file). Now, let us check these tokens in the target language.

```
[...]
<wf wid="w1_23" sent="1">of</wf>
<wf wid="w1_24" sent="1">the</wf>
<wf wid="w1_25" sent="1">Maronite</wf>
<wf wid="w1_26" sent="1">Patriarchal</wf>
<wf wid="w1_27" sent="1">Synod</wf>
[...]
<term tid="t1_23" type="open" lemma="of" pos="IN">
<span><target id="w1_23"/></span>
</term>
<term tid="t1_26" type="open" lemma="Patriarchal" pos="NP">
<span><target id="w1_26"/></span>
</term>
[...]
```

Thus, "sinodo patriarcale" is wrongly aligned to "of Patriarchal" (the right alignment is "Patriarchal Synod"). In PoS terms, NOM ADJ gets aligned to IN NP. The constraints are checked and they are not fulfilled in this case, as the PoS correspondence for "sinodo→of" (ADJ=IN) does not match the constraint ADJ=JJ. Thus, this instance of the checkpoint is filtered out.

3.4. Evaluating MT systems on the linguistic checkpoint

The performance of a MT system over a linguistic checkpoint is calculated by using an n-gram based evaluation metric. We split each system-generated translation and reference for a checkpoint into a set of n-grams, compute the number of matching n-grams and sum up the gains over all the n-grams as the score for this checkpoint. If the reference of the checkpoint is not consecutive, we use a wildcard character ("*") which can be matched by any word sequence. Below are some examples for the Italian→English language direction to demonstrate the n-gram matching.

When we calculate the recall of a set of checkpoints C, the references r of all checkpoints c in C (c can be a single checkpoint, a category, or a category group) are merged

Consecutive checkpoint:

Checkpoint:	Le proteste per la carne americana
Reference:	Protests over American meat
Candidate:	The protests for the American meat
Matched n-grams:	protests, American, meat, American meat

Non-consecutive checkpoint:

Checkpoint:	Le proteste * carne [*]
Reference:	Protests * meat
Candidate:	The protests for the American meat
Matched n-grams:	protests, meat, protests * meat

into one reference set R and the recall of C is obtained on R using equation 1.

$$R(C) = \frac{\sum_{r \in R} \sum_{ngram \in r} match(ngram)}{\sum_{r \in R} \sum_{ngram \in r} count(ngram)} \quad (1)$$

Since *n*-grams appearing in the target equivalents of instances of a linguistic phenomenon are searched for in the candidate translations, longer candidate translations have a better chance of returning higher scores. So we have implemented a length-based penalty to penalize longer candidate translations. We set the length-based penalty as in 2:

$$penalty = \begin{cases} \frac{length(R)}{length(C)} & \text{if length(C) >length(R)} \\ 1 & \text{otherwise} \end{cases} \quad (2)$$

where $length(C)$ is the average candidate sentence length and $length(R)$ is the average reference sentence length. Thus systems producing longer candidate sentences are penalized. The final score is calculated as in 3:

$$Score(C) = R(C) * penalty \quad (3)$$

The evaluation module is run in the following way:

```
java -jar delic4mt.jar -alg it-en.alignment -sl_kaf en-it.it.kaf -tl_kaf en-it.en.kaf \
 -lc out_n_a_it.xml -run mt_output.iten > mt_output.iten.n
```

The five parameters are: *alg* - word alignments for the gold standard (test set), against which the evaluation is performed; *sl_kaf* - source side of the test set in KAF; *tl_kaf* - target side of the test set in KAF; *lc* - Kybot output (i.e., instances of the checkpoint); and *run* - the output of a MT system, which is to be evaluated.

Below there is sample output which shows the matching of a translation hypothesis with the reference for the checkpoint instance "carne americana".

```
Sen_id: 62
Linguistic checkpoint identified on the Source: carne americana
Target equivalent (Reference): American meat
```

	Google	Bing	Systran
Score	0.6350	0.5537	0.4806

Table 1. Scores for the MT systems on the linguistic checkpoint `N ADJ`

```
Checking for n-gram matches for checkpoint instance: 65
Ref: American meat
Hypo: The protests for the American meat , [...]

n_gram matches : American, meat, American meat
# of n-grams in reference = 3
# of matching n-grams in hypothesis 62 = 3
```

The evaluation module finally sums up the number of matching n-grams (across the whole testset) for the linguistic checkpoint, and divides it by the total number of (checkpoint) n-grams in the reference set. The scores obtained by three online MT systems (Google Translate,[14] Microsoft Bing,[15] and Systran[16]) when evaluated over the example linguistic checkpoint (`N ADJ`) are shown in Table 1. Google obtains the highest score, 8.13 points higher than Bing, which in its turn is 7.13 points higher than Systran.

3.5. Statistical Significance Tests

Finally, for each pair of systems we can check whether the difference is statistically significant. A script included provides this functionality using paired bootstrapping resampling (Koehn, 2004). Let us check if the differences between Google's and Bing's outputs are significant:

```
lingcheckp_sig.pl google.iten.n bing.iten.n
Num results: 1204, times iterations: 5, num elements per iteration: 0.3
Randomised bootstrapping 6020 iterations with 361 elements
System a better than b in 6020 iterations out of 6020, i.e. 100%
```

There are 1,204 instances, we run $1{,}024 \cdot 5$ iterations with 30% of the instances in each iteration (randomly selected). This means running 6,020 iterations with 361 instances each. For all of the iterations the score of system a is higher than that of system b, thus we can say that the difference is statistically significant for $p = 0.01$.

4. Conclusions and Future Work

This paper has presented DELiC4MT, a tool for evaluating MT systems over user-specified linguistic phenomena. The tool makes extensive use of already available open-source software and standards and is easily adaptable to new languages and

[14] `http://translate.google.com`

[15] `http://www.microsofttranslator.com`

[16] `http://www.systran.co.uk`

linguistic phenomena. We have presented a case study which illustrates how the tool can be adapted to evaluate a specific linguistic phenomenon of a given language pair.

Regarding future work, we envisage the following tasks: (i) use alternative aligners and alignment heuristics to investigate if we can extract accurate alignments without the need for a significant amount of additional parallel data, (ii) compare the correlation of the diagnostic evaluation metric against that of other existing automatic evaluation metrics as well as against human judgements, and (iii) introduce a precision-based component into the diagnostic evaluation metric.

Acknowledgements

This work has been funded in part by the European Commission through the CoSyne project (FP7-ICT-4-248531) and Science Foundation Ireland (Grant No. 07/CE /I1142) as part of the Centre for Next Generation Localisation (www.cngl.ie) at Dublin City University.

Bibliography

Banerjee, Satanjeev and Alon Lavie. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, Proceedings of the ACL-05 Workshop*, pages 65–72, University of Michigan, Ann Arbor, Michigan, USA, 2005.

Bosma, W. E., Piek Vossen, Aitor Soroa, German Rigau, Maurizio Tesconi, Andrea Marchetti, Monica Monachini, and Carlo Aliprandi. Kaf: a generic semantic annotation format. In *Proceedings of the GL2009 Workshop on Semantic Annotation*, Sept. 2009.

Giménez, Jesús and Lluís Màrquez. Asiya: An Open Toolkit for Automatic Machine Translation (Meta-)Evaluation. *The Prague Bulletin of Mathematical Linguistics*, (94):77–86, 2010.

Koehn, Philipp. Statistical significance tests for machine translation evaluation. In Lin, Dekang and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, July 2004. Association for Computational Linguistics.

Naskar, Sudip Kumar, Antonio Toral, Federico Gaspari, and Andy Way. A framework for diagnostic evaluation of mt based on linguistic checkpoints. In *Proceedings of the 13th Machine Translation Summit*, pages 529–536, Xiamen, China, September 2011.

Och, Franz Josef and Hermann Ney. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29:19–51, March 2003. ISSN 0891-2017. doi: http://dx.doi.org/10.1162/089120103321337421.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 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, 2002. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/1073083.1073135.

Snover, Mathew, Bonnie Dorr, Richard Schwartz, John Makhoul, and Linnea Micciula. A Study of Translation Error Rate with Targeted Human Annotation. In *AMTA 2006: Proceedings of*

the 7th Conference of the Association for Machine Translation in the Americas: Visions for the Future of Machine Translation, pages 223–231, Cambridge, Massachusetts, USA, 2006.

Toral, Antonio, Federico Gaspari, Sudip Kumar Naskar, and Andy Way. A comparative evaluation of research vs. online mt systems. In Forcada, Mikel L., Heidi Depraetere, and Vincent Vandeghinste, editors, *Proceedings of the 15th Annual Conference of the European Associtation for Machine Translation*, pages 13–20, Leuven, Belgium, 2011. European Association for Machine Translation.

Vossen, Piek, German Rigau, Eneko Agirre, Aitor Soroa, Monica Monachini, and Roberto Bartolini. Kyoto: an open platform for mining facts. In *Proceedings of the 6th Workshop on Ontologies and Lexical Resources* , pages 1–10, Beijing, China, August 2010. Coling 2010 Organizing Committee.

Zhou, Ming, Bo Wang, Shujie Liu, Mu Li, Dongdong Zhang, and Tiejun Zhao. Diagnostic evaluation of machine translation systems using automatically constructed linguistic checkpoints. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 1121–1128, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. ISBN 978-1-905593-44-6.

Address for correspondence:
Antonio Toral
atoral@computing.dcu.ie
School of Computing, Dublin City University, Dublin 9, Ireland