Asiya

An Open Toolkit for Automatic
Machine Translation (Meta-)Evaluation

Technical Manual
Version 3.0

Meritxell Gonzàlez, Jesús Giménez
TALP Research Center, LSI Department
Universitat Politècnica de Catalunya
Jordi Girona Salgado 1–3, 08034, Barcelona

February 2014

Abstract

This document describes the installation and usage of the Asiya Open Toolkit for Automatic Machine Translation (Meta-)Evaluation (Giménez & Màrquez, 2010).[1] We also overview the tSearch functionality, the on-line interfaces and the AsiyaWS.

Asiya offers system and metric developers a text interface to a rich repository of evaluation metrics and meta-metrics, and a tool for a quick search and examination of the results.

The Asiya toolkit is the natural evolution/extension of its predecessor, the IQmt Framework (Giménez & Amigó, 2006).

Asiya is publicly available at `http://asiya.lsi.upc.edu`.

[1] This work has been partially funded by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 247762 (FAUST project, FP7-ICT-2009-4-247762) and grant agreement number 247914 (MOLTO project, FP7-ICT-2009-4-247914), and by the Spanish Government project OpenMT-2, TIN2009-14675-C03.

Contents

1 Introduction 4

2 Installation 5
2.1 Building Asiya . . . 5
2.2 External Components . . . 7
2.2.1 Borrowing Metrics . . . 7
2.2.2 Borrowing Linguistic Processors . . . 7
2.3 Building tSearch . . . 9
2.3.1 Installing Cassandra . . . 9

3 Tool Description and Usage 10
3.1 Evaluation Options . . . 12
3.2 Meta-Evaluation Options . . . 13
3.2.1 Finding Optimal Metrics and Metric Sets . . . 15
3.3 General Options . . . 15
3.4 tSearch Options . . . 17

4 Metric Set 18
4.1 Lexical Similarity . . . 19
4.2 Syntactic Similarity . . . 21
4.3 Semantic Similarity . . . 24

5 Confidence Estimation 44
5.1 Translation Quality . . . 44
5.2 Translation Difficulty . . . 46

6 Learning to combine CE measures for quality pairwise ranking 47

7 On-line Interfaces and Web Service 47
7.1 Asiya Online Interface . . . 49
7.2 Asiya tSearch . . . 49
7.3 AsiyaWS . . . 50

8 Ongoing and Future Steps 51

References 52

A Glossary of Evaluation Measures 57

B tSearch User Manual 59

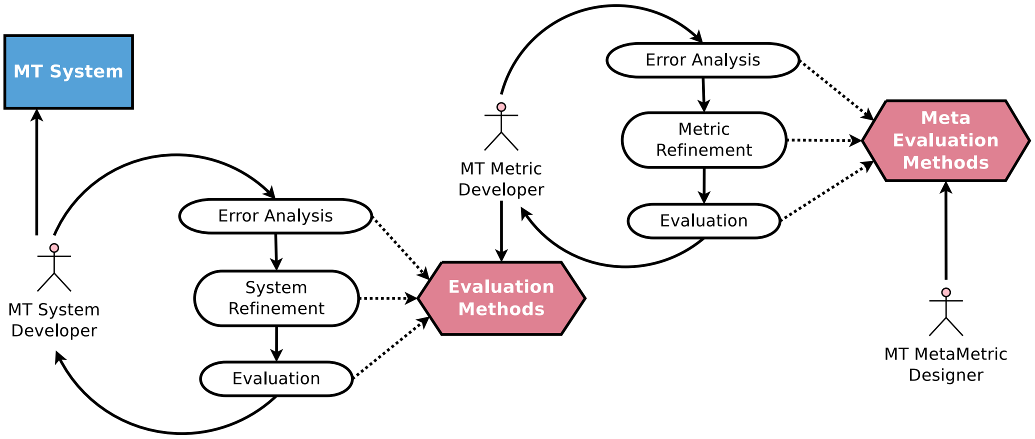


Figure 1: System development cycle in Machine Translation

1 Introduction

Evaluation methods are a key ingredient in the development cycle of Machine Translation (MT) systems (see Figure 1). They are used to identify the system weak points (error analysis), to adjust the internal system parameters (system refinement) and to measure the system performance, as compared to other systems or to different versions of the same system (evaluation). Evaluation methods are not a static component. On the contrary, far from being perfect, they evolve in the same manner that MT systems do. Their development cycle is similar: their weak points are analyzed, they are refined, and they are compared to other metrics or to different versions of the same metric so as to measure their effectiveness. For that purpose they rely on additional meta-evaluation methods.

ASIYA is an open toolkit aimed at covering the evaluation needs of system and metric developers along the development cycle[2]. In short, ASIYA provides a common interface to a compiled collection of evaluation and meta-evaluation methods (i.e., hexagonal boxes in Figure 1). The metric repository incorporates the latest versions of most popular metrics, operating at different linguistic dimensions (lexical, syntactic, and semantic) and based on different similarity assumptions (precision, recall, overlap, edit rate, etc.). ASIYA also incorporates schemes for metric combination, i.e., for integrating the scores conferred by different metrics into a single measure of quality. The meta-metric repository includes both measures based on human acceptability (e.g., correlation with human assessments), and human likeness, such as ORANGE (Lin & Och, 2004b) and KING (Amigó et al., 2005).

The ASIYA tSEARCH (Gonzàlez et al., 2013) is a complementary tool for translation

[2] Asiya was the Israelite wife of the Pharaoh who adopted Moses after her maids found him floating in the Nile river (see `http://en.wikipedia.org/wiki/Asiya`).

error analysis and system comparison. It allows to search for those translations (of a given testbed) that match some criteria related to their quality (as assessed by the automatic scores) and the intermediate analysis results (output of the analyzers). This tool has been specially designed to aid developers to alleviate the burden of manually inspecting the quality of their translations.

Finally, in order to ease the use of the evaluation toolkit and provide visual information related to translation quality, we developed three different on-line interfaces. The AsiyaWSis a RESTful web service to access a remote instance of the Asiya Toolkit running on a GRID cluster. In the line of today's *cloud computing* services, this web service is intended to facilitate the remote usage of the application without the need for downloading and locally installing all the modules. The on-line interfaces (Gonzàlez et al., 2012) provide graphical interaction. They are intended to allow users to familiarize with the Asiya functionalities and to analyze real testbeds in a graphical and interactive environment. They favour a rapid evaluation and examination of testbeds using just a browser and a quick inspection of the results obtained, including fast search, graphs, annotations and visualization of parse trees.

2 Installation

The following subsections provide the basic set of instructions for building the Asiya Toolkit (Section 2.1), the external software components required for metric computation (Section 2.2) and the tSearch tool (Section 2.3).

2.1 Building Asiya

Check out the latest development version from the subversion repository:

- svn co http://svn-rdlab.lsi.upc.edu/subversion/asiya/public asiya

To configure this module cd into to the newly created './asiya' directory and type the following:

```
perl Makefile.PL
```

Alternatively, if you plan to install this tool somewhere other than your system's perl library directory, you can type something like this:

```
perl Makefile.PL PREFIX=/home/me/perl
```

This will check whether all the required modules are installed or not. Prerequisites are:

- XML management:
    - XML::Twig 3.34[3]
    - XML::DOM 1.44 (requires, XML::Parser::PerlSAX, available inside libxml-perl-0.08)

[3] http://www.xmltwig.com/xmltwig/

    - XML::Parser 2.36 (requires expat)[4]
    - XML::RegExp 0.03

- Benchmark 1.11
- Modern::Perl 1.03
- Getopt::Long 2.38
- Data::Dumper 2.126
- Data::UUID 1.218
- IO::File 1.14
- Modern::Perl 1.03
- POSIX 1.08
- Unicode::String 2.09
- File::Basename 2.78
- File::ReadBackwards 1.04
- Scalar::Util 1.23
- Scalar::Numeric 0.22
- Statistics::Descriptive 3.0100
- Statistics::Distributions 1.02
- Statistics::LSNoHistory 0.01
- Statistics::RankCorrelation 0.11_3
- SVMTool 1.3

All required Perl modules are available at the CPAN repository[5] except SVMTool which is available under the './tools' directory and also in the SVMTool public website[6]. Then, build the package by typing:

```
make
```

If you have write access to the installation directories, you may then become super user and install it so it is available to all other users:

```
sudo make install
```

Otherwise, remember to properly set the PERL5LIB variable so Perl programs may find Asiya modules:

```
export PERL5LIB=$PERL5LIB:/home/me/soft/asiya/lib
```

The './tools' directory must be included in the PERL5LIB variable:

[4] http://sourceforge.net/projects/expat/

[5] http://search.cpan.org/

[6] http://nlp.lsi.upc.edu/svmtool/

```
export PERL5LIB=$PERL5LIB:/home/me/soft/asiya/tools/
```

The 'ASIYA_HOME' environment variable (pointing to the target installation folder) must be declared:

```
export ASIYA_HOME=/home/me/soft/asiya
```

Finally, include the folder containing Asiya executable files in the PATH variable:

```
export PATH=$PATH:/home/me/soft/asiya/bin
```

2.2 External Components

Asiya relies on several external components for metric computation. They all are located in the './tools' directory, and some may require re-compilation. In this case, simply 'cd' to the corresponding directory and follow the instructions in the corresponding 'README' and/or 'INSTALL' files.

It is not necessary to install all the external components listed below, but only those required by the metrics intended to be used. However, using a metric without properly installing it or any of its pre-requisites will cause an execution error.

2.2.1 Borrowing Metrics

- METEOR, GTM and TER require Java[7].
- METEOR and TER also require WordNet[8]. In its turn, WordNet requires Tcl/tk[9]. After installation, you must properly set the WNHOME and PATH variables:

  ```
  export PATH=$PATH:/usr/local/WordNet-3.0/bin
  export WNHOME=/usr/local/WordNet-3.0
  ```

- BLEU, NIST, and ROUGE require Perl[10].

2.2.2 Borrowing Linguistic Processors

Linguistic metrics rely on automatic processors:

- Shallow Parsing metrics
  - SVMTool (Giménez & Màrquez, 2004a)[11] for part-of-speech tagging and lemmatization. SVMTool requires Perl. Remember to properly edit the 'PERL5LIB' and 'PATH' variables:

    ```
    export PERL5LIB=$PERL5LIB:/home/me/soft/asiya/tools/svmtool-1.3/lib
    export PATH=$PATH:/home/me/soft/asiya/tools/svmtool-1.3/bin
    ```

[7] http://www.java.com

[8] http://wordnet.princeton.edu

[9] http://www.tcl.tk/

[10] http://www.perl.org/

[11] http://nlp.lsi.upc.edu/svmtool/

  - BIOS for base phrase chunking (Surdeanu et al., 2005)[12], which requires Java.

- Constituent Parsing metrics
  - Charniak-Johnson Constituent Parser (Charniak & Johnson, 2005)[13], which requires C++.
  - Berkeley Parser constituent parser (Petrov et al., 2006; Petrov & Klein, 2007)[14]. Remember to properly set the following and variables:

    ```
    export BKY_PARSER=$ASIYA_HOME/tools/berkeleyparser
    export PATH=$BKY_PARSER:$PATH
    export CLASSPATH=$BKY_PARSER:$CLASSPATH
    ```

- Dedendency Parsing metrics
  - MINIPAR dependency parser (Lin, 1998)[15]. MINIPAR requires the GNU Standard C++ Library v3 (libstdc++5). Remember to properly set the 'MINIPATH' and 'PATH' variables:

    ```
    export MINIPATH=/home/me/soft/asiya/tools/minipar/data
    export PATH=$PATH:/home/me/soft/asiya/tools/minipar/pdemo
    ```

  - Bonsai v3.2 (Candito et al., 2010b)[16] is used for both dependency and constituent parsing of French. It was trained on a dependency version of the French Treebank (Candito et al., 2010a). It requires python 2.5 or higher and MALT or Berkeley parser. We use the MALT variant in Asiya. Remember to properly set the following variables:

    ```
    export BONSAI=$ASIYA_HOME/tools/bonsai_v3.2
    export MALT_BONSAI_DIR=$ASIYA_HOME/tools/malt-1.3.1
    export PYTHONPATH=/usr/local/lib/python2.6/site-packages
    ```

  - MALT parser 1.7.1 (Nivre et al., 2007)[17], which requires Melt Tagger (Denis & Sagot, 2009). The parsing model for French was trained on a dependency version of the French Treebank (Candito et al., 2010a), and the SVMTool was also trained on the same Treebank, so Asiyauses it instead of the MElt tagger. Remember to properly set the following variables:

    ```
    export MALT_DIR=$ASIYA_HOME/tools/malt-1.7.2
    ```

- Named Entities metrics
  - SVMTool for part-of-speech tagging and lemmatization.
  - BIOS for base phrase chunking and named entity recognition and classification.

- Semantic Roles metrics use:
  - BIOS suite.

[12] http://www.surdeanu.name/mihai/bios/

[13] ftp://ftp.cs.brown.edu/pub/nlparser/

[14] http://code.google.com/p/berkeleyparser/

[15] http://www.cs.ualberta.ca/~lindek/minipar.htm

[16] http://alpage.inria.fr/statgram/

[17] http://www.maltparser.org

  - Charniak-Johnson Parser.
  - SwiRL semantic role labeler (Surdeanu & Turmo, 2005; Màrquez et al., 2005)[18]. SwiRL requires JAVA.
  - XLike semantic role labeler (Lluís et al., 2013)[19]. XLike requires Freeling (Carreras et al., 2004).
- Discourse Representations metrics use the C&C Tools[20], which require C++ and SWI PROLOG[21]. Detailed installation instructions are available in the C&C Tools website[22]. Apart from the CCG parser, remember to install the BOXER component. BOXER expects the prolog interpreter under the name of 'pl'. Thus, you may need to edit the PROLOG variable in the Makefile. Alternatively, you can create a soft link (i.e., '`ln -s /usr/bin/swipl /usr/bin/pl`').

2.3 Building tSearch

tSearch is not strictly required to run Asiya. This tool helps to do searches on the evaluation results making use of the Cassandra database. To use the tSearch tool, install the following Perl modules required available at the CPAN repository:

- Parse::RecDescent
- Math::Round
- Scalar::Util::Numeric
- JSON

Install the following python libraries required by the tSearch database:

- pycassa [23]
- python-storable [24]

2.3.1 Installing Cassandra

Cassandra is a NoSQL database solution used by tSearch[25]. The most basic configuration is a single node configuration which is described as follows[26].

Download the Cassandra sources at:

- http://archive.apache.org/dist/cassandra/1.1.7/

[18] `http://www.surdeanu.name/mihai/swirl/`

[19] `http://www.xlike.org/`

[20] `http://svn.ask.it.usyd.edu.au/trac/candc/`

[21] `http://www.swi-prolog.org/`

[22] `http://svn.ask.it.usyd.edu.au/trac/candc/wiki/Installation`

[23] `http://pycassa.github.io/pycassa/installation.html`

[24] `https://gitorious.org/python-storable`

[25] http://cassandra.apache.org/

[26] Instructions of initializing a multinode cluster are available at `http://www.datastax.com/docs/1.1/initialize/index`

Note that Cassandra is a Java progam, so that it requires the Java Runtime Environment (JRE) 1.6 or later installed on linux systems.

Afterwards, you should create the log directory:

```
tar -zxvf apache-cassandra-\$VERSION.tar.gz
cd apache-cassandra-\$VERSION
sudo mkdir -p /var/log/cassandra
sudo chown -R ‘whoami‘ /var/log/cassandra
sudo mkdir -p /var/lib/cassandra
sudo chown -R ‘whoami‘ /var/lib/cassandra
```

You are free to edit the path names of the file-system locations that Cassandra uses for logging and data storage. Just edit the configuration files in the ‘conf/’ folder.

Setup the CASSANDRA_HOME variable to the Cassandra root directory and compile the sources.

```
cd apache-cassandra
ant
```

Run Cassandra using one of the following commands:

```
sudo service cassandra starts
/<install_directory>/bin/cassandra (background)
/<install_directory>/bin/cassandra -f (foreground)
```

Check that Cassandra is up and running:

```
cd /<install_directory>
$ bin/nodetool ring -h localhost
```

Once Cassandra is running, copy and execute the following code in a Python shell in order to create the column families required:

```
import pycassa
from pycassa.system_manager import *
sys = SystemManager()
sys.create_keyspace(’tsearch’, SIMPLE_STRATEGY, {’replication_factor’:’1’})
sys.create_column_family(’tsearch’, ’scores’, super=False,
                         comparator_type=FLOAT_TYPE)
sys.create_column_family(’tsearch’, ’linguistic_elements’)
sys.create_column_family(’tsearch’, ’metric_basic_info’)
sys.close()
```

3 Tool Description and Usage

Asiya operates over predefined test suites, i.e., over fixed sets of translation test cases (King & Falkedal, 1990). A test case consists of a source segment, a set of candidate translations and a set of manually-produced reference translations. The utility of a test suite is intimately related to its representativity, which depends on a number of variables (e.g., language pair, translation domain, number and type of references, system typology, etc.). These variables determine the space in which MT systems and evaluation metrics will be allowed to express their capabilities, and, therefore, condition the results of any evaluation and meta-evaluation process conducted upon them.

Asiya requires the user to provide the test suite definition through a configuration file. Different test suites must be placed in different folders with their corresponding configuration files. Preferred input format is the NIST XML, as specified in the Metrics MaTr Evaluation Plan (Callison-Burch et al., 2010)[27]. For instance, the sample configuration file in Table 1 defines source material (source.xml), candidate translations (candidates.xml), and reference translations (references.xml). If the source file is not provided, the first reference will be used as source for those metrics which take it into consideration. Candidate and reference files are required.

```
# lines starting with '#' are ignored

src=source.xml
sys=candidates.xml
ref=references.xml

some_metrics=-TERp METEOR-pa CP-STM-6 DP-Or(*) SR-Or(*) DR-Or(*) DR-STM-6
some_systems=system01 system05 system07
some_refs=reference02 reference04
```

Table 1: Sample configuration file ('sample.config')

Asiya may be then called by typing the following on the command line:

```
Asiya.pl sample.config
```

When called without any additional option further than the name of the configuration file, Asiya will read the file, check its validity (i.e., whether the defined files exist and are well-formed) and terminate. Setting the '-v' option adds some verbosity to the process. No output will be delivered to the user other than status and error messages. However, several files will be generated. Input XML files are processed and texts are extracted and saved as plain '.txt' files in the original data folder. There will be one source file, and as many candidate and reference files as systems and reference sets are specified in the XML file. The correspondence between text files and document and segment identifiers is kept through simple index files ('.idx').

[27] http://www.nist.gov/itl/iad/mig/metricsmatr10.cfm

3.1 Evaluation Options

Evaluation reports are generated using the '-eval' option followed by a comma-separated list of evaluation schemes to apply. The following schemes are currently available:

- **Single** metric scores
- **Ulc** normalized arithmetic mean of metric scores
- **Queen** scores as defined by Amigó et al. (2005)
- **Model <file>** learned combination of scores (<file> should contain the learned model). See Section 6 for details about the learning methods.

Thus, for instance:

```
Asiya.pl -v -eval single,ulc,queen sample.config
```

will compute and print individual metric scores, their normalized arithmetic mean, and Queen scores (all based on a predefined set of metrics, see Section 3.3).

Several output formats are available through the '-o' option. Default format is '-o mmatrix' (one system, doc or segment per line, each metric in a different column). By default metrics are sorted according to the order as typed by the user. It is also possible to sort them alphabetically using the '-sorted name' option. Other output formats are '-o smatrix' (one metric per line, each system in a different column) and '-o nist' which saves metric scores into files complying with the NIST output format as specified in the Metrics MaTr Evaluation Plan.

As an additional option, evaluation scores for the reference translations may be also retrieved through the '-include_refs' option. References will be evaluated against all other references in the test suite.

```
Asiya.pl -v -eval single -include_refs sample.config
```

Besides evaluation reports, Asiya generates, for convenience, several intermediate files:

- **Metric scores:** Results of metric executions are stored in the './scores/' folder in the working directory, so as to avoid having to re-evaluate already evaluated translations. It is possible, however, to force metric recomputation by setting the '-remake' flag. Moreover, because each metric generates its reports in its own format, we have designed a specific XML representation format which allows us to access metric scores in a unified manner. For instance, the report in Table 2 corresponds to the scores conferred by the BLEU metric to system 'system05' when compared to reference 'reference01' over two documents totaling 5 segments. Our XML format allows for representing metric scores at the segment, document, and system levels.
- **Linguistic annotations:** Metrics based on syntactic and semantic similarity may perform automatic linguistic processing of the source, candidate and reference material. When necessary, these will be stored in the original data folder so as to avoid having to repeat the parsing of previously parsed texts.

```
<?xml version=”1.0”?>
<!DOCTYPE asiya SYSTEM ”asiya.dtd” []>
<SET metric=”BLEU” n_docs=”2” n_segments=”5” hyp=”system05”
        ref=”reference01” score=”0.40442589”>
 <DOC id=”AFP_ARB_20060206.0155” n=”1” n_segments=”2” score=”0.29500965”>
  <SEG n=”1”>0.22033597</S>
  <SEG n=”2”>0.31347640</S>
 </DOC>
 <DOC id=”AFP_ARB_20060207.0030” n=”2” n_segments=”3” score=”0.46204650”>
  <SEG n=”3”>0.15106877</S>
  <SEG n=”4”>0.56761755</S>
  <SEG n=”5”>0.35930885</S>
 </DOC>
<SET>
```

Table 2: Sample XML metric score file

3.2 Meta-Evaluation Options

Meta-evaluation reports are generated using the ‘-metaeval’ option followed by a comma-separated list of metric combination schemes and a comma-separated list of meta-evaluation criteria to apply. Five criteria are currently available:

- **Pearson** correlation coefficients (Pearson, 1914)
- **Spearman** correlation coefficients (Spearman, 1904)
- **Kendall** correlation coefficients (Kendall, 1955)
- **King** scores (Amigó et al., 2005)
- **Orange** scores (Lin & Och, 2004b)

For instance:

```
Asiya.pl -v -metaeval single king,orange sample.config
```

will compute and print King and Orange scores for each metric in the default metric set.

In order to compute correlation coefficients, human assessments must be provided using the ‘-assessments’ option followed by the name of the file containing them. The assessments file must comply with the NIST CSV format (i.e., comma-separated fields, one assessment per line, see an example in Table 3). The assessments file may also contain a header line and comments (lines starting with ‘#’). The purpose of the header is to describe the position of the fields identifying the referent item (i.e., system, document and segment identifiers) and the score itself. The ‘systemId’ and ‘score’ field descriptors are mandatory (i.e., system-level scores). If the ‘documentId’ and 'segmentId' descriptors are added, Asiya prepares to read document and segment-level scores. In the absence of a header, the one from the example in Table 3 will be used (i.e., segment-level scores).

```
# systemId, documentId, segmentId, score
sample_system, AFP_ARB_20060206.0155, 1, 3
sample_system, AFP_ARB_20060206.0155, 2, 2
sample_system, AFP_ARB_20060206.0155, 3, 3
...
```

Table 3: Sample assessments CSV file

The header is followed by assessments. System, document and segment identifiers must match those specified in the test suite input files. If the NIST XML input format is used, identifiers are taken from the corresponding XML attributes. In the case of the raw input format, system identifiers correspond to their respective input file names, all segments are assumed to correspond to a single document named *'UNKNOWN_DOC'*, and line numbers are used as segment identifiers (starting at line 1). If only system and segment identifiers are given, then Asiya interprets that segment identifiers are absolute and will try to automatically assign them the corresponding document and document-relative segment identifiers by following the document order in the source file.

If several scores for the same referent are provided (e.g., by different human assessors) Asiya will take their average. Additionally, Asiya allows a single CSV assessments file to contain assessments at different levels of granularity (i.e., system, document and segment-level scores), which may be set using the '-g' option. If document or system-level scores are not provided, they are computed by averaging over individual segments (or documents, if segment scores are not available).

For instance:

```
Asiya.pl -v -metaeval single pearson,spearman,kendall -g seg
         -assessments human_scores.csv sample.config
```

will print Pearson, Spearman and Kendall correlation coefficients between segment-level metric scores and human assessments provided in the 'human_cores.csv' file for each metric in the default metric set.

By default, correlation coefficients are accompanied by 95% confidence intervals computed using the Fisher's z-distribution (Fisher, 1924). Since the sampling distribution of correlation coefficients is not normally distributed, they are first converted to Fisher's z using the Fisher transformation (Fisher, 1921). The values of Fisher's z in the confidence interval are then converted back into correlation coefficients. It is also possible to compute correlation coefficients and confidence intervals applying bootstrap resampling (Efron & Tibshirani, 1986). If the number of samples is reasonably small, as it may be the case when computing correlation with system-level assessments, exhaustive resampling is feasible ('-ci xbootstrap'). Otherwise, the number of resamplings may be selected using the '-ci bootstrap' and '-n_resamplings' options (1,000 resamplings by default). Also, the degree of statistical may be adjusted using the '-alfa' option. For instance:

```
Asiya.pl -v -metaeval single pearson,spearman,kendall
         -g seg -assessments human_scores.csv -ci boostrap
         -n_resamplings 100 -alfa 0.01 sample.config
```

compute segment-level correlation coefficients based on bootstrap resampling, over 100 resamplings, at a 99% statistical significance. Asiya implements also paired metric bootstrap resampling (Koehn, 2004). All metrics are compared pairwise. The proportion of times each metric outperforms the other, in terms of the selected criterion, is retrieved.

3.2.1 Finding Optimal Metrics and Metric Sets

Finally, Asiya provides a mechanism to determine optimal metric sets. These may be found using the '-optimize' option followed by a specific evaluation scheme and meta-evaluation criterion (see Section 3.2). Because exploring all possible metric combinations becomes prohibitive as the number of metrics grows, Asiya currently implements an approximate suboptimal search. The algorithm is simple. First, metrics are ranked by their individual quality according the selected meta-evaluation criterion. Then, they are progressively added to the optimal metric set if and only if in doing so the global quality increases. If the meta-evaluation criterion involves human assessments these must be provided using the '-assessments' option as described in Section 3.2. For instance:

```
Asiya.pl -v -optimize ulc pearson -g seg
         -assessments human_scores.seg sample.config
```

will find a suboptimal metric set, among the default set of metrics for English, by maximizing correlation with the collection of segment-level human assessments provided in the 'human_scores.seg' file.

3.3 General Options

Input Format Candidate and reference translations may be represented in a single file or in separate files. Apart from the NIST XML format, previous NIST SGML and plain text formats are also accepted. Input format is specified using the '-i' option followed by any of the formats available ('nist' or 'raw'). If the input is already tokenized, used the '-no_tok' option to skip the tokenization within Asiya.

Language Pair By default, Asiya assumes the test suite to correspond to an into-English translation task. This behavior may be changed using the '-srclang' (source language) and 'trglang' (target language) options. Metrics based on linguistic analysis, or using dictionaries or paraphrases, require a proper setting of these values. It is also possible to tell Asiya whether text case matters or not. By default, Asiya will assume the text to be case-sensitive. This behavior may be changed using the '-srccase' (source case) '-trgcase' (target case) options. For instance:

```
Asiya.pl -v -srclang fr -srccase cs -trglang es -trgcase ci
         sample.config
```

will tell Asiya that the test suite corresponds to a French-to-Spanish translation task, being the source case sensitive, whereas target texts are not.

Pre-defined Sets By default, all systems and references are considered, and scores are computed based on a predefined set of metrics which varies depending on the target language. The set of metrics to be used may be specified using the '-metric_set' and/or the '-m' options. The '-metric_set' option must be followed by the name of the set as specified in the config file (see Table 1). The '-m' option must be followed by a comma-separated list of metric names. The effect of these options is cumulative. For instance:

```
Asiya.pl -v -eval single -metric_set some_metrics -m Ol,GTM-2,
         sample+.config
```

will compute the metrics specified in the 'some_metrics' set (see Table 1) together with the 'O_l' and 'GTM-2' metrics. Analogously, you may tell Asiya to focus on specific system sets ('-system_set' and '-s') and reference sets ('-reference_set' and '-r').

```
Asiya.pl -v -metric_set some_metrics -system_set some_systems
         -reference_set some_refs sample+.config
```

The full list of metric, system and reference names defined in the test suite may be listed using the '-metric_names', '-system_names' and '-reference_names' options, respectively[28]. For instance:

```
Asiya.pl -v -metric_names sample.config
```

In all cases, Asiya will check that the defined sets are valid, i.e., that the metric, system and reference names are correct.

Other Options Another important parameter is the granularity of the results. Setting the granularity allows developers to perform separate analyses of system-level, document-level and segment-level results, both over evaluation and meta-evaluation reports. This parameter may be set using the '-g' option to either system-level ('-g sys'), document-level ('-g doc'), segment-level ('-g seg') granularity, or all levels ('-g all'). Default granularity is at the system level. The length and precision of floating point numbers may be adjusted using the '-float_length' (10 by default) and '-float_precision' options (8 by default). Finally, the '-tex' flag produces, when applicable, (meta-)evaluation reports directly in LaTeX format.

[28] The set of available metrics depends on language pair settings.

3.4 tSearch Options

The tSearch sources are located under the './tools' directory. To start using the tSearch command line interface, create the TSEARCH_HOME variable and include the folder containing the tSearch files in the PATH and the PERL5LIB variables:

```
TSEARCH_HOME=$ASIYA_HOME/tools/tsearch/search
export TSEARCH_HOME
export PERL5LIB=\$TSEARCH_HOME:\$PERL5LIB
export PATH=\$TSEARCH_HOME:\$PATH
```

Within the 'tsearch' directory, we find 1) the scripts used by the Asiya toolkit to insert in the database the data being calculated during the evaluation, 2) a tarball with a testkit to checkout whether the tSearch and the Cassandra database is working correctly, and 3) the tSearch toolkit under the 'tsearch/search' folder.

The main script to use the tSearch tookit is 'tSearch.pl'. There are some options defined:

```
-i : inserts the data into the Cassandra database
-t : the testbed id
-q : queries the database
-o : the output format ('xml', 'json')
-c : the confirmation to insert the data on demand
```

Insert There are two ways of inserting the output of Asiya in the Cassandra database:

1. During the evaluation. Just use the '-tsearch' option when you call the Asiya script:

   ```
   Asiya.pl -v -eval single -tsearch Asiya.config
   ```

   tSearch identifies each testbed by a unique id. You can assign this identifies using the option '-testbedid'. If this option is not given, Asiya will create a unique id and inform at the end of the execution.

   ```
   Asiya.pl -v -eval single -tsearch -testbedid sample Asiya.config

   [INSERT DONE] testbed-id to use for querying: sample
   ```

2. Anytime after the evaluation. The '-i' option of the 'tSearch.pl' script reads the results of Asiya and feeds the database. You should give the path to the folder where you have 'scores' folder created by Asiya and, optionally, the testbed identifier. If the database contains a previous dataset with the same identifier, all previous the data will be replaced. By default, the system assigns a unique new testbed identifier when not given by the user.

   ```
   tSearch.pl -i  -p <datapath> [-t <testbedid>]
        datapath: the full path to the testbed workspace where the
                  scores folder is located.
        testbedid: the identifier of the data in the database.

   [INSERT DONE] testbed-id to use for querying: sample
   ```

Query The tSearch command line interface allows you to ask for translations matching a spcefic criteria. To do so, you must use the option '-q' followed by the query, the option '-t' indicating the tesbted id and the option '-p' indicating the path to the testbed.

```
tSearch.pl -q "BLEU > AVG" -t sample -p <datapath>
```

Output format By default, tSearch print the output in JSON format. This behavior may be changed using the '-o' option followed by 'xml' which prints the results in a XML format. For instance:

```
tSearch.pl -t "sample" -q "BLEU > AVG" -o "xml"
```

will print all the translations from the testbed evaluated having a BLEU score above the average in a xml format.

On demand Confirmation Some of the queries need information that is computed on demand. For instance:

```
tSearch.pl -t sample -q "LE[NE(ORG)]"
Your query asks for data not precalculated.
It will take few seconds but it will be required only once.
Would you like to continue? [Y/N]
```

This behaviour is a design decision motivated by the time required to compute all the possible queries. Instead, tSearch initializes only the data required by most common queries (score based ones) and let up the user choice to initialize all the remaining data upon request. The option '-c' allows you initialize all the data at once. Note that using this option will take longer only the first time you do a query. Then, all the data will be ready.

```
tSearch.pl -t "sample" -q "LE[NE(ORG)]" -c
```

4 Metric Set

We have compiled a rich set of measures which evaluate translation quality based on different viewpoints and similarity assumptions. In all cases, automatic translations are compared against a set of human reference translations. We have borrowed existing measures and we have also implemented new ones. The set of available metrics depends on the source and target language. A complete list of metrics can may be obtained by typing on the command line:

```
Asiya.pl -metric_names -srclang <srclang> -trglang <trglang> s
```

In the following subsections, we provide a description of the metric set. We have grouped metrics according to the linguistic level at which they operate (lexical, syntactic, and semantic).

4.1 Lexical Similarity

Below, we describe the set of lexical measures used in this work, grouped according to the type of measure computed.

Edit Distance

WER (Word Error Rate) (Nießen et al., 2000) We use $-$WER to make this into a precision measure. This measure is based on the Levenshtein distance (Levenshtein, 1966) —the minimum number of substitutions, deletions and insertions that have to be performed to convert the automatic translation into a valid translation (i.e., a human reference).

PER (Position-independent Word Error Rate) (Tillmann et al., 1997) We use $-$PER. A shortcoming of the WER measure is that it does not allow reorderings of words. In order to overcome this problem, the position independent word error rate (PER) compares the words in the two sentences without taking the word order into account. Word order is not taken into account.

TER (Translation Edit Rate) (Snover et al., 2006; Snover et al., 2009) TER measures the amount of post-editing that a human would have to perform to change a system output so it exactly matches a reference translation. Possible edits include insertions, deletions, and substitutions of single words as well as shifts of word sequences. All edits have equal cost. We use $-$TER. Four variants are included:

- **-TER→** default (i.e., with stemming and synonymy lookup but without paraphrase support).
- **-TER$_{\mathbf{base}}$** → base (i.e., without stemming, synonymy lookup, nor paraphrase support).
- **-TER$_{\mathbf{p}}$** → with stemming, synonymy lookup and paraphrase support (i.e., phrase substitutions).
- **-TER$_{\mathbf{pA}}$** → TER$_p$ tuned towards adequacy.

ALGN Rate of aligned words. Alignments are computed with the Berkeley aligner[29]. Three variants are available, depending on the alignment used:

- **ALGN$_s$** rate of aligned words between the candidate translation and the source.
- **ALGN$_p$** comparisson between the number of aligned words between the candidate translation and source vs. the alignments between the reference and the source.
- **ALGN$_r$** rate of aligned words between the reference and the candidate translations using the source as a pivot.

[29] `https://code.google.com/p/berkeleyaligner/`

Lexical Precision

BLEU (Papineni et al., 2001)[30] We use accumulated and individual BLEU scores for several n-gram lengths ($n = 1...4$, default is 4). Default is accumulated BLEU score up to 4-grams and smoothed as described by Lin and Och (2004b).

NIST (Doddington, 2002) We use accumulated and individual NIST scores for several n-gram lengths ($n = 1...5$, default is 5). Default is NIST score up to 5-grams.

$\mathbf{P}_{\boldsymbol{l}}$ stands for Lexical Precision, it computes the min-interesection of items (tokens) in the reference and the candidate divided by the items in the candidate.

Lexical Recall

ROUGE (Lin & Och, 2004a) Eight variants are available[31]:

$\mathbf{ROUGE}_{\boldsymbol{n}} \rightarrow$ for several n-gram lengths ($n = 1...4$).

$\mathbf{ROUGE}_{\boldsymbol{L}} \rightarrow$ longest common subsequence (LCS).

$\mathbf{ROUGE}_{\boldsymbol{S\star}} \rightarrow$ skip bigrams with no max-gap-length.

$\mathbf{ROUGE}_{\boldsymbol{SU\star}} \rightarrow$ skip bigrams with no max-gap-length, including unigrams.

$\mathbf{ROUGE}_{\boldsymbol{W}} \rightarrow$ weighted longest common subsequence (WLCS) with weighting factor $w = 1.2$.

$\mathbf{R}_{\boldsymbol{l}}$ stands for Lexical Recall, it computes the max-interesection of items (tokens) in the reference and the candidate divided by the items in the reference.

F-Measure

$\mathbf{GTM}_{\boldsymbol{e}}$ (Melamed et al., 2003) Three variants, corresponding to different values of the e parameter controlling the reward for longer matchings ($e \in \{1, 2, 3\}$), are available [32].

METEOR (Banerjee & Lavie, 2005; Denkowski & Lavie, 2010) Four variants have been computed[33]:

$\mathbf{METEOR}_{\mathbf{ex}} \rightarrow$ only exact matching.

$\mathbf{METEOR}_{\mathbf{st}} \rightarrow$ plus stem matching.

$\mathbf{METEOR}_{\mathbf{sy}} \rightarrow$ plus synonym matching.

$\mathbf{METEOR}_{\mathbf{pa}} \rightarrow$ plus paraphrase matching.

$\mathbf{F}_{\boldsymbol{l}}$ Lexical F1, is the F-mesure for P_l and R_l, that is $(2 * P_l * R_l)/(P + R)$ for a single reference.

$\mathbf{O}_{\boldsymbol{l}}$ Lexical overlap is a measure inspired on the Jaccard coeficient for sets similarity. Lexical items associated to candidate and reference translations are considered as two separate sets of items. Overlap is computed as the cardinality of their intersection divided into the cardinality of their union.

[30] BLEU and NIST measures are computed using the NIST MT evaluation kit v13a, which is available at `http://www.nist.gov/speech/tools/`.

[31] We use ROUGE version 1.5.5. Options are '`-z SPL -2 -1 -U -m -r 1000 -n 4 -w 1.2 -c 95 -d`'.

[32] We use GTM version 1.4, which is available at `http://nlp.cs.nyu.edu/GTM/`.

[33] We use METEOR version 1.2, which is available at `http://www.cs.cmu.edu/~alavie/METEOR/`.

NGRAM Cosine and Jaccard coefficient similarity measures for both token and character n-grams considering $n \in [2, 5]$ (i.e., sixteen features). Additionally, one Jaccard-based similarity measure for "pseudo-prefixes" (considering only up to four initial characters for every token).

NGRAM-$_{cos}$Char$_N$ngrams Cosine coefficient similarity for character n-grams considering $N \in [2, 5]$.

NGRAM-$_{cos}$Tok$_N$ngrams Cosine coefficient similarity for token n-grams considering $N \in [2, 5]$.

NGRAM-$_{jac}$Cognates Jaccard-based similarity measure for "pseudo-prefixes".

NGRAM-$_{jac}$Char$_N$ngrams Jaccard coefficient similarity for character n-grams considering $N \in [2, 5]$.

NGRAM-$_{jac}$Tok$_N$ngrams Jaccard coefficient similarity for token n-grams considering $N \in [2, 5]$.

NGRAM-lenratio

4.2 Syntactic Similarity

Syntactic measures have been grouped into three different families: *SP*, *DP* and *CP*, which respectively capture similarities over shallow-syntactic structures, dependency relations and constituent parse trees.

On Shallow Parsing (SP)

SP measures analyze similarities at the level of parts of speech, word lemmas, and base phrase chunks. Sentences are automatically annotated using the SVMTool (Giménez & Màrquez, 2004b) and BIOS (Surdeanu et al., 2005) linguistic processors. Table 4 and Table 5 show the PoS tag set used for English, derived from the Penn Treebank[34] tag set (Marcus et al., 1993). Several coarse classes are included. Word lemmas have been obtained by matching word-PoS pairs against an off-the-shelf lemmary containing 185,201 different <word, PoS> entries. Table 6 shows base phrase chunk types for English.

As for texts in Catalan and Spanish, we used the Ancora corpus (Taulé et al., 2008) to train the SVMTool and the 3LB corpus[35] to train the BIOS processor. Tag set for Spanish, derived from the PAROLE tag set, is shown in Table 7, Table 8 and Table 9.

The texts in French are parsed using the BONSAI v3.2tool[36] (Candito et al., 2010b). It was trained with the French Treebank (Candito et al., 2010a) and adapted for dependency parsing. The Tag set derived from the corpus is shown in Table 10.

Finally, German texts are parsed using the BERKELEY PARSER[37] and the German model provided (Petrov & Klein, 2007), which was trained on the TIGER

[34] http://www.cis.upenn.edu/~treebank/

[35] The 3LB project is funded by the Spanish Ministry of Science and Technology (FIT-15050-2002-244), visit the project website at http://www.dlsi.ua.es/projectes/3lb/

[36] http://alpage.inria.fr/statgram/frdep/

[37] http://code.google.com/p/berkeleyparser/

Treebank (Brants et al., 2002) and the Tüba-D/Z Treebank (Telljohann et al., 2004). The Tag set derived from the grammar model is shown in Table 11 and Table 12.

We instantiate overlap over parts of speech and chunk types (only English, Catalan and Spanish). The goal is to capture the proportion of lexical items correctly translated according to their shallow syntactic realization:

SP-$O_p(t)$ Lexical overlap according to the part-of-speech 't'. For instance, SP-O_p(NN) roughly reflects the proportion of correctly translated singular nouns. We also offer a coarser measure, SP-$O_p(\star)$ which computes the average lexical overlap over all parts of speech.

SP-$O_c(t)$ Lexical overlap according to the base phrase chunk type 't'. For instance, SP-O_c(NP) roughly reflects the proportion of successfully translated noun phrases. We also include the SP-$O_c(\star)$ measure, which computes the average lexical overlap over all chunk types.

At a more abstract level, we also use the NIST measure to compute accumulated/individual (optional 'i') scores over sequences of ($n = 1...5$):

SP-NIST(i)$_l$-n Lemmas.

SP-NIST(i)$_p$-n Parts of speech.

SP-NIST(i)$_c$-n Base phrase chunks.

SP-NIST(i)$_{iob}$-n Chunk IOB labels[38]

On Dependency Parsing (DP)

DP measures capture similarities between dependency trees associated to automatic and reference translations. Dependency trees are obtained using MINIPAR (Lin, 1998) for English texts and MALT v3.2 (Hall & Nivre, 2008) for English, Spanish, Catalan and German. Hence, we have created two families of measures to distinguish the parser used:

DP- Measures calculated by MINIPAR. A brief description of grammatical categories and relations used by MINPAR may be found in Table 13 and Table 14.

DPm- Measures calculated by MALT v3.2 parser. The pretrained models for English and French were obtained with the Penn Treebank (Marcus et al., 1993) and the French Treebank (Candito et al., 2010a), respectively. The grammatical relations for Spanish and Catalan were trained using the 3LB corpus (Navarro et al., 2003).

Then, two subfamilies of measures have been included for each of the above families:

DP(m)-HWCM(i)-l These measures correspond to variants of the head-word chain matching (HWCM) measure presented by Liu and Gildea (2005). All head-word chains are retrieved. The fraction of matching head-word chains

[38] IOB labels are used to denote the position (Inside, Outside, or Beginning of a chunk) and, if applicable, the type of chunk.

of a given length $l \in [1..9]$ between the candidate and the reference translation is computed. 'i' is the optional parameter for "individual" rather than cummulated scores. The '(m)' stands for MALT V3.2measures. We have slightly modified so as to consider different head-word chain types:

DP(m)-HWCM(i)$_w$-l words.

DP(m)-HWCM(i)$_c$-l grammatical categories.

DP(m)-HWCM(i)$_r$-l grammatical relations.

Average accumulated scores up to a given chain length are also used. For instance, DP-HWCMi$_w$-4 retrieves matching proportion of length-4 word-chains and DP-HWCM$_w$-3 retrieves average accumulated proportion of matching word-chains up to length 3. Analogously, DP-HWCM$_c$-3 and DP-HWCM$_r$-3 compute average accumulated proportion of category/relation chains up to length 2. Default length is 4.

DP(m)-$O_l|O_c|O_r$ These measures correspond exactly to the LEVEL, GRAM and TREE measures introduced by Amigó et al. (2006).

DP(m)-$O_l(l)$ Overlap between words hanging at level $l \in [1..9]$, or deeper.

DP(m)-$O_c(t)$ Overlap between words *directly hanging* from terminal nodes (i.e. grammatical categories) of type 't'.

DP(m)-$O_r(t)$ Overlap between words ruled by non-terminal nodes (i.e. grammatical relationships) of type 't'.

Node types are determined by grammatical categories and relations as defined by the dependency parser. For instance, DP-O_r-s reflects lexical overlap between subtrees of type 's' (subject). Additionally, we consider three coarser measures, (DP-$O_l(\star)$, DP-$O_c(\star)$ and DP-$O_r(\star)$) which correspond to the uniformly averaged values over all levels, categories, and relations, respectively.

On Constituent Parsing (CP)

CP measures analyze similarities between constituent parse trees associated to automatic and reference translations. Constituent trees are obtained using the Charniak and Johnson (2005) Max-Ent reranking parser for English, the Bonsai v3.2 tool for French (Candito et al., 2010b), and the Berkeley Parser for German (Petrov & Klein, 2007). description of the tag set employed is available in Table 15, 16 and 17 for English, French and German respectively. Three types of measures have been defined:

CP-STM(i)$_l$ These measures correspond to variants of the syntactic tree matching (STM) measure by Liu and Gildea (2005). All semantic subpaths in the candidate and the reference trees are retrieved. The fraction of matching subpaths of a given length $l \in [1..9]$ is computed. Average accumulated scores up to a given tree depth d may be used as well. For instance, CP-STMi$_5$ retrieves the proportion of length-5 matching subpaths. Average accumulated scores may be computed as well. For instance, CP-STM$_4$ retrieves average accumulated proportion of matching subpaths up to length 4.

CP-$O_p(t)$ Similarly to the SP-$O_p(t)$ metrics, these measures compute lexical overlap according to the part-of-speech 't'.

CP-$O_c(t)$ These measures compute lexical overlap according to the phrase constituent type 't'. The difference between these measures and SP-$O_c(t)$ variants is in the phrase scope. In contrast to base phrase chunks, constituents allow for phrase embedding and overlap.

4.3 Semantic Similarity

We have designed three new families of measures: *NE*, *SR*, and *DR*, which are intended to capture similarities over named entities, semantic roles, and discourse representations, respectively.

On Named Entities (NE)

NE measures analyze similarities between automatic and reference translations by comparing the named entities which occur in them. Sentences are automatically annotated using the BIOS package (Surdeanu et al., 2005). BIOS requires at the input shallow parsed text, which is obtained as described in Section 4.2. At the output, BIOS returns the text enriched with NE information. The list of NE types utilized is available in Table 18.

We have defined two types of measures:

NE-$O_e(t)$ Lexical overlap between NEs according to their type t. For instance, NE-O_e(PER) reflects lexical overlap between NEs of type 'PER' (i.e., person), which provides a rough estimate of the successfully translated proportion of person names. We also use the NE-$O_e(\star)$ measure, which considers average lexical overlap over all NE types. This measure focus only on actual NEs. We use also another variant, NE-$O_e(\star\star)$, which includes overlap among items of type 'O' (i.e., Not-a-NE).

NE-$M_e(t)$ Lexical matching between NEs according to their type t. For instance, NE-M_e(LOC) reflects the proportion of fully translated locations. The NE-$M_e(\star)$ measure considers average lexical matching over all NE types, excluding type 'O'.

On Explicit Semantic Analysis (ESA)

ESA (Gabrilovich & Markovitch, 2007) creates a similarity vector between a sentence and a set of documents. We compare the vector similarities given for the source, the reference and the candidate translations. Our set of documents correspond to the opening paragraphs of 100k Wikipedia articles as in 2010. We have defined two types of measures (avaiable for English, Spanish and German):

ESA-1 Compares the similarity vectors between the reference and the candidate translations.

ESA-2 Compares the similarity vectors between the source and the candidate translations.

On Semantic Roles (SR)

SR measures analyze similarities between automatic and reference translations by comparing the SRs (i.e., arguments and adjuncts) which occur in them. Sentences are automatically annotated using the SwiRL package (Surdeanu &

Turmo, 2005). SwiRL returns the text annotated with SRs following the notation of the Proposition Bank (Palmer et al., 2005). A list of SR types is available in Table 19.

We have defined three types of measures:

SR-$\boldsymbol{O_r(t)}$ Lexical overlap between SRs according to their type t. For instance, SR-O_r(Arg0) reflects lexical overlap between 'Arg0' arguments. SR-$O_r(\star)$ considers the average lexical overlap over all SR types.

SR-$\boldsymbol{M_r(t)}$ Lexical matching between SRs according to their type t. For instance, the measure SR-M_r(MOD) reflects the proportion of fully translated modal adjuncts. The SR-$M_r(\star)$ measure considers the average lexical matching over all SR types.

SR-$\boldsymbol{O_r}$ This measure reflects 'role overlap', i.e., overlap between semantic roles independently of their lexical realization.

We also use more restrictive versions of these measures (SR-$M_{rv}(t)$, SR-$O_{rv}(t)$, and SR-O_{rv}), which require SRs to be associated to the same verb.

On Discourse Representations (DR)

DR measures analyze similarities between automatic and reference translations by comparing their discourse representations. For the discursive analysis of texts, DR measures rely on the C&C Tools (Curran et al., 2007). Tables 20 to 24 describe some aspects of the DRS representations utilized. For instance, Tables 20 and 21 respectively show basic and complex DRS conditions. Table 22 shows DRS subtypes. Tables 23 and 24 show symbols for one-place and two-place relations.

Three kinds of measures have been defined:

DR-STM(i)$_{\boldsymbol{l}}$ These measures are similar to the *CP-STM* variants discussed above, in this case applied to DR structures instead of constituent trees. All semantic subpaths in the candidate and the reference trees are retrieved. The fraction of matching subpaths of a given length $l \in [1..9]$ is computed.

DR-$\boldsymbol{O_r(t)}$ These measures compute lexical overlap between discourse $\underline{r}$epresentations structures (i.e., discourse referents and discourse conditions) according to their type 't'. For instance, DR-O_r(pred) roughly reflects lexical overlap between the referents associated to predicates (i.e., one-place properties), whereas DR-O_r(imp) reflects lexical overlap between referents associated to implication conditions. We also use the DR-$O_r(\star)$ measure, which computes average lexical overlap over all DRS types.

DR-$\boldsymbol{O_{rp}(t)}$ These measures compute morphosyntactic overlap (i.e., between grammatical categories –$\underline{p}$arts-of-speech– associated to lexical items) between discourse $\underline{r}$epresentation structures of the same type. We also use the DR-$O_{rp}(\star)$ measure, which computes average morphosyntactic overlap over all DRS types.

Type	**Description**
CC	Coordinating conjunction, e.g., and,but,or...
CD	Cardinal Number
DT	Determiner
EX	Existential there
FW	Foreign Word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List Item Marker
MD	Modal, e.g., can, could, might, may...
NN	Noun, singular or mass
NNP	Proper Noun, singular
NNPS	Proper Noun, plural
NNS	Noun, plural
PDT	Predeterminer, e.g., all, both ... when they precede an article
POS	Possessive Ending, e.g., Nouns ending in 's
PRP	Personal Pronoun, e.g., I, me, you, he...
PRP$	Possessive Pronoun, e.g., my, your, mine, yours...
RB	Adverb. Most words that end in -ly as well as degree words like quite, too and very.
RBR	Adverb. comparative Adverbs with the comparative ending -er, with a strictly comparative meaning.
RBS	Adverb, superlative
RP	Particle
SYM	Symbol. Should be used for mathematical, scientific or technical symbols
TO	to
UH	Interjection, e.g., uh, well, yes, my...

Table 4: PoS tag set for English (1/2)

Type	**Description**
VB	Verb, base form subsumes imperatives, infinitives and subjunctives
VBD	Verb, past tense includes the conditional form of the verb to be
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner, e.g., which, and that when it is used as a relative pronoun
WP	Wh-pronoun, e.g., what, who, whom...
WP$	Possessive wh-pronoun
WRB	Wh-adverb, e.g., how, where why
#	Punctuation Tags
$	
”	
(	
)	
,	
.	
:	
“	

COARSE TAGS	
N	Nouns
V	Verbs
J	Adjectives
R	Adverbs
P	Pronouns
W	Wh- pronouns
F	Punctuation

Table 5: PoS tag set for English (2/2)

Type	**Description**
ADJP	Adjective phrase
ADVP	Adverb phrase
CONJP	Conjunction
INTJ	Interjection
LST	List marker
NP	Noun phrase
PP	Preposition
PRT	Particle
SBAR	Subordinated Clause
UCP	Unlike Coordinated phrase
VP	Verb phrase
O	Not-A-Phrase

Table 6: Base phrase chunking tag set for English

Type	Description
NOUN	
NC	Noun, Common
NP	Noun, Proper
VERB	
VAG	Verb, Auxiliary, Gerund
VAI	Verb, Auxiliary, Indicative
VAM	Verb, Auxiliary, Imperative
VAN	Verb, Auxiliary, Infinitive
VAP	Verb, Auxiliary, Participle
VAS	Verb, Auxiliary, Subjunctive
VMG	Verb, Main, Gerund
VMI	Verb, Main, Indicative
VMM	Verb, Main, Imperative
VMN	Verb, Main, Infinitive
VMP	Verb, Main, Participle
VMS	Verb, Main, Subjunctive
VSG	Verb, Semi-Auxiliary, Gerund
VSI	Verb, Semi-Auxiliary, Indicative
VSM	Verb, Semi-Auxiliary, Imperative
VSN	Verb, Semi-Auxiliary, Infinitive
VSP	Verb, Semi-Auxiliary, Participle
VSS	Verb, Semi-Auxiliary, Subjunctive
ADJECTIVE	
AO	Adjective, Ordinal
AQ	Adjective, Qualifier
AQP	Adjective, Qualifier and Past Participle
ADVERB	
RG	Adverb, General
RN	Adverb, Negative
PRONOUN	
P0	Pronoun, Clitic
PD	Pronoun, Demonstrative
PE	Pronoun, Exclamatory
PI	Pronoun, Indefinite
PN	Pronoun, Numeral
PP	Pronoun, Personal
PR	Pronoun, Relative
PT	Pronoun, Interrogative
PX	Pronoun, Possessive

Table 7: PoS tag set for Spanish and Catalan (1/3)

Type	**Description**
	ADPOSITON
SP	Adposition, Preposition
	CONJUNCTION
CC	Conjunction, Coordinate
CS	Conjunction, Subordinative
	DETERMINER
DA	Determiner, Article
DD	Determiner, Demonstrative
DE	Determiner, Exclamatory
DI	Determiner, Indefinite
DN	Determiner, Numeral
DP	Determiner, Possessive
DT	Determiner, Interrogative
	INTERJECTION
I	Interjection
	DATE TIMES
W	Date Times
	UNKNOWN
X	Unknown
	ABBREVIATION
Y	Abbreviation
	NUMBERS
Z	Figures
Zm	Currency
Zp	Percentage

Table 8: PoS tag set for Spanish and Catalan (2/3)

Type	Description
PUNCTUATION	
Faa	Fat Punctuation, !
Fc	Punctuation, ,
Fd	Punctuation, :
Fe	Punctuation, ``
Fg	Punctuation, -
Fh	Punctuation, /
Fia	Punctuation,
Fit	Punctuation, ?
Fp	Punctuation, .
Fpa	Punctuation, (
Fpt	Punctuation,)
Fs	Punctuation, ...
Fx	Punctuation, ;
Fz	Punctuation, other than those

COARSE TAGS	
A	Adjectives
C	Conjunctions
D	Determiners
F	Punctuation
I	Interjections
N	Nouns
P	Pronouns
S	Adpositions
V	Verbs
VA	Auxiliary Verbs
VS	Semi-Auxiliary Verbs
VM	Main Verbs

Table 9: PoS tag set for Spanish and Catalan (3/3)

Type	**Description**
ADJ	Adjective
ADJWH	Adjective
ADV	Adverb
ADVWH	Adverb
CC	Coordinating Conjunction
CLO	Weak Clitic Pronoun
CLR	Weak Clitic Pronoun
CLS	Weak Clitic Pronoun
CS	Subordinating Conjunction
DET	Determiner
ET	Foreign Word
I	Interjection
NC	Common Noun
NPP	Proper Noun
P	Preposition
P+D	Preposition and Determiner
P+PRO	Preposition and Pronoun
PONCT	Punctuation mark: , : . ” -LRB- -RRB-
PREF	Prefix
PRO	Strong Pronoun
PROREL	Relative Pronoun
V	Verb
VIMP	Verb
VINF	Verb
VPP	Verb
VPR	Verb
VS	Verb

Table 10: PoS tag set for French

Type	**Description**
PUNCTUATION	
$(	other punctuation (within the sentence)
$,	Punctuation: comma
$.	Punctuation: end of sentence
COARSE TAGS	
ADJA	Attributive adjective
ADJD	Adverbial or predicative adjective
ADV	Adverb
APPO	Postposition
APPR	Prepositions and left parts of circumpositions
APPRART	Prepositions with articles
APZR	Circumpositions, right parts
ART	Articles
CARD	Cardinal numbers
FM	Foreing words
ITJ	Interjections
KOKOM	Comparison particle ('wie'), without sentence
KON	Coordinating conjunctions
KOUI	Subordinating conjunctions with 'zu' (to) and infinitive
KOUS	Subordinating conjunctions
NE	Proper name
NN	Noun

Table 11: PoS tag set for German (1/2)

Type	**Description**
PUNCTUATION	
PDAT	Attributive demonstrative pronouns
PDS	Substitute demonstrative pronouns
PIAT	Attributive indefinit pronoun without determiner
PIDAT	Attributive indefinit pronoun with determiner
PIS	Substitute indefinit pronoun
PPER	Irreflexive personal pronoun
PPOSAT	Attributive possesive pronoun
PPOSS	Substitute possesive pronoun
PRELAT	Attributive relative pronoun
PRELS	Substitute relative pronoun
PRF	Reflexive personal pronoun
PROAV	Pronominal adverb
PTKA	Particles next to adjectives or adverbs
PTKANT	Answer particle
PTKNEG	Negation particle
PTKVZ	separated sentences
PTKZU	’zu’ (to) before infinitive
PWAT	Attributive interrogative pronouns
PWAV	Adverbial interrogative or relative pronouns
PWS	Substitute interrogative pronouns
TRUNC	Compositions of first terms
VAFIN	Finite of an auxiliar verb
VAIMP	Imperative of an auxiliar verb
VAINF	Infinitive of an auxiliar verb
VAPP	Participle of an auxiliar verb
VMFIN	Finite of modal verbs forms
VMINF	Infinitive of a modal
VMPP	Participle of a modal
VVFIN	Finite verb, full
VVIMP	Imperative, full
VVINF	Infinitive
VVIZU	Infinitive with ’zu’ (to)
VVPP	Past participle
XY	Non-word, special characters

Table 12: PoS tag set for German (2/2)

Type	**Description**
Det	Determiners
PreDet	Pre-determiners
PostDet	Post-determiners
NUM	Numbers
C	Clauses
I	Inflectional Phrases
V	Verb and Verb Phrases
N	Noun and Noun Phrases
NN	Noun-noun modifiers
P	Preposition and Preposition Phrases
PpSpec	Specifiers of Preposition Phrases
A	Adjective/Adverbs
Have	Verb 'to have'
Aux	Auxiliary verbs, e.g. should, will, does, ...
Be	Different forms of verb 'to be': is, am, were, be, ...
COMP	Complementizer
VBE	'to be' used as a linking verb. E.g., I am hungry
V_N	Verbs with one argument (the subject), i.e., intransitive verbs
V_N_N	Verbs with two arguments, i.e., transitive verbs
V_N_I	Verbs taking small clause as complement

Table 13: Grammatical categories provided by MINIPAR

Type	**Description**
appo	“ACME president, –appo-> P.W. Buckman”
aux	“should <-aux– resign”
be	“is <-be– sleeping”
by-subj	subject with passives
c	clausal complement “that <-c– John loves Mary”
cn	nominalized clause
comp1	first complement
desc	description
det	“the <-det ‘– hat”
gen	“Jane’s <-gen– uncle”
fc	finite complement
have	“have <-have– disappeared”
i	relationship between a C clause and its I clause
inv-aux	inverted auxiliary: “Will <-inv-aux– you stop it?”
inv-be	inverted be: “Is <-inv-be– she sleeping”
inv-have	inverted have: “Have <-inv-have– you slept”
mod	relationship between a word and its adjunct modifier
pnmod	post nominal modifier
p-spec	specifier of prepositional phrases
pcomp-c	clausal complement of prepositions
pcomp-n	nominal complement of prepositions
post	post determiner
pre	pre determiner
pred	predicate of a clause
rel	relative clause
obj	object of verbs
obj2	second object of ditransitive verbs
s	surface subject
sc	sentential complement
subj	subject of verbs
vrel	passive verb modifier of nouns
wha, whn, whp	wh-elements at C-spec positions (a|n|p)

Table 14: Grammatical relationships provided by MINIPAR

Type	**Description**
Clause Level	
S	Simple declarative clause
SBAR	Clause introduced by a (possibly empty) subordinating conjunction
SBARQ	Direct question introduced by a wh-word or a wh-phrase
SINV	Inverted declarative sentence, i.e. one in which the subject follows the tensed verb or modal
SQ	Inverted yes/no question, or main clause of a wh-question, following the wh-phrase in SBARQ
Phrase Level	
ADJP	Adjective Phrase
ADVP	Adverb Phrase
CONJP	Conjunction Phrase
FRAG	Fragment
INTJ	Interjection
LST	List marker
NAC	Not a Constituent; used to show the scope of certain prenominal modifiers within a NP
NP	Noun Phrase
NX	Used within certain complex NPs to mark the head of the NP
PP	Prepositional Phrase
PRN	Parenthetical
PRT	Particle. Category for words that should be tagged RP
QP	Quantifier Phrase (i.e. complex measure/amount phrase); used within NP
RRC	Reduced Relative Clause
UCP	Unlike Coordinated Phrase
VP	Verb Phrase
WHADJP	Wh-adjective Phrase
WHAVP	Wh-adverb Phrase
WHNP	Wh-noun Phrase
WHPP	Wh-prepositional Phrase
X	Unknown, uncertain, or unbracketable

Table 15: Clause/phrase level tag set for English

Type	**Description**
AP	adjectival phrases
AdP	adverbial phrases
NP	noun phrases
PP	prepositional phrases
VN	verbal nucleus
VPinf	infinitive clauses
VPpart	nonfinite clauses
SENT	sentences
Sint, Srel, Ssub	finite clauses

Table 16: Clause/phrase level tag set for French

Type	**Description**
AA	superlative phrase with ”am”
AP	adjektive phrase
AVP	adverbial phrase
CAC	coordinated adposition
CAP	coordinated adjektive phrase
CAVP	coordinated adverbial phrase
CCP	coordinated complementiser
CH	chunk
CNP	coordinated noun phrase
CO	coordination
CPP	coordinated adpositional phrase
CS	coordinated sentence
CVP	coordinated verb phrase (non-finite)
CVZ	coordinated zu-marked infinitive
DL	discourse level constituent
ISU	idiosyncratis unit
MPN	multi-word proper noun
MTA	multi-token adjective
NM	multi-token number
NP	noun phrase
PP	adpositional phrase
QL	quasi-language
S	sentence
VP	verb phrase (non-finite)
VZ	zu-marked infinitive

Table 17: Clause/phrase level tag set for German

Type	**Description**
ORG	Organization
PER	Person
LOC	Location
MISC	Miscellaneous
O	Not-A-NE
DATE	Temporal expressions
NUM	Numerical expressions
ANGLE_QUANTITY	Quantities
DISTANCE_QUANTITY	
SIZE_QUANTITY	
SPEED_QUANTITY	
TEMPERATURE_QUANTITY	
WEIGHT_QUANTITY	
METHOD	Other
MONEY	
LANGUAGE	
PERCENT	
PROJECT	
SYSTEM	

Table 18: Named Entity types

Type	**Description**
A0 A1 A2 A3 A4 A5	Arguments associated with a verb predicate, defined in the PropBank Frames scheme.
AA	Causative agent
AM-ADV	Adverbial (general-purpose) adjunct
AM-CAU	Causal adjunct
AM-DIR	Directional adjunct
AM-DIS	Discourse marker
AM-EXT	Extent adjunct
AM-LOC	Locative adjunct
AM-MNR	Manner adjunct
AM-MOD	Modal adjunct
AM-NEG	Negation marker
AM-PNC	Purpose and reason adjunct
AM-PRD	Predication adjunct
AM-REC	Reciprocal adjunct
AM-TMP	Temporal adjunct

Table 19: Semantic Roles

Type	**Description**
pred	One-place properties (predicates)
rel	Two-place properties (relations)
named	Named entities
timex	Time expressions
card	Cardinal expressions
eq	Equalities

Table 20: Discourse Representation Structures. Basic DRS-conditions

Type	**Description**
or	disjunction
imp	implication
not	negation
whq	question
prop	propositional attitude

Table 21: Discourse Representation Structures. Complex DRS-conditions

Type	**Description**
Types of anaphoric information	
pro	anaphoric pronoun
def	definite description
nam	proper name
ref	reflexive pronoun
dei	deictic pronoun
Part-of-speech type	
n	noun
v	verb
a	adjective/adverb
Named Entity types	
org	organization
per	person
ttl	title
quo	quoted
loc	location
fst	first name
sur	surname
url	URL
ema	email
nam	name (when type is unknown)
Cardinality type	
eq	equal
le	less or equal
ge	greater or equal

Table 22: Discourse Representation Structures. Subtypes

Type	**Description**
topic,a,n	elliptical noun phrases
thing,n,12	used in NP quantifiers: 'something', etc.)
person,n,1	used in first-person pronouns, 'who'-questions)
event,n,1	introduced by main verbs)
group,n,1	used for plural descriptions)
reason,n,2	used in 'why'-questions)
manner,n,2	used in 'how'-questions)
proposition,n,1	arguments of propositional complement verbs)
unit_of_time,n,1	used in 'when'-questions)
location,n,1	used in 'there' insertion, 'where'-questions)
quantity,n,1	used in 'how many')
amount,n,3	used in 'how much')
degree,n,1	
age,n,1	
neuter,a,0	used in third-person pronouns: it, its)
male,a,0	used in third-person pronouns: he, his, him)
female,a,0	used in third-person pronouns: she, her)
base,v,2	
bear,v,2	

Table 23: Discourse Representation. Symbols for one-place predicates used in basic DRS conditions

Type	**Description**
rel,0	general, underspecified type of relation
loc_rel,0	locative relation
role,0	underspecified role: agent,patient,theme
member,0	used for plural descriptions
agent,0	subject
theme,0	indirect object
patient,0	semantic object, subject of passive verbs

Table 24: Discourse Representation. Symbols for two-place relations used in basic DRS conditions

5 Confidence Estimation

Confidence Estimation (CE) measures differ from standard evaluation measures (seen in Section 4) in that they do not have a set of reference translations to compare candidate translations against. Their estimates are based on the analysis of the candidate (target), source, system information and external resources. CE measures may be classified according to two complementary criteria:

- system-dependent vs. system-independent measures
- translation quality estimation vs. translation difficulty estimation measures

Asiya's initial set of CE metrics consists only of system-independent measures. In the following, we include a description. We have separated evaluation measures in two groups, respectively devoted to capture translation quality and translation difficulty.

5.1 Translation Quality

Below, we describe the set of measures based on the estimation of the translation quality (Specia et al., 2010) currently implemented in Asiya. We distinguish measures which limit to inspect the target segment (i.e., the candidate translation under evaluation) and those which inspect the source segment (i.e., the original segment to be translated) as well.

Target-based

CE-ippl This measure calculates the inverse perplexity of the target segment according to a pre-defined language model. The underlying assumption is that the likelier the sentence (according to the language model) the more fluent. Current language models have been estimated based on the latest version of the Europarl corpus (Koehn, 2003) using the SRILM Toolkit (Stolcke, 2002) (5-gram language model, applying Knesser-Ney smoothing). Two additional variants have been included:

-CE-ippl$_c$ → inverse perplexity of the target segment according to a language model calculated over sequences of base phrase chunk tags

-CE-ippl$_p$ → inverse perplexity of the target segment according to a language model calculated over sequences of part-of-speech tags

CE-logp This measure corresponds to the log probability of the target sentence according to the pre-defined language models (built as previously described). We also include two additional variants:

-CE-logp$_c$ → base phrase chunk target language model log probability

-CE-logp$_p$ → part-of-speech target language model log probability

CE-oov (Blatz et al., 2003) Out-of-vocabulaty tokens ratio. This measure is calculated as $1 - \frac{number\ of\ oov\ tokens\ in\ target}{total\ number\ of\ tokens\ in\ target}$ in the candidate translation. Currently, the base vocabulary for each of the languages included has been extracted from the Europarl corpus (Koehn, 2003).

Source/Target-based

CE-BiDictO Bilingual dictionary based overlap. This measure calculates the overlap between the words in the source segment and those in the translation candidate according to a pre-defined bilingual dictionary. This measure requires the availability of a bilingual dictionary. Currently, ASIYA resorts to the set of bilingual dictionaries available inside the Apertium MT system (Tyers et al., 2010).

CE-length Ratio between the length (in number of tokens) of the source and the target segments. The underlying assumption is that the length of correct candidate translations should be directly related to the length of the source segment. Because different language pairs have different length relations we have estimated a compression factor, α, for each language based on available parallel corpora, in our case Europarl (Koehn, 2003).

$$\text{CE-length} = \frac{min(\alpha \cdot \text{length}_{src}, \text{length}_{trg})}{max(\alpha \cdot \text{length}_{src}, \text{length}_{trg})}$$

CE-long Same as CE-length, but only shorter candidates penalize.

$$\text{CE-long} = \frac{\text{length}_{src}}{max(\alpha \cdot \text{length}_{src}, \text{length}_{trg})}$$

CE-short Same as CE-length, but only longer candidates penalize.

$$\text{CE-short} = \frac{\text{length}_{trg}}{max(\alpha \cdot \text{length}_{src}, \text{length}_{trg})}$$

CE-N This measure is similar to the CE-length measure but applied to linguistic elements instead of lexical items. It correspond to the pure ratio between the number of linguistic elements of a specific kind in the source and the target. The underlying assumption is that good translations and source segment should use a similar number of linguistic elements. Two variants are currently considered:

-CE-N$_c$ → ratio between number of base phrase chunks in source and target segments.

-CE-N$_e$ → ratio between number of named entities in source and target segments.

LeM This measure is similar to the CE-length measure but using a different technique. The Length Model measure estimates the quality likelihood of a candidate sentence by considering the "expected length" of a proper translation from the source. The measure was introduced by (Pouliquen et al., 2003) to identify document translations. We estimated its parameters over standard MT corpora, including Europarl, Newswire, Newscommentary and UN.

CE-O This measure computes overlap between source and target segments for different linguistic elements. In short, overlap is computed as the cardinality of the intersection divided into the cardinality of the union (Giménez &

Màrquez, 2010). The assumption is that good translations and source segment should use similar types of linguistic elements. Three variants of the overlap between the two sentences have been included:

-CE-O$_{c}$ → overlap over phrase chunks,

-CE-O$_{e}$ → overlap over named entities,

-CE-O$_{p}$ → overlap over part-of-speech tags.

CE-symbols This measure computes lexical overlap between symbols. The set of symbols includes punctuation marks (e.g., ‘.’, ‘,’, ‘!’, ‘?’, ‘”, ‘(’, ‘)’, ‘[’, ‘]’, ’’, ’’, ‘$’, ‘%’, ‘&’, ‘/’, ‘\’, ‘=’, ‘*’, ‘-’, ‘—’, ‘_’, ‘|’, ‘<’, ‘>’, ‘@’, ‘#’) and anything that looks like a number. The assumption is that source segment and good candidate translations should have a similar number of numbers and punctuation symbols.

5.2 Translation Difficulty

Below, we describe the set of measures based on the estimation of the translation difficulty. These measures are calculated only on the source language.

Source-based

CE-BiDictA This measure comptues bilingual-dictionary-based ambiguity. The underlying assumption is that more ambiguous words are harder to translate. This measure is computed as $\frac{1}{ambiguity(source)}$, where the ambiguity of the source is determined as the average number of translations available in a given bilingual dictionary for each n-gram in the source segment[39]. Bilingual dictionaries are borrowed from the Apertium open source project (Tyers et al., 2010).

CE-srcippl This measure calculates the inverse perplexity for the source segment according to a pre-defined language model. The assumption is that the likelier the sentence the easier to translate. Language models are built as described in the case of the CE-ippl measure. Two additional variants have been considered:

-CE-srcippl$_{c}$ → base phrase chunk source language model inverse perplexity

-CE-srcippl$_{p}$ → part-of-speech source language model inverse perplexity

CE-srclog This measure corresponds to the log probability of the source segment according to the pre-defined language models (built as previously described). We also include two additional variants:

-CE-srclogp$_{c}$ → base phrase chunk source language model log probability

-CE-srclogp$_{p}$ → part-of-speech language source model log probability

CE-srclen This measure is based on the source length and is computed as $\frac{1}{len(source)}$. The underlying assumption is that longer sentences are harder to translate.

[39] Bilingual dictionaries may contain multiwords.

CE-srcoov This measure is based on the number of out-of-vocabulary tokens in the source segment. It is calculated as $1 - \frac{number\ of\ oov\ tokens\ in\ source}{total\ number\ of\ tokens\ source}$ in the candidate translation. The underlying assumption is that the larger the number of unknown tokens the harder to translate the source sentence.

6 Learning to combine CE measures for quality pairwise ranking

As an alternative to mere uniformly-averaged combinations of combinations (ULC), we have designed and implemented an on-line learning architecture. The goal is to combine the scores conferred by different evaluation measures into a single measure of quality such that their relative contribution is adjusted based based on human feedback (i.e., from human assessments). The architecture is based on a ranking perceptron. In short, on-line learning works as follows. First, the perceptron is initialized by setting the weight of all individual measures (i.e., the features) to 0. Then, assessors are presented test cases. These consist of pairwise comparisons, i.e., a source segment and two candidate translations *a* and *b*. Assessors must tell whether translation *a* is better than *b*, worse, or equal in quality. After each feedback step we ask the perceptron to rank translations *a* and *b* based on the scalar product between individual measure scores and their current weights. If there is agreement between the perceptron and the assessor we leave the weights unchanged. Otherwise, we update them towards the human assessment.

Models are learned using the "-learn <scheme>" option:

```
Asiya.pl -learn <scheme> -assessment human_scores.csv sample.config
```

The only implemented <scheme> is the *perceptron*, which requires the human assessments file (see Section 3.2). We can adjust some parameters as the number of epochs ('-n_epochs' option, set to 100 by default), the minimum distance between human scores ('-min_dist' option, 0 by default), the proportion of training examples ('-train_prop' option, 0.8 by default).

The model created during the learning process is saved in a file by using the '-model <s>' option (by default the following path will be used './models/perceptron.mod'). The model can be used with the evaluation option (see Section 3.1).

Once learned, models are used via the "-eval model" option. Thus, for instance:

```
Asiya.pl -eval single,model -model perceptron.mod sample.config
```

will compute and print individual metric scores and the score given by the 'perceptron.mod' learned model.

7 On-line Interfaces and Web Service

The Asiya on-line interfaces provide a graphical and interactive access to Asiya intended to allow users to familiarize with its functionalities and to analyze real testbeds

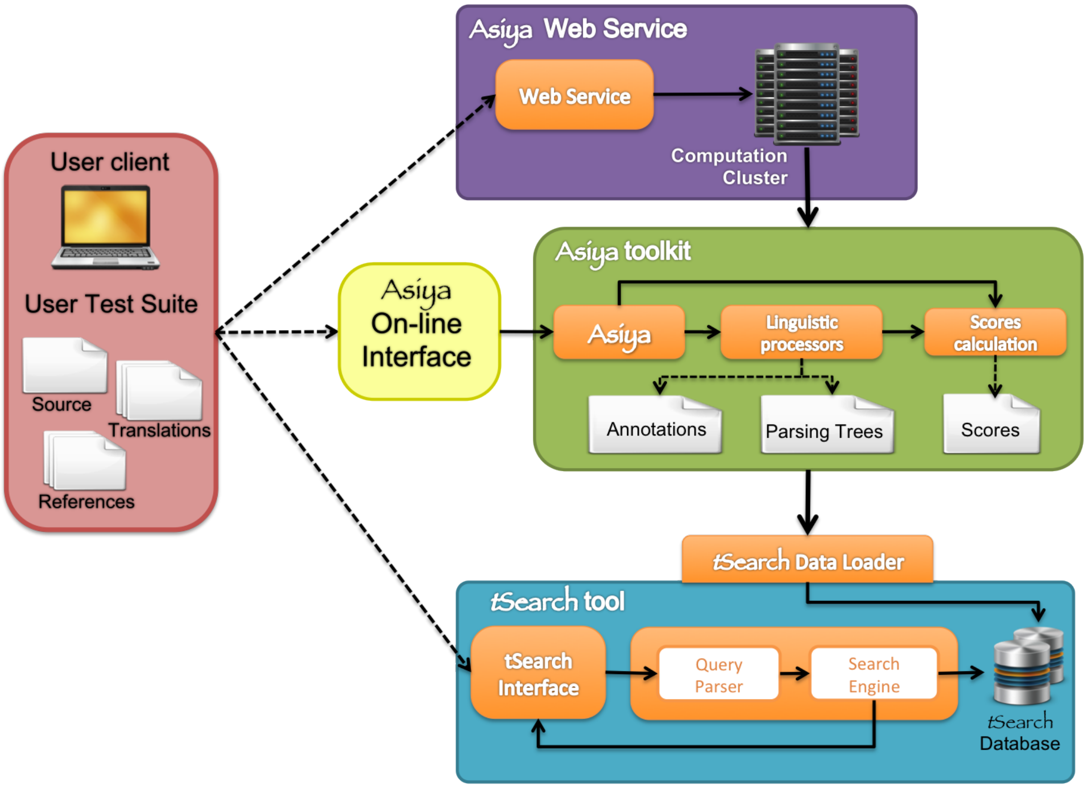


Figure 2: The Asiya Platform

in a friendly environment. Figure 2 shows a complete overview of the application architecture and its modules. It consists of three main modules that users can access independently from each other.

Although installing Asiya is not too difficult, setting additional tools up can represent a barrier to people not familiarized with the installation and configuration of software packages and libraries. The following online applications address this drawback and aimed at helping users to get familiarized with the MT evaluation tools:

1. Asiya Online Interface (Section 7.1), provides a graphical interface to access an on-line version of Asiya. This GUI favours a rapid evaluation of testbeds using just a browser and a quick inspection of the results obtained, including graphs, annotations and visualization of parse trees.
2. Asiya tSearch (Section 7.2), allows to search for output translations (of a given testbed) that match some specific criteria related to their quality (as assessed by the automatic scores). This is a complementary tool for Asiya Online Interface, intended to facilitate translation error analysis and system comparison.
3. AsiyaWS (Section 7.3), is a RESTful web service to run Asiya. This web service

allows for using Asiya from any remote client running on any platform. In the line of today's *cloud computing* services, this service is intended to facilitate the remote usage of the application without the need for downloading and locally installing all the modules.

7.1 Asiya Online Interface

The primary goal of providing graphical interfaces is to allow MT developers to analyze their systems using a friendly environment. To this end, we have set up a web application that makes possible a graphical visualization and interactive access to Asiya results (Gonzàlez et al., 2012).

The benefits of the online interface are multiple. First, it facilitates the use of the Asiya toolkit for rapid evaluation of test beds. Then, we aim at aiding the analysis of the errors produced by the MT systems by creating a significant visualization of the information related to the evaluation metrics, and also an engine able to search for translations that match some criteria related to the metric scores.

The web application can be reached at: `http://asiya.lsi.upc.edu/`.

The Asiya Online Interface allows any user to upload a test beds, obtain a large set of metric scores and then, detect and analyze the errors of the systems, just using an Internet browser.

The interface consists of a simple web form to supply the data required to run Asiya, and then, it offers several views that display the results in friendly and flexible ways such as interactive score tables, graphical parsing trees in SVG format and interactive sentences holding the linguistic annotations captured during the computation of the metrics.

The website that hosts the Asiya Online Interface includes a tarball with sample input data. A video demo showing the main functionalities of the interface and how to use it is available at the website.

7.2 Asiya tSearch

The Asiya tSearch interface (Gonzàlez et al., 2013) has been built on top of Asiya. It offers a graphical search module that allows to retrieve from a concrete testbed all translation examples that satisfy certain properties on the systems' evaluation scores, or on the linguistic information used to calculate the evaluation measures.

A video demo is available at the website. It contains a brief explanation about the most important features described in this section. Furthermore, you can find the tSearch online interface user manual in Appendix B.

The tSearch architecture consists of three components: the web-based interface, the storage system based on *N*oSQL technology and the tSearch core, composed of a query parser and a search engine.

The amount of data generated by Asiya can be very large for test sets with thousands of sentences. In order to handle the high volume of information, we decided to use the Apache Cassandra database[40], a *N*oSQL solution that deals successfully with this problem.

[40] `http://cassandra.apache.org/`

The databases are fed through the *tSearch Data Loader* API used by ASIYA. At run-time, during the calculation of the measures, ASIYA *inserts* all the information being calculated (metrics and parses) and a number of precalculated variables (e.g., average, mean and percentiles). These operations are made in parallel, which makes the overhead of filling the database marginal.

The query parser module is one of the key ingredients in the tSEARCH application because it determines the query grammar and the allowed operations, and it provides a parsing method to analyze any query and produce a machine-readable version of its semantics. It is also necessary in order to validate the query.

There are several types of queries, depending on the operations used: arithmetic comparisons, statistical functions (e.g., average, quartiles), range of values, linguistic elements and logical operators. Furthermore, the queries can be applied at segment-, document- and/or system-level, and it is even possible to create any group of systems or metrics. This is useful, for instance, in order to limit the search to certain type of systems (e.g., rule-based vs. statistical) and specific metrics (e.g., lexical vs. syntactic). All possible query types are described in the user manual in Appendix B.

In the tSEARCH interface, one can see a tools icon on the right of the search box. It shows the toolbar with all available metrics, functions and operations. The search box allows to query the database using the query language.

After typing a query, the user can navigate the results using three different views that organize them according to the user preferences: 1) *All segments* shows all segments and metrics mentioned in the query, the segments can be sorted by the score, in ascendent or descendent order, just tapping on the metric name; 2) *Grouped by system* groups the segments by system and, for each system, by document; 3) *Grouped by segment* displays the segment organization, which allows an easy comparison between several translations. Each group contains all the information related to a segment number, such as the source and the reference sentences along with the candidate translations that matched the query.

All output data obtained during the search can be exported as an XML file. It is possible to export all segments, or the results structured *by system*, *by segment*, or more specific information from the views.

7.3 AsiyaWS

The ASIYAWS is intended to facilitate the remote usage of ASIYA without the need for downloading and locally installing all the modules. It allows to access the application from any remote client running on any platform or developed using other tools. Thereby, the service eases the integration of ASIYA as part of other applications that may be working on heterogeneous platforms.

The ASIYAWS follows a RESTful architecture, and therefore it provides stateless interactions. The server side includes a mechanism to manage the user requests and keep the authoring of the data. Also, ASIYA is computationally demanding. In order to handle big dataset and multiple ASIYA executions, the service makes use of a GRID cluster by means of a new protocol that submits jobs remotely to the cluster, and the engine to manage the ASIYAWS queue.

The service information can be found in the ASIYA website: `http://asiya.lsi.`

upc.edu/. A simple HTTP client and sample data showing how to access the service can be downloaded also from the site.

8 Ongoing and Future Steps

The current development of the Asiya toolkit goes in two main directions. First, we are augmenting the metric repository and associated procedures. We are incorporating new metrics and we are porting linguistic metrics to other languages. We have recently incorporated other linguistic processors as the language-independent MALT dependency parser (Nivre & Hall, 2005). We currently support German, but the parser has been trained on a variety of languages. We also plan to design and implement a mechanism so users can easily incorporate their own metrics.

Other more complex translation difficulty measures, based on alignments are also being explored now and planned to be incorporated to Asiya in the future.

Recently, we have included a supervised learning process, based on a ranking perceptron, to combine different measures of quality adjusting their contribution on the grounds of human assessments (described in Section 6). In the future, we plan to experiment with this architecture and study several metric combination schemes and alternative meta-evaluation criteria.

The second direction refers to the use of Asiyaonline and the construction of visual interfaces. We have released the two web applications (http://asiya.lsi.upc.edu/) for monitoring the whole development cycle. This application allows system and metric developers to upload their test suites and perform error analysis, automatic and manual evaluation, and meta-evaluation, using their Internet browsers. Future releases will include visualization of linguistic information, additional interaction funcionalities, and the automation of the error discovery and report generation.

We have also released the first version of a web service that allows to submit Asiyarequests remotely. The first release and a simple HTML client are already available.

References

Amigó, E., Giménez, J., Gonzalo, J., & Màrquez, L. (2006). MT Evaluation: Human-Like vs. Human Acceptable. *Proceedings of the Joint 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL)* (pp. 17–24).

Amigó, E., Gonzalo, J., Peñas, A., & Verdejo, F. (2005). QARLA: a Framework for the Evaluation of Automatic Summarization. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)* (pp. 280–289).

Banerjee, S., & Lavie, A. (2005). METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. *Proceedings of ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization.*

Blatz, J., Fitzgerald, E., Foster, G., Gandrabur, S., Goutte, C., Kulesza, A., Sanchis, A., & Ueffing, N. (2003). *Confidence estimation for machine translation. Final Report of Johns Hopkins 2003 Summer Workshop on Speech and Language Engineering* (Technical Report). Johns Hopkins University.

Brants, S., Dipper, S., Hansen, S., Lezius, W., & Smith, G. (2002). The TIGER treebank. *Proceedings of the Workshop on Treebanks and Linguistic Theories.* Sozopol.

Callison-Burch, C., Koehn, P., Monz, C., Peterson, K., Przybocki, M., & Zaidan, O. (2010). Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR* (pp. 17–53). Revised August 2010.

Candito, M., Crabbé, B., & Denis, P. (2010a). Statistical French dependency parsing: treebank conversion and first results. *The seventh international conference on Language Resources and Evaluation (LREC).* Valletta, Malta.

Candito, M., Nivre, J., Denis, P., & Anguiano, E. H. (2010b). Benchmarking of Statistical Dependency Parsers for French. *COLING 2010: Poster volume* (pp. 108—-116). Beijing, China.

Carreras, X., Chao, I., Padró, L., & Padró, M. (2004). FreeLing: An Open-Source Suite of Language Analyzers. *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC)* (pp. 239–242).

Charniak, E., & Johnson, M. (2005). Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL).*

Curran, J., Clark, S., & Bos, J. (2007). Linguistically motivated large-scale nlp with c&c and boxer. *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions* (pp. 33–36).

Denis, P., & Sagot, B. (2009). Coupling an annotated corpus and a morphosyntactic lexicon for state-of-the-art pos tagging with less human effort. *The Pacific Asia Conference on Language, Information and Computation (PACLIC 23).* Hong Kong, China.

Denkowski, M., & Lavie, A. (2010). Meteor-next and the meteor paraphrase tables: Improved evaluation support for five target languages. *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR* (pp. 339–342).

Doddington, G. (2002). Automatic Evaluation of Machine Translation Quality Using N-gram Co-Occurrence Statistics. *Proceedings of the 2nd International Conference on Human Language Technology* (pp. 138–145).

Efron, B., & Tibshirani, R. (1986). Bootstrap Methods for Standard Errors, Confidence Intervals, and Other Measures of Statistical Accuracy. *Statistical Science*, *1*, 54–77.

Fisher, R. A. (1921). On the 'probable error' of a coefficient of correlation deduced from a small sample. *Metron*, *11*, 3—32.

Fisher, R. A. (1924). On a Distribution Yielding the Error Functions of Several Well Known Statistics. *Proceedings of the International Congress of Mathematics* (pp. 805–813).

Gabrilovich, E., & Markovitch, S. (2007). Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. *Proceedings of the 20th International Joint Conference on Artificial Intelligence* (pp. 1606–1611). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Giménez, J., & Amigó, E. (2006). IQMT: A Framework for Automatic Machine Translation Evaluation. *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)* (pp. 685–690).

Giménez, J., & Màrquez, L. (2004a). Fast and Accurate Part-of-Speech Tagging: The SVM Approach Revisited. *Recent Advances in Natural Language Processing III* (pp. 153–162). Amsterdam: John Benjamin Publishers. ISBN 90-272-4774-9.

Giménez, J., & Màrquez, L. (2004b). SVMTool: A general POS tagger generator based on Support Vector Machines. *Proceedings of 4th International Conference on Language Resources and Evaluation (LREC)* (pp. 43–46).

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

Giménez, J., & Màrquez, L. (2010). Linguistic measures for automatic machine translation evaluation. *Machine Translation*, *24*, 209–240.

Gonzàlez, M., Giménez, J., & Màrquez, L. (2012). A graphical interface for mt evaluation and error analysis. *Annual Meeting of the Association for Computational Linguistics (ACL). System Demonstration.* Jeju, South Korea.

Gonzàlez, M., Mascarell, L., & Màrquez, L. (2013). tSearch: Flexible and Fast Search over Automatic translation for Improved Quality/Error Analysis. *Proc. Annual Meeting of the Association for Computational Linguistics (ACL). System Demonstration.* Sofia, Bulgaria.

Hall, J., & Nivre, J. (2008). A Dependency-Driven Parser for German Dependency and Constituency Representations. *ACL Workshop on Parsing German (PaGe08).* Columbus, Ohio, USA.

Kendall, M. (1955). *Rank Correlation Methods.* Hafner Publishing Co.

King, M., & Falkedal, K. (1990). Using Test Suites in Evaluation of MT Systems. *Proceedings of the 13th International Conference on Computational Linguistics (COLING)* (pp. 211–216).

Koehn, P. (2003). *Europarl: A Multilingual Corpus for Evaluation of Machine Translation* (Technical Report). `http://people.csail.mit.edu/people/koehn/publications/europarl/`.

Koehn, P. (2004). Statistical Significance Tests for Machine Translation Evaluation. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 388–395).

Levenshtein, V. I. (1966). Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, *8*, 707–710.

Lin, C.-Y., & Och, F. J. (2004a). Automatic Evaluation of Machine Translation Quality Using Longest Common Subsequence and Skip-Bigram Statics. *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL).*

Lin, C.-Y., & Och, F. J. (2004b). ORANGE: a Method for Evaluating Automatic Evaluation Metrics for Machine Translation. *Proceedings of the 20th International Conference on Computational Linguistics (COLING).*

Lin, D. (1998). Dependency-based Evaluation of MINIPAR. *Proceedings of the Workshop on the Evaluation of Parsing Systems.*

Liu, D., & Gildea, D. (2005). Syntactic Features for Evaluation of Machine Translation. *Proceedings of ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization* (pp. 25–32).

Lluís, X., Carreras, X., & Màrquez, L. (2013). Joint Arc-factored Parsing of Syntactic and Semantic Dependencies. *Transactions of the Association for Computational Linguistics*, *1*, 219–230.

Marcus, M. P., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, *19*, 313–330.

Màrquez, L., Surdeanu, M., Comas, P., & Turmo, J. (2005). Robust Combination Strategy for Semantic Role Labeling. *Proceedings of the Joint Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT-EMNLP).*

Melamed, I. D., Green, R., & Turian, J. P. (2003). Precision and Recall of Machine Translation. *Proceedings of the Joint Conference on Human Language Technology and the North American Chapter of the Association for Computational Linguistics (HLT-NAACL).*

Navarro, B., Civit, M., Martí, M. A., Marcos, R., & Fernández, B. (2003). Syntactic, Semantic and Pragmatic Annotation in Cast3LB. *Proceedings of SProLaC* (pp. 59–68).

Nießen, S., Och, F. J., Leusch, G., & Ney, H. (2000). An Evaluation Tool for Machine Translation: Fast Evaluation for MT Research. *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC).*

Nivre, J., & Hall, J. (2005). Maltparser: A language-independent system for data-driven dependency parsing. *In Proc. of the Fourth Workshop on Treebanks and Linguistic Theories* (pp. 13–95).

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., & Marsi, E. (2007). Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, *13*, 95–135.

Palmer, M., Gildea, D., & Kingsbury, P. (2005). The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, *31*, 71–106.

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2001). *Bleu: a method for automatic evaluation of machine translation, RC22176* (Technical Report). IBM T.J. Watson Research Center.

Pearson, K. (1914). *The life, letters and labours of Francis Galton.* (3 volumes: 1914, 1924, 1930).

Petrov, S., Barrett, L., Thibaux, R., & Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. *21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics* (pp. 433–440). Stroudsburg, PA, USA: Association for Computational Linguistics.

Petrov, S., & Klein, D. (2007). Improved Inference for Unlexicalized Parsing. *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference* (pp. 404–411). Association for Computational Linguistics.

Pouliquen, B., Steinberger, R., & Ignat, C. (2003). Automatic Identification of Document Translations in Large Multilingual Document Collections. *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP-2003)* (pp. 401–408). Borovets, Bulgaria.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., & Makhoul, J. (2006). A Study of Translation Edit Rate with Targeted Human Annotation. *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas (AMTA)* (pp. 223–231).

Snover, M., Madnani, N., Dorr, B., & Schwartz, R. (2009). Fluency, adequacy, or HTER? Exploring different human judgments with a tunable MT metric. *Proceedings of the Fourth Workshop on Statistical Machine Translation* (pp. 259–268).

Spearman, C. (1904). The Proof and Measurement of Association Between Two Rings. *American Journal of Psychology*, *15*, 72–101.

Specia, L., Raj, D., & Turchi, M. (2010). Machine translation evaluation versus quality estimation. *Machine Translation*, *24*, 39–50.

Stolcke, A. (2002). SRILM - An Extensible Language Modeling Toolkit. *Proceedings of ICSLP.*

Surdeanu, M., & Turmo, J. (2005). Semantic Role Labeling Using Complete Syntactic Analysis. *Proceedings of CoNLL Shared Task.*

Surdeanu, M., Turmo, J., & Comelles, E. (2005). Named Entity Recognition from Spontaneous Open-Domain Speech. *Proceedings of the 9th International Conference on Speech Communication and Technology (Interspeech).*

Taulé, M., Martí, M. A., & Recasens, M. (2008). Ancora: Multilevel annotated corpora for catalan and spanish. *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08).* Marrakech, Morocco: European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2008/.

Telljohann, H., Hinrichs, E., Kübler, S., Kübler, R., & Tübingen, U. (2004). The tüba-d/z treebank: Annotating german with a context-free backbone. *In Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004* (pp. 2229–2235).

Tillmann, C., Vogel, S., Ney, H., Zubiaga, A., & Sawaf, H. (1997). Accelerated DP based Search for Statistical Translation. *Proceedings of European Conference on Speech Communication and Technology.*

Tyers, F. M., Sánchez-Martínez, F., Ortiz-Rojas, S., & Forcada, M. L. (2010). Free/open-source resources in the Apertium platform for machine translation research and development. *The Prague Bulletin of Mathematical Linguistics*, 67–76.

A Glossary of Evaluation Measures

WER word error rate

PER position-independent word error rate

TER$_{\mathbf{[p|pA|base]}}$ variants of translation edit rate

ALGN ratio of shared alignments between source, reference and candidate

BLEU smoothed 4-gram BLEU score

NIST default 5-gram NIST score

ROUGE$_{\boldsymbol{L|S\star|SU\star|W}}$ variants of ROUGE

GTM$_{\mathbf{1|2|3}}$ variants of GTM rewarding longer matchings

METEOR$_{\mathbf{ex|st|sy|pa}}$ variants of METEOR

$\boldsymbol{O_l}$ lexical overlap

$\boldsymbol{P_l}$ lexical precision

$\boldsymbol{R_l}$ lexical recall

$\boldsymbol{F_l}$ lexical F-measure

NGRAM cosine and Jaccard similarities on character and token $n-$grams.

SP-$\boldsymbol{O_p(\star)}$ average lexical overlap over parts of speech

SP-$\boldsymbol{O_c(\star)}$ average lexical overlap over chunk types

SP-NIST$_{\boldsymbol{l|p|c|iob}}$ NIST score over sequences of: lemmas, parts of speech, phrase chunks, and chunk IOB labels

DP-HWCM$_{w|c|r}$ head-word chain matching over word forms, grammatical categories, or grammatical relations

DP-$\boldsymbol{O_{l|c|r}(\star)}$ average overlap between lexical items according to their tree level, grammatical category, or grammatical relationship

CP-$\boldsymbol{O_{p|c}(\star)}$ average lexical overlap over parts of speech, or constituents

CP-STM$_{\boldsymbol{l}}$ variants of Syntactic Tree Matching for different depths

NE-$\boldsymbol{O_e(\star)}$ average lexical overlap over named entities

NE-$\boldsymbol{M_e(\star)}$ average lexical matching over named entities

ESA explicit semantic analysis using Wikipedia documents

SR-$\boldsymbol{O_{r[v]}(\star)}$ average lexical overlap over semantic roles

SR-$\boldsymbol{M_{r[v]}(\star)}$ average lexical matching over semantic roles

SR-$\boldsymbol{O_{r[v]}}$ average role overlap

DR-STM$_{\boldsymbol{l}}$ variants of Semantic Tree Matching for different depths

DR-$\boldsymbol{O_r(\star)}$ average lexical overlap over discourse representations

DR-$\boldsymbol{O_{rp}(\star)}$ average part-of-speech overlap over discourse representations

CE-*ippl*$_{\boldsymbol{[c|p]}}$ candidate language model inverse perplexity over lexical forms, base phrase chunks or parts of speech candidate phrase

CE-*logp*$_{[c|p]}$ candidate language model log probabililty over lexical forms, base phrase chunks or parts of speech

CE-*oov* candidate language model out-of-vocabulary tokens ratio

CE-*BiDictO* source/candicate bilingual dictionary based overlap

CE-*length* source/candidate length ratio

CE-*long* source/candidate length ratio where only shorter candidates penalize

CE-*short* source/candidate length ratio where only longer candidates penalize

LeM candidate expected length

CE-*N*$_{c|e}$ source/candidate phrase chunk and named entity ratio

CE-*O*$_{c|e|p}$ source/candidate phrase chunk, named entity and PoS overlap

CE-*symbols* source/candidate symbol overlap

CE-*BiDictA* bilingual dictionary-based source ambiguity

CE-*scrippl*$_{[c|p]}$ source language model inverse perplexity over lexical forms, base phrase chunks or parts of speech candidate phrase

CE-*srclen* 1 / source length

CE-*srclogp*$_{[c|p]}$ source language model log probabililty over lexical forms, base phrase chunks or parts of speech

CE-*srcoov* source language model out-of-vocabulary tokens ratio

B tSearch User Manual

B.1 Getting started

Let us introduce the user manual of tSEARCH[41], a web-based application that aids the error analysis stage of machine translation development facilitating the qualitative analysis of translation quality. The tSEARCH ONLINE INTERFACE is accessible at `http://asiya.lsi.upc.edu/demo/` where you can find to ways to access it. The first one consists of evaluating a testbed with ASIYA and once the evaluation is completed, tSEARCH appears as one of the tools that the user can run. However, if you have the data already evaluated by ASIYA, the second option allows you to upload the compressed folder that contains the ASIYA evaluation output and start using tSEARCH.

B.1.1 Getting to know tSearch

The Figure 3 describes some of the features available in tSEARCH ONLINE INTERFACE:

1. **Toolbar**: use the toolbar to find all metrics, systems and documents, operate with groups, and view examples and select the functions and operations available.
2. **Output area**: this area displays the results of your query.
3. **Query input**: use this input box to write yout query.
4. **View tabs**: navigate through the different organization views: all segments, by system or by segment.
5. **Info panel**: gives you additional information related to the query such as groups of metrics, systems and documents, and the actual values used for the statistical functions such as the `MIN`, `MAX`, `AVG`, `MEDIAN`, `TH()`, `PERC()` or `Q()`.

Click the following icons to...
create-edit groups

B.1.2 Views

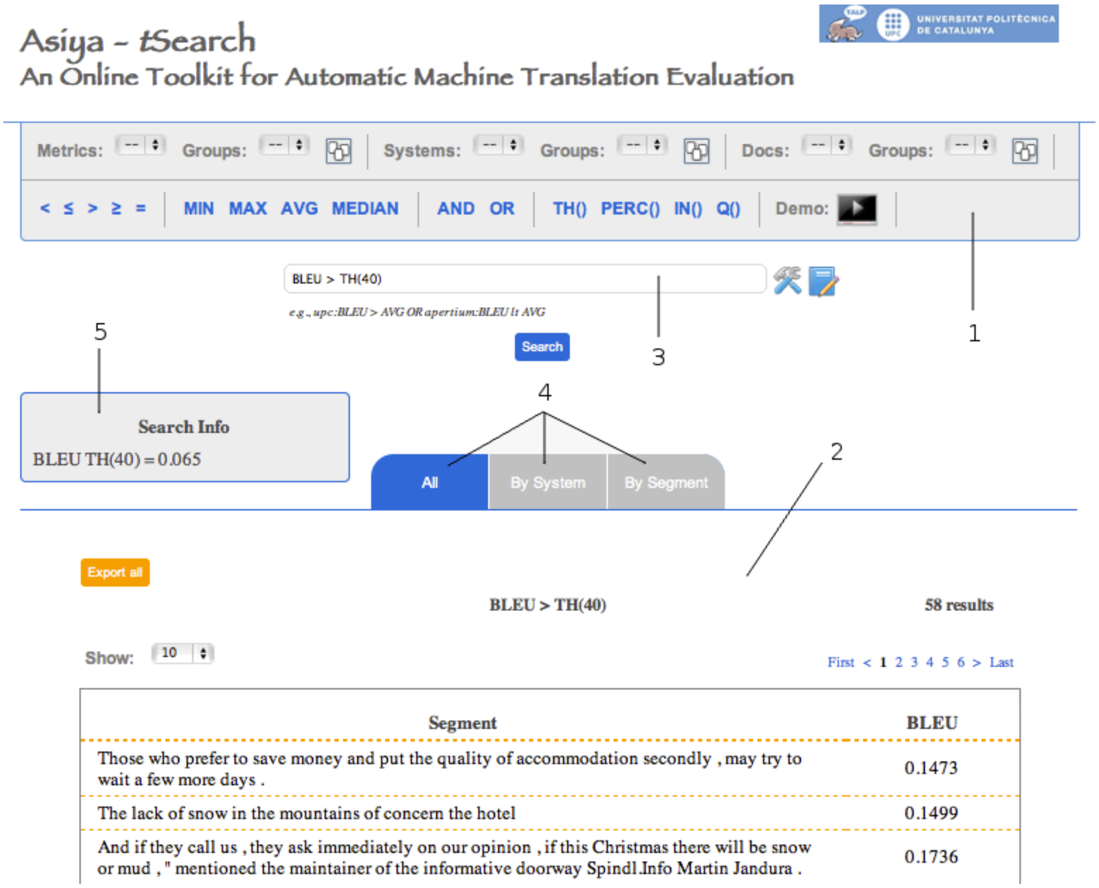
tSEARCH lets you navigate the results of the search accross all the automatic translations selected and their evaluations. Three different views organize the segments according to the user preferences:

All: this view shows all segments and the scores for the metrics involved in the query.

By system: it groups the segments by system name and, for each system, by document name.

By segment: this view offers the segment organization, which facilitates the comparison between several translations, the reference and the source for each segment.

[41] There is also a video tutorial available at `www.youtube.com/watch?v=GBEnmOsKmT4&vq=hd720`.



Search Info

BLEU TH(40) = 0.065

BLEU > TH(40)

58 results

Segment	BLEU
Those who prefer to save money and put the quality of accommodation secondly , may try to wait a few more days .	0.1473
The lack of snow in the mountains of concern the hotel	0.1499
And if they call us , they ask immediately on our opinion , if this Christmas there will be snow or mud , " mentioned the maintainer of the informative doorway Spindl.Info Martin Jandura .	0.1736

Figure 3: Getting to know tSearch

Toolbar

Show and hide the toolbar.

Create or edit a group of metrics, systems or documents.

See the video manual.

Other

Export as an XML file the partial results depending on the current view.

Show and hide the examples window.

B.2 Create and Edit Groups

The interface allows to create groups of systems, documents and/or metrics. The purpose of this feature is to facilitate the comparison between types of systems (e.g., stadistical vs. rule-based) or metrics (e.g., lexical vs. syntactic) or even, groups of documents that belong to different domains.

The following steps describe how to create a new group:

1. From the toolbar, click the Groups button . The button is in three different blocks in order to distinguish between metrics, systems and documents.
2. Write the name of your new group at the *Group name* field.
3. Chose form the left panel what do you want to include in your group passing them to the right panel.
4. Click the create button.

Later on, if you want to edit an existing group...

1. From the toolbar, click the Groups button . The button is in three different blocks in order to distinguish between metrics, systems and documents.
2. Select from the *Groups* list the one you want to edit and then, its name and elements are displayed.
3. Edit the values you want to change, i.e., the name of the group or the elements, passing to the left panel the ones you want to eliminate from the group or passing to the right panel the ones you want to include.
4. Click the update button.

B.3 Let's query

There are several types of queries, depending on the operations used: arithmetic comparisons, statistical functions (e.g., average, quartiles), range of values, linguistic elements and logical operators. Table 25 lists some of the most representative queries of each group.

Regarding metric-based queries, the arithmetic comparison queries let you obtain all segments scored above/below a value for a concrete metric. Such value can be a real number or also a statistical variable such as minimum `MIN`, maximum `MAX`, median `MEDIAN`, average `AVG` or the threshold function `TH()`. We have also implemented statistical functions such as the quartile function `Q()` or the percentile `PERC(n,M)`, which returns all the segments with a score in the n^{th} part, when the range of scores is divided in M parts of equal size. The last query in this group refers to the system comparison. Thus, given an evaluation measure, it allows comparing its score between systems.

Concerning linguistic-based queries, we have implemented queries that match N-grams of lemmas `lemma`, parts-of-speech `pos` and items of shallow `SP` or constituent parsing `CP`, dependency relations `DP`, semantic roles `SR` and named entities `NE`. The `DP` function allows specifying a structure composition criterion (i.e., the categories of two words and their dependency relationship) and even a chain of relations. The `SR` function obtains the segments that match a verb and its list of arguments. The use of the asterisk symbol substitutes any value, e.g., `LE[CP(NP,*,PP),DP(*,*,V)]`. However, when combined with semantic roles, one asterisk substitutes any verb that has all the arguments specified, e.g., `LE[SR(*,A0,A1)]`, whereas two asterisks in a row allow arguments to belong to different verbs in the same sentence.

The above queries are applied at segment level. However, applying them at system and document-level is as easy as specifying the system and/or document names, e.g., `(upc:BLEU > AVG) AND (upc:LE[DP(*,nsubj,*)])`. In addition, there is also the possiblity to use a group of metrics, systems and/or documents instead, e.g., `(LEX:RBMT > AVG) AND (RBMT:LE[DP(*,nsubj,*)])`, where `RBMT` is a group of rule-based systems and `LEX` is a group of lexical metrics defined and created by the user.

Metric-based Queries	**Arithmetic Comparison**	BLEU > 0.4 BLEU > TH(40) BLEU le MEDIAN
	Range of Values	BLEU IN [0.2, 0.3) BLEU IN Q(4) BLEU IN PERC(2,10) BLEU IN (TH(20),TH(40))
	Sistem comparison	upc:BLEU > dfki:BLEU
LE-based Queries	**N-grams**	LE[SP(NN,*,VBZ)] LE[CP(NP,PP)] LE[lemma(be),CP(VP,PP)] LE[pos(DT,JJ,*)] LE[NE(ORG)]
	Semantic Roles	LE[SR(ask,A1,AM-TMP)] LE[SR(*,A1,AM-TMP)] LE[SR(**,A1,AM-TMP)]
	Dependency Relationships	LE[DP(N,nsubj,V)] LE[DP(N,nsubj,V,dep,V)] LE[DP(*,nsubj,*)]
Group Creation and Complex Queries	**Logical Composition**	BLEU > AVG AND LE[DP(N,nsubj,V)] LEX = {BLEU,NIST} SYN = {DP-Or(*),SP-Op(*)} SMT = {bing,google} (SMT:LEX > AVG OR apertium:LEX < AVG) AND (SMT:SYN < AVG OR apertium:SYN > AVG)

Table 25: tSearch query examples