Automatic MT Error Analysis: Hjerson Helping Addicter

Jan Berka[1], **Ondřej Bojar**[1], **Mark Fishel**[2], **Maja Popović**[3], **Daniel Zeman**[1]

[1] Charles University in Prague, Faculty of Mathematics and Physics,
Institute of Formal and Applied Linguistics
[2] Institute of Computer Linguistics, University of Zurich
[3] German Research Center for Artificial Intelligence (DFKI), Berlin
{berka,bojar,zeman}@ufal.mff.cuni.cz
fishel@cl.uzh.ch
maja.popovic@dfki.de

Abstract

We present a complex, open source tool for detailed machine translation error analysis providing the user with automatic error detection and classification, several monolingual alignment algorithms as well as with training and test corpus browsing. The tool is the result of a merge of automatic error detection and classification of Hjerson (Popović, 2011) and Addicter (Zeman et al., 2011) into the pipeline and web visualization of Addicter. It classifies errors into categories similar to those of Vilar et al. (2006), such as: morphological, reordering, missing words, extra words and lexical errors. The graphical user interface shows alignments in both training corpus and test data; the different classes of errors are colored. Also, the summary of errors can be displayed to provide an overall view of the MT system's weaknesses. The tool was developed in Linux, but it was tested on Windows too.

Keywords: machine translation, error analysis, visualization

1. Introduction

Until recently, most efforts in machine translation (MT) evaluation were to define methods producing a single score of MT output quality, correlating with human judgments as much as possible. This can be valuable for comparison of different systems' performances but the task of developing a translation system needs more detailed information – classification of errors, statistics of the error classes over a testing corpus and so on. It would be also helpful for the developer to be able to easily browse the analyzed data, e.g. to see all sentences with an occurrence of the error from a certain class.

We introduce a complex tool for such a detailed machine translation error analysis, which can be operated via command line as well as with a graphical user interface in a web browser. The tool provides the user with handy features, such as automatic detection and classification of errors, summarization of its results, test data browser and word explorer.

The paper is organized as follows: In Section 2. the automatic error detection and classification is briefly described. Section 3. introduces the graphical user interface in web browser. Section 4. gives insight of how to install and use Addicter[1] together with Hjerson, and finally Section 5. discusses some related work.

2. Automatic Error Detection and Classification

Automatic error classification is based on finding erroneous words in the translation output and assigning a corresponding error class to each of them. The classification of errors is vital for presentation and navigation in the system outputs. The user gets an overall picture of the output quality in the given test set and he or she can start by exploring the most frequent error types first.

The workflow of Addicter with Hjerson is shown in Figure 1, while the algorithms of Addicter and Hjerson error detection and classification are described in (Zeman et al., 2011) and (Popović, 2011) with more detail.

2.1. Error Detection

The first step of Hjerson and Addicter's analysis is the automatic detection of errors in the hypothesis translation. This is done by comparing tokens in the hypothesis with the reference translation, relying on some word alignment between the two texts.

Previous experience (i.a. Bojar (2011)) shows that the quality of the alignment is critical, otherwise many errors can be mis-classified (e.g. a pair of "missing" and "extra" errors instead of one error of bad lexical choice). Our tool circumvents current limitations of word alignment by providing several methods and allowing the user to choose which of them works best for the particular language pair, MT system and dataset.

Currently, Addicter internally implements the alignments of WER (Levenshtein, 1966), LCS (Hunt and McIlroy, 1976), a greedy injective alignment, and an injective HMM (Fishel et al., 2011). The user can provide any additional alignment, e.g. GIZA++ (Och and Ney, 2003) alignments, or alignments obtained by linking a reference-to-source alignment with the source-to-hypothesis alignment as reported in a verbose output of the MT system (briefly called "via-source").

[1] https://wiki.ufal.ms.mff.cuni.cz/user:zeman:addicter

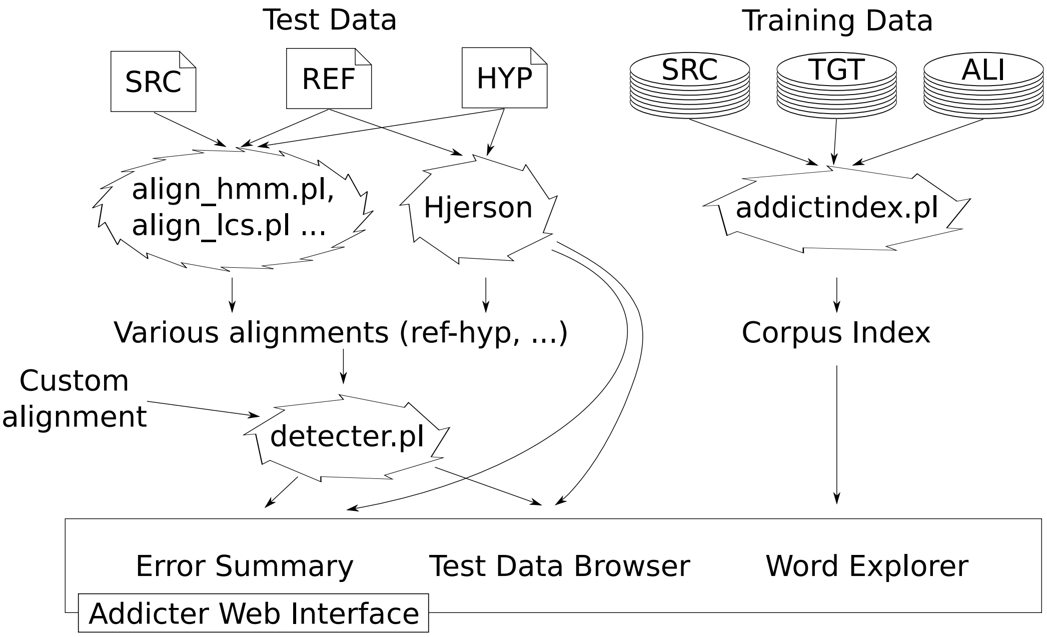


Figure 1: Overview of Addicter with Hjerson.

2.2. Error Classification

Hjerson and Addicter try to automatically classify errors into categories similar to those of Vilar et al. (2006), such as: morphological, reordering, missing words, extra words and lexical errors.

Currently, there are two algorithms available: the one from Addicter which strictly relies on the selected alignment and the one from Hjerson which simply compares the sets of words identified as erroneous due to a mismatch with the reference. (The latter case actually implies an injective alignment of the hypothesis and the reference.)

When using Hjerson for the classification, the user has two options: first, he can just use its own HTML or plain text output, as is described in (Popović, 2011). Second, he can use Addicter's script to transform Hjerson outputs into XML format readable by Addicter and then browse through the dataset using only Addicter.

The transformation of missing reference and extra hypothesis words is straightforward, as Addicter's classification contains these two categories as well. Because Hjerson works only with reference and hypothesis translations without the source text, it cannot distinguish between wrong lexical choice and untranslated token errors. Therefore, Hjerson's category "lexical error" is visualized by Addicter as "other mismatch" error class. Inflectional and reordering errors keep their category name.

The errors can be then summarized into a table showing their counts in the test data. When using GUI, the table is connected to the test data browser, so with just one click, the user can see the list of sentences with the occurrence of the given error type and even look at the sentences one by one in more detail.

3. Graphical User Interface

Addicter creates static or dynamic HTML pages, providing its user with the possibility to use a web browser as a simple graphical user interface. The starting point is the Addicter main page, which contains links to all defined experiments. Each experiment has its own main page, consisting of three parts: the Error Summary, the Test Data Browser and the Word Explorer.

3.1. Error Summary

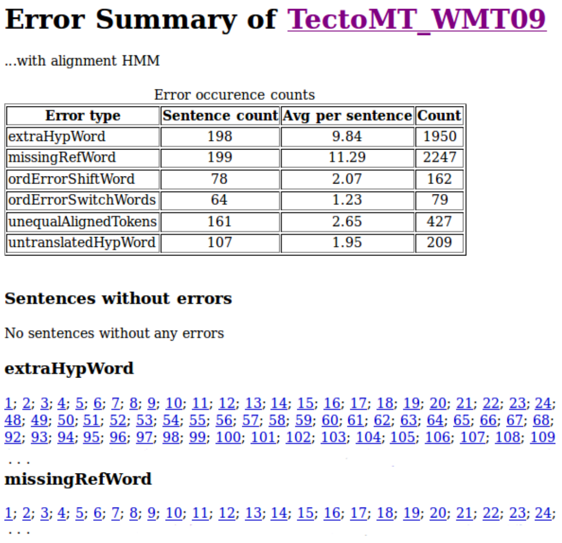
Test Data Browser and Word Explorer serve for examining individual sentences or words, the Error Summary page gives the user a global view of the dataset, providing a table with error counts for all error classes. This feature can reveal weak spots of the examined MT system and thus serve as a hint in what direction to focus further efforts first.

Under the summary table, for each error class there are links to sentences containing error(s) of given type to the Test Data Browser, so it is very simple to switch between global and local viewpoint and examine observed problems more profoundly.

An example of Error Summary page is shown in Figure 2 for experiment with name TectoMT_WMT09. The error detection and classification was done using the Hidden Markov Model alignment algorithm. At the top of the page is the summary table with absolute and relative counts of detected errors of recognized categories and under it are links to sentences containing given error types, e.g. there were no error-free sentences in the given dataset.

3.2. Test Data Browser

Test Data Browser is a tool for visual analysis of individual sentences in the test dataset. It displays the source text together with reference and hypothesis translations and their alignments. The underlying alignment method can be switched by selecting a different tab. Detected errors are highlighted, each error class having its own color, and summarized below. The user can switch between different alignments used for the error detection and classification.



Error Summary of TectoMT_WMT09

...with alignment HMM

Error occurence counts

Error type	Sentence count	Avg per sentence	Count
extraHypWord	198	9.84	1950
missingRefWord	199	11.29	2247
ordErrorShiftWord	78	2.07	162
ordErrorSwitchWords	64	1.23	79
unequalAlignedTokens	161	2.65	427
untranslatedHypWord	107	1.95	209

Sentences without errors

No sentences without any errors

extraHypWord

1; 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 12; 13; 14; 15; 16; 17; 18; 19; 20; 21; 22; 23; 24;
48; 49; 50; 51; 52; 53; 54; 55; 56; 57; 58; 59; 60; 61; 62; 63; 64; 65; 66; 67; 68;
92; 93; 94; 95; 96; 97; 98; 99; 100; 101; 102; 103; 104; 105; 106; 107; 108; 109
. . .

missingRefWord

1; 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 12; 13; 14; 15; 16; 17; 18; 19; 20; 21; 22; 23; 24;
. . .

Figure 2: Error Summary Page with the HMM Alignment

All words in the sentence are links to their pages in the Word Explorer (see Section 3.3.). By moving cursor over a particular word, all words matched by the active alignment are highlighted. Also, if the dataset contained not only surface forms, but also lemmas or other information, these are shown when hovering the cursor on any word in the error summary at the bottom of the page.

Figure 3 shows a sample sentence in the Test Data Browser. With the use of HMM alignment (highlighted), errors of three categories are detected (untranslated hypothesis word, missing reference word, extra hypothesis word errors). Clicking on the "WER" or other tab would display sentence with a different alignment algorithm used, probably leading to detection of different errors.

3.3. Word Explorer

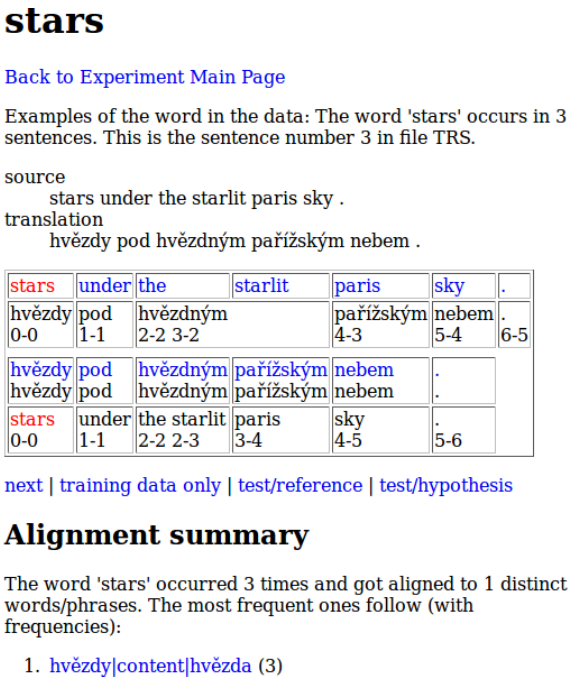
The Word Explorer allows to browse the training and test data and search for sentences containing a given word. It also displays simple statistics of word co-occurrence in the data and observed translations.

An example of Word Explorer usage is shown in Figure 4. The word *stars* occurred in three sentences in the corpus and got translated to the same Czech word *hvězdy* every time. Word Explorer automatically displays the first sentence with the word *stars* (the source text with the translation and the alignment), enabling a quick navigation through other sentences in the dataset. Also all the words in the table in the Word Explorer page are links to other pages of Word Explorer.

The user can navigate into Word Explorer from the Test Data Browser by clicking on any displayed word, or directly from the experiment main page, where he has the opportunity to list all words on source or target side of the dataset starting with a given letter or matching a given Perl-like regular expression.

3.4. Interpreting Results

In the example presented in Figure 2, most frequent errors are by far of the classes missing reference word and extra hypothesis word, which may lead to conclusion, that the MT system translates worse than it actually does, because this problem may be caused not only by poor translation, but (more probably) by poor performance of the alignment between reference and hypothesis. This is even more likely, when the numbers of missing reference and extra hypothesis words are similar, as in presented example.



stars

Back to Experiment Main Page

Examples of the word in the data: The word 'stars' occurs in 3 sentences. This is the sentence number 3 in file TRS.

source
    stars under the starlit paris sky .
translation
    hvězdy pod hvězdným pařížským nebem .

stars	under	the	starlit	paris	sky	.
hvězdy 0-0	pod 1-1	hvězdným 2-2 3-2		pařížským 4-3	nebem 5-4	. 6-5

hvězdy hvězdy	pod pod	hvězdným hvězdným	pařížským pařížským	nebem nebem	. .
stars 0-0	under 1-1	the starlit 2-2 2-3	paris 3-4	sky 4-5	. 5-6

next | training data only | test/reference | test/hypothesis

Alignment summary

The word 'stars' occurred 3 times and got aligned to 1 distinct words/phrases. The most frequent ones follow (with frequencies):

1. hvězdy|content|hvězda (3)

Figure 4: The Word *stars* in Word Explorer

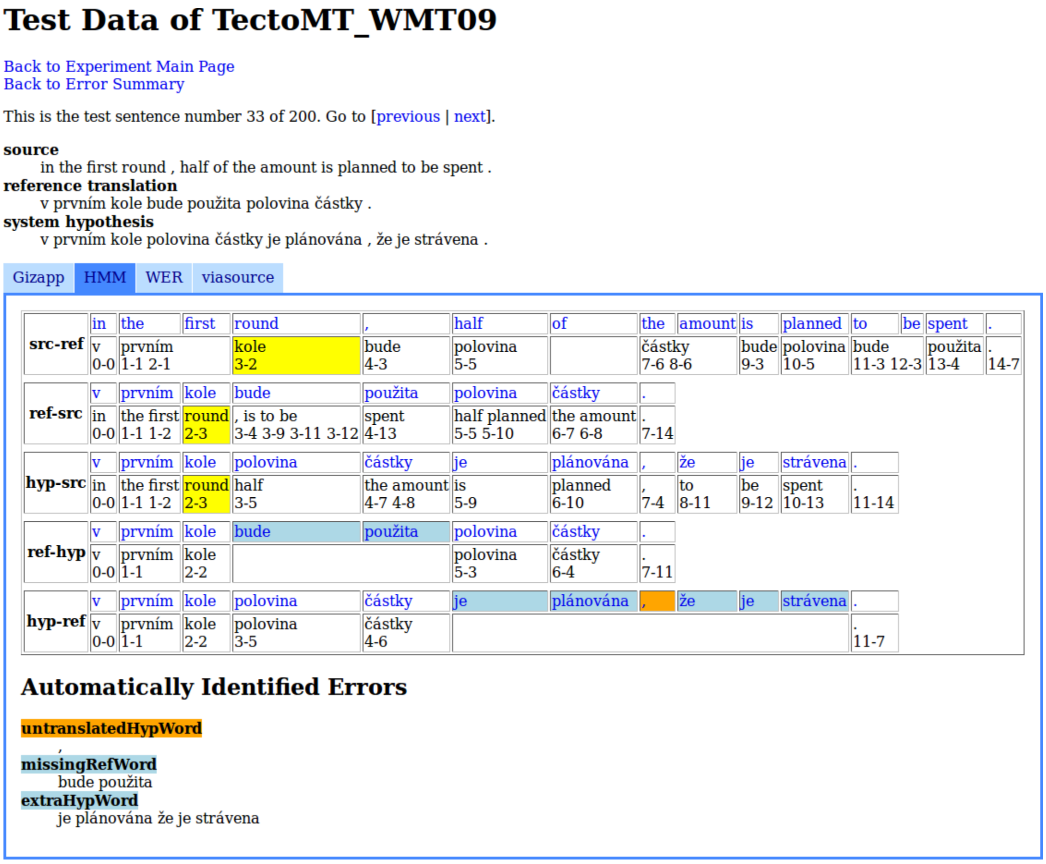
This suspicion can be then confirmed or refused by looking at individual sentences in Test Data Browser. As we see in example in Figure 3, the reference words "bude" and "použita" were labeled as missing in the hypothesis, while words "je" and "strávena" were labeled as extra. In fact, though, the word "bude" (meaning "will be" or "is going to be" in English) is just a future time of the word "je" (meaning "is" in English) and the meaning of "strávena" can be the same as the word "použita" (both meaning "used" in English). This implies that the MT system translated the sentence correctly (or to be more precise, did not make mistakes in the words "bude" and "použita") but the alignment failed to align corresponding words, possibly because of missing dictionary of synonyms.

4. Installation and Usage

Addicter is written in Perl and it needs a Perl interpreter. This is usually no problem on Unix systems, but on Windows the user may need to install Perl version $\geq$ 5.8.[2] To install Addicter, one has to simply check it out from our public SVN repository.[3]

[2] Possibilities include Strawberry Perl (`http://strawberryperl.com/`) and ActiveState Perl (`http://www.activestate.com/activeperl`).

[3] On Linux, use the following command: `svn checkout https://svn.ms.mff.cuni.cz/svn/statmt/trunk/addicter`. On Windows, use e.g. TortoiseSVN,



Test Data of TectoMT_WMT09

Back to Experiment Main Page
Back to Error Summary

This is the test sentence number 33 of 200. Go to [previous | next].

source
in the first round , half of the amount is planned to be spent .
reference translation
v prvním kole bude použita polovina částky .
system hypothesis
v prvním kole polovina částky je plánována , že je strávena .

src-ref	in	the	first	round	,	half	of	the	amount	is	planned	to	be	spent	.
	v 0-0	prvním 1-1 2-1		kole 3-2	bude 4-3	polovina 5-5		částky 7-6 8-6		bude 9-3	polovina 10-5	bude 11-3 12-3		použita 13-4	. 14-7

ref-src	v	prvním	kole	bude	použita	polovina	částky	.
	in 0-0	the first 1-1 1-2	round 2-3	, is to be 3-4 3-9 3-11 3-12	spent 4-13	half planned 5-5 5-10	the amount 6-7 6-8	. 7-14

hyp-src	v	prvním	kole	polovina	částky	je	plánována	,	že	je	strávena	.
	in 0-0	the first 1-1 1-2	round 2-3	half 3-5	the amount 4-7 4-8	is 5-9	planned 6-10	, 7-4	to 8-11	be 9-12	spent 10-13	. 11-14

ref-hyp	v	prvním	kole	bude	použita	polovina	částky	.
	v 0-0	prvním 1-1	kole 2-2			polovina 5-3	částky 6-4	. 7-11

hyp-ref	v	prvním	kole	polovina	částky	je	plánována	,	že	je	strávena	.
	v 0-0	prvním 1-1	kole 2-2	polovina 3-5	částky 4-6							. 11-7

Automatically Identified Errors

untranslatedHypWord
,
missingRefWord
bude použita
extraHypWord
je plánována že je strávena

Figure 3: Example of a Sentence in Test Data Browser.

Because Addicter uses some non-standard Perl libraries, it is necessary to check them out, too, and export them to Perl system path:

```
svn checkout \
  https://svn.ms.mff.cuni.cz/svn/dzlib \
  ~/dzlib
export PERL5LIB=~/dzlib:\$PERL5LIB
  # add the above line also to your ~/.bashrc
```

Hjerson is written in Python and so it needs a Python interpreter. Addicter now contains a slightly modified Hjerson script in its repository, so the user does not need to install Hjerson separately.[4]

4.1. Experiment Preparation

Addicter needs the dataset to be in a given, but simple and straightforward file structure: each experiment should be in its own directory[5] in the `cgi` directory. The data of the experiment have to be in following files in the experiment folder:

- `train.src` – source side of training corpus
- `train.tgt` – target side of training corpus
- `train.ali` – alignment of training corpus
- `test.src` – source side of test data
- `test.tgt` – reference translation of test data
- `test.system.tgt` – system output for test data
- `test.ali` – alignment of the source and reference translation of test data
- `test.system.ali` – alignment of the source and the system output for test data

Training corpus is mandatory for indexing (Word Explorer part in the GUI), alignments of source and reference and source and hypothesis translations of test corpus are optional and serve for displaying these alignments in the Test Data Browser.

There are number of alignment, indexing and error detection and classification scripts in Addicter. Alignment scripts are located in the `testchamber` directory and are named `align-x.pl`, where the `x` is the name of alignment algorithm. Scripts for running error detection and corpus indexing are in the `prepare` directory. By running any script without parameters, its meaning and usage are outputed to the command line.

If the user wants to run all alignment algorithms implemented in Addicter and do the error detection and classification based on them, there is script `rundetection.pl`, which does all the work for him. E.g., running the command

`http://tortoisesvn.net/`. In both cases, use the word `public` for both the username and the password.

[4] The original Hjerson is available at `http://www.dfki.de/~mapo02/hjerson/`

[5] which will be then displayed as the experiment name in the GUI

```
./rundetection.pl \
 --src=../cgi/MT/test.src \
 --ref=../cgi/MT/test.tgt \
 --hyp=../cgi/MT/test.system.tgt \
 --work=../cgi/MT \
```

from the `prepare` folder will make LCS, WER, HMM and greedy injective alignment of the experiment `MT`, run error detection and classification and make sure everything is in the right file structure, ready for viewing in Addicters GUI.

Hjerson error detection and classification is done separately by script `runhjerson.pl`. Running

```
./runhjerson.pl \
 --ref=../cgi/MT/test.tgt \
 --hyp=../cgi/MT/test.system.tgt \
 --baseref=../cgi/MT/test.base.tgt \
 --basehyp=../cgi/MT/test.system.base.tgt \
 --work=../cgi/MT/Hjerson
```

will make folder `Hjerson` in the experiment folder, run Hjerson on the data and export detected errors to Addicter-readable format, assuming that the files including `base` in their suffixes are base forms of corresponding data.

Once the experiment is ready for viewing, all the user has to do is to run the script `server.pl` from Addicter main folder and copy the outputed link to his web browser.

5. Related Work

Note that both Addicter's and Hjerson's accuracy was evaluated on four datasets by Fishel et al. (2012). Both tools, and especially Hjerson, achieve reasonable correlations when ranking error types. In other words, the order of error types in Error Summary is relatively reliable and can be used to steer the development of the MT system. The precision and recall of error marking in individual sentences (as displayed in Test Data Browser) are less satisfactory, ranging from just 5% for missed words up to 90% for errors in form with the average precision around 30% and recall around 50%.

There is a number of different MT evaluation systems with different functionalities. One of them is Meteor (Denkowski and Lavie, 2011), which implements its own monolingual alignment of reference and hypothesis translations based on aligning the exact forms in the first step, the rest is then stemmed and the aligning algorithm runs on the stems. For the unaligned words, the alignment of synonymous words is attempted and the alignment based of paraphrases serves as the last resort. Based on the alignment, a single score is produced. Meteor also includes Meteor-xRay for visualization of the results via Gnuplot and XeTeX. The xRay can display alignment tables and histograms of score distribution either for one or for comparison of two systems. We believe that the aggressive alignment algorithm of Meteor would in many cases lead to better results than the alignments currently available in Addicter. We plan to integrate Meteor alignment into Addicter, too.

iBleu (Madnani, 2011) is a visualization tool for computing the BLEU score of a single MT system or comparing outputs from different systems in a web browser. Asiya (Giménez and Màrquez, 2010) is a tool implementing a large number of different metrics in one application.

All systems introduced above focus on computing a single score in one or more metrics. Woodpecker (Zhou et al., 2008) is a rare exception but it is available for Windows only and it is currently aimed primarily at errors in English to Chinese and Chinese to English translation because it relies on a pre-defined set of linguistic phenomena.

One of the early systems to simplify the assessment of translation quality is EvalTrans[6] (Nießen et al., 2000). EvalTrans supports much less fine-grained evaluation than Addicter, each sentence is simply ranked on one or more scales. The most interesting feature of EvalTrans is the ability to extrapolate past manual quality judgments to new sentences based on sentence similarity. The reliability of these semi-automatic judgments is then reasonably high and can be easily improved by providing further manual annotation.

The Experimental Management System (Koehn, 2010) is a tool allowing smart execution of training and testing of machine translation experiment with one configuration file, automatically detecting re-usable steps for multiple runs with changed settings. It also provides a detailed analysis of the experiment run, including n-gram precision and recall and color-coded output in web browser.

6. Conclusion

We presented Addicter, a tool for inspection of parallel data and machine translation outputs that also includes several variants of automatic error detection. The main addition presented here is the incorporation of Hjerson, another error-detection tool into Addicter user interface.

We hope that our tool will simplify the error analysis both when fine-tuning an established MT system but perhaps more importantly when trying to bootstrap an MT system for a new language pair.

7. Acknowledgements

The work on this project was supported by the project EuroMatrixPlus (FP7-ICT-2007-3-231720 of the EU and 7E09003+7E11051 of the Czech Republic) and the Czech Science Foundation grants P406/11/1499 and P406/10/P259.

8. References

Ondřej Bojar. 2011. Analyzing Error Types in English-Czech Machine Translation. *Prague Bulletin of Mathematical Linguistics*, 95:63–76, March.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the EMNLP 2011 Workshop on Statistical Machine Translation*.

Mark Fishel, Ondřej Bojar, Daniel Zeman, and Jan Berka. 2011. Automatic Translation Error Analysis. In *Text, Speech and Dialogue: 14th International Conference, TSD 2011*, volume LNAI 3658. Springer Verlag, September.

[6] `http://www-i6.informatik.rwth-aachen.de/web/Software/EvalTrans/index.html`

Mark Fishel, Ondřej Bojar, and Maja Popović. 2012. Terra: a Collection of Translation Error-Annotated Corpora. In *International Conference on Language Resources and Evaluation*, Istanbul, Turkey, May.

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

James W. Hunt and M. Douglas McIlroy. 1976. An Algorithm for Differential File Comparison. Computing Science Technical Report 41, Bell Laboratories, June.

Phillip Koehn. 2010. An Experimental Management System. *The Prague Bulletin of Mathematical Linguistics*, 94.

Vladimir Iosifovich Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10(8):707–710, February.

Nitin Madnani. 2011. iBLEU: Interactively Debugging & Scoring Statistical Machine Translation Systems. In *Proceedings of the Fifth IEEE International Conference on Semantic Computing*.

Sonja Nießen, Franz Josef Och, Gregor Leusch, and Hermann Ney. 2000. An evaluation tool for machine translation: Fast evaluation for mt research. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation*, pages 39–45, Athens, Greece, May-June.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

Maja Popović. 2011. Hjerson: An open source tool for automatic error classification of machine translation output. *The Prague Bulletin of Mathematical Linguistics*, pages 59–68.

David Vilar, Jia Xu, Luis Fernando D'Haro, and Hermann Ney. 2006. Error Analysis of Machine Translation Output. In *International Conference on Language Resources and Evaluation*, pages 697–702, Genoa, Italy, May.

Daniel Zeman, Mark Fishel, Jan Berka, and Ondřej Bojar. 2011. Addicter: What is wrong with my translations? *The Prague Bulletin of Mathematical Linguistics*, 96:79–88.

Ming Zhou, Bo Wang, Shujie Liu, Mu Li, Dongdong Zhang, and Tiejun Zhao. 2008. Diagnostic evaluation of machine translation systems using automatically constructed linguistic check-points. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 1121–1128, Manchester, UK.