ACT Manual

Najeh Hajlaoui <Najeh.Hajlaoui@idiap.ch>
Idiap Research Institute
P.O. Box 592
CH-1920 Martigny
Switzerland
2013

Current Version: ACT 1.7

Please note: when citing ACT in your own publications, please be sure to include the version number and what configuration of ACT (Without Alignment, Without Training, With Training) you used.

For more information, bug reports, fixes, contact Najeh.Hajlaoui@idiap.ch or Najeh.Hajlaoui@gmail.com

Recommended Reading and Citing: Hajlaoui N. and Popescu-Belis A. *Assessing the Accuracy of Discourse Connective Translations: Validation of an Automatic Metric,* in: 14th International Conference on Intelligent Text Processing and Computational Linguistics, University of the Aegean, Samos, Greece, March 24-30, 2013, 12 p.

1 Introduction

ACT for Accuracy of Connective Translation is a reference-based metric to measure the accuracy of discourse connective translation, mainly for statistical machine translation systems. The metric relies on automatic word-level alignment (using GIZA++, not included in ACT) between a source sentence and respectively the reference and candidate translations, along with other heuristics for comparing translations of discourse connectives. Using a dictionary of equivalents, the translations are scored automatically, or, for more accuracy, semi-automatically. The accuracy of the ACT metric was assessed by human judges on sample data for English/French, English/Arabic, English/Italian and English/German translations; the ACT scores are within 2-4% of human scores.

As of the time of this writing, ACT focuses on a small number of English connectives (although, though, even though, while, meanwhile, since, yet, however), and evaluates their translation into French, Italian, German and Arabic. But, it is possible to port the ACT metric to other language pairs and to other linguistic phenomena (verbs and pronouns) that still pose problems for current SMT systems.

ACT is more accuracy using the disambiguation module activated by the `$UsingAlignment` parameter in the configuration file (`$UsingAlignment=1`). Once this disambiguation module is activated, two alignment options are allowed:

- Without Training alignment module activated by the `$GizaModel` parameter in the ACT configuration file (`$GizaModel=1` and `$RunGiza=1`).
- With Training alignment module (saving model) activated by the `$SavingModel` parameter in the ACT configuration file (`$SavingModel=1`).

If the `$UsingAlignment` parameter is activated, one of these 2 options has to be activated. If none of them is activated, an alert message will be printed. If both of them are activated, an automatic configuration will choose for you the first option (`$GizaModel=1`).

We therefore encourage the use of ACT on its "using alignment - without training" version (first option) since it require only the installation of GIZA++ tool and ACT is a good indicator of the accuracy of connective translation, especially in its ACTa5+6 and ACTm versions (see below for the ACT scores explanation).

2 Installation and requirements

Unzip the ACT-V1.7 archive file in directory you would like to install it to. This directory will be referred to as ACTHOME throughout the ACT configuration file. As an example:

```
$ACTHOME=/Users/Hajlaoui/tools/ACT/ACT-V1.7;
```

ACT is Perl-based. Depending on the version to be used, ACT requires only Perl on the first use (without alignment), Perl and GIZA++ software in the second use (without training) and a saving alignment system in the third use (with training).

As shown in FIGURE 1, ACT can be configured and used with two main versions: with or without disambiguation module. Two subversions of the disambiguation version can be used:

- Without Training or without saving alignment model using just GIZA++ as alignment tool at the word level.

- With training and saving an alignment model using MGIZA++ (a multithreaded version of GIZA++) which is trained in a first step on large corpus (Ex. Europarl) giving an alignment model to be applied on the new data (Source, Reference) and (Source, Candidate).

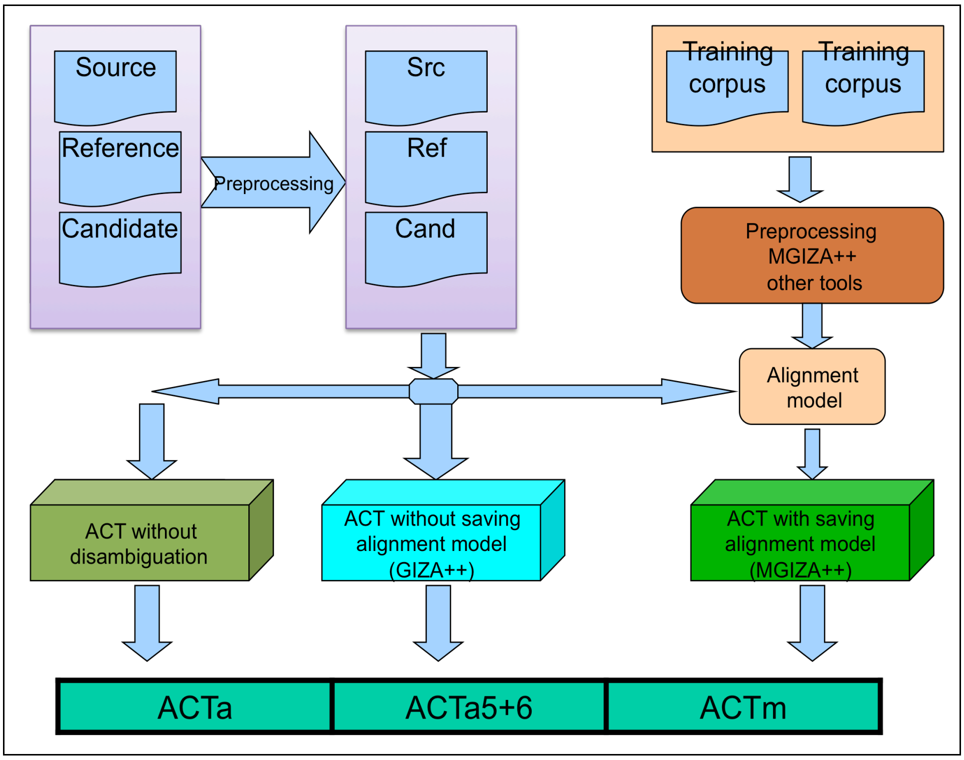

FIGURE 1 – ACT architecture.

2.1 Version 1 without alignment

This ACT version requires only Perl, go directly to the next step to configure and run ACT.

2.2 Version 2 without training

The only software requirement is GIZA++ (Och and Ney, 2000) for word-aligning your parallel data (source reference) and (source candidate).

GIZA++ is hosted at Google Code, and a mirror of the original documentation can be found here. We recommend that you download the latest version from Google Code – We are using 1.0.7:

```
Wget http://giza-pp.googlecode.com/files/giza-pp-v1.0.7.tar.gz
tar xzvf giza-pp-v1.0.7.tar.gz
cd giza-pp
make
```

The directory containing GIZA++ and plain2snt.out binary will be referred to as `PathToGIZA` throughout the ACT configuration file.

```
$PathToGIZA=/Users/Hajlaoui/tools/giza-pp-v1.0.7/giza-pp/GIZA++-v2;
```

#The GIZA's outputs will be created automatically in in a new folders `$WorkingDir/Giza-Src_Ref/.tst.A3.final` and `$WorkingDir/Giza-Src_Cand/.tst.A3.final`

#If your Giza tool is installed and/or configured differently, please adapt your configuration or change your files names to the proposed ones.

2.3 Version 3 with training

This ACT version allow to users to train an alignment model in a first step to be applied to align the data sets (source // reference) and (source // candidate) in a second step since this data set may be too small to have a good alignment result.

A saving alignment system can be based on Moses script and others tools (mainly MGIZA++ or both MGIZA++ and GIZA++) or whatever. If you have your own saving alignment system, please follow the instructions in the ACT file configuration to set up the following parameters:

```
$UsingAlignment=1; #should be activated

$GizaModel=0; #should be desactivated

$SavingModel=1; #should be activated

$SavingModel_Src_Ref= 20undoc.2000.Src-Ref.toact.v1.4.en-fr.grow-diag-final-and.giza; # as an example

$SavingModel_Src_Cand= 20undoc.2000.Src-Cand.toact.v1.4.en-fr.grow-diag-final-and.giza; # as an example
```

Two scripts are included in the ACT-V1.7 archive file:

- ACT-preprocess-xx_yy-V1.z.pl for an obligatory preprocessing step of data for the saving alignment system. (xx_yy depend on the source and target languages).
- Symmetrization2Giza-V1.pl for a postprocessing step of the saving model alignment's outputs to convert them from the format produced by simmetrized giza (or Mgiza) to giza output because ACT treats only GIZA output. This postprocessing step is not obligatory but it depends on the output of your saving model alignment system.

3 Running ACT

Once ACT is simply copied (version 1) or installed locally in your computer, a few step need to be followed to run correctly ACT.

3.1 Pre-processing of data

Data that is given to ACT should be formatted to be one-sentence-per line. Data should be tokenised and lowercased. For Arabic, ACT treates translitered (Bakwalter coding) and tokenised Arabic using MADA (Habash and Rambow, 2005). These data have to be in the working directory which will contains also the output. As an example,

`$WorkingDir=/Users/Hajlaoui/tools/ACT/ACT-V1.7/Samples/En-Fr/WithoutAlignment` (for the first version)

or

`$WorkingDir=/Users/Hajlaoui/tools/ACT/ACT-V1.4/Samples/En-Fr/WithoutTraining` (for the second version)

or

`$WorkingDir=/Users/Hajlaoui/tools/ACT/ACT-V1.4/Samples/En-Fr/WithTraining`
(for the third version)

3.2 Edit ACT configuration file

Edit the ACT configuration file: `$ACTHOME/ACT-V1.7.config`

By default, ACT configuration variables use the English-French sample data included in the ACT archive.

3.3 Run perl ACT-V1.7.pl

Go to the `$ACTHOME` directory and run perl ACT-V1.7.pl

```
cd /Users/Hajlaoui/tools/ACT/ACT-V1.7/

perl ACT-V1.7.pl

or perl ACT-V1.7.pl >/Users/Hajlaoui/tools/ACT/ACT-V1.7/Samples/En-Fr/WithoutAlignment/TerminalTrace.txt
```

4 ACT scores

For each discourse connective in the source text that must be evaluated, the metric first attempts to identify its reference translation and its candidate translation. To identify translations, ACT uses in a first step a dictionary of possible translations of each discourse connective type, collected from training data and validated by humans. If a reference or a candidate translation contains more than one possible translation of the source connective, we apply ACT by using alignment information to detect the correct connective translation. If we have irrelevant alignment information (not equal to a connective), we compare the word position (word index) between the source connective alignment in the translation sentence (candidate or reference) and the set of candidate connectives to disambiguate the connectives translation situation, and we take the nearest one to the alignment. If the alignment information is empty, we take the nearest one to the source connective.

6 cases are detected:

- Case 1: same connective in the reference and in the candidate translation.
- Case 2: synonymous connective in the reference and in the candidate translation.
- Case 3: incompatible connective in the reference and in the candidate translation.
- Case 4: the source connective is translated in the reference but not in the candidate translation.
- Case 5: the source connective is translated in the candidate but not in the reference translation.
- Case 6: the source connective is not translated in the reference nor in the candidate translation.

For case 1 (identical translations) and case 2 (equivalent translations), ACT counts one point, otherwise zero (for cases 3-6). A key point here is thus the use of a dictionary of equivalents to rate as correct synonyms of connectives classified by senses (case 2), as opposed to identity only.

One cannot automatically decide for case 5 if the candidate translation is correct, given the absence of a reference translation. We advise then to check manually these candidate translations by one or more human evaluators. Occurrences in case5 can be printed if the `$PrintCase5` is activated. After cheking these occurences, the number of correct sentences of case 5 can be set up with the `case5manual` parameter et you have to run again ACT.

Similarly, for case 6, it is not possible to determine automatically the correctness of each sentence. Therefore, we count them as wrong to adopt a strict scoring procedure (to avoid giving credit for wrong translations), or we check them manually as with the ACTm score.

ACT generates as output a general report (`Report.txt`), with scores of each case, sentences classified by cases (6 files case 1 to case 6), and a score file (`Score.txt`) containing the 3 ACT scores. Here, an example of input and output.

Input: 3 files

```
20undoc.2000.en-fr.tok.lc.src    : source file
20undoc.2000.en-fr.tok.lc.ref    : reference file
20undoc.2000.en-fr.tok.lc.cand   : candidate file
```

Output: 11 files

```
Scr    : pre-processed source file
Ref    : pre-processed reference file
Cand   : pre-processed candidate file


Case1.txt     :occurrences of case 1
Case2.txt     :occurrences of case 2
Case3.txt     :occurrences of case 3
Case4.txt     :occurrences of case 4
Case5.txt     :occurrences of case 5
Case6.txt     :occurrences of case 6

Report.txt    : general report
Score.txt     : score file
```

The total ACT score is the ratio of the total number of points to the number of source connectives, with several possibilities to calculate it. One version is to augment the score by the number of validated translations from case 5 and case 6. Three versions of the score are proposed, shown in Equations (1)–(3) below. A strict but fully automatic version is ACT**a**, which counts only Cases 1 and 2 as correct and all others as wrong. A more lenient automatic version excludes Case 5 and Case 6 from the counts and is called ACT**a5+6**. Finally, ACT**m** also considers the correct translations found by manual scoring of Case 5 and Case 6 (their number is noted respectively |Case5corr| and |Case6corr|).

$$ACTa = (|case1| + |case2|) \Big/ \sum_{i=1}^{6} |casei| \quad (1)$$

$$ACTa5 + 6 = (|case1| + |case2|) \Big/ \sum_{i=1}^{4} |casei| \quad (2)$$

$$ACTm = (|case1| + |case2| + |case5corr| + |case6corr|) \Big/ \sum_{i=1}^{6} |casei| \quad (3)$$

where |case$N$| is the total number of discourse connectives classified in case$N$.

5 Acknowledgments

We are grateful for the funding of this work to the Swiss National Science Foundation (SNSF) under the COMTIS Sinergia Project, n CRSI22_127510 (see www.idiap.ch/comtis/)

6 References

Habash N. and Rambow O. 2005. *Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop*. In Proc. of ACL 2010, pages 573–580, Ann Arbor, Michigan.

Och F., J. and Ney H. 2000. Improved Statistical Alignment Models. Proc. of the 38th Annual Meeting of the Association for Computational Linguistics, pages 440-447, Hong-Kong, China.

7 Recommended Reading: publications

- Hajlaoui N. and Popescu-Belis A. Assessing the Accuracy of Discourse Connective Translations: Validation of an Automatic Metric, in: 14th International Conference on Intelligent Text Processing and Computational Linguistics, University of the Aegean, Samos, Greece, March 24-30, 2013, 12 p.
- Hajlaoui N. and Popescu-Belis A. Translating English Discourse Connectives into Arabic: a Corpus-based Analysis and an Evaluation Metric. Proc. of the CAASL4 Workshop at AMTA 2012 (Fourth Workshop on Computational Approaches to Arabic Script-based Languages), San Diego, CA, 8 p.
- Meyer T., Popescu-Belis A., Hajlaoui N. and Gesmundo A. Machine Translation of Labeled Discourse Connectives. Proc. of AMTA 2012 (10th Conference of the Association for Machine Translation in the Americas), San Diego, CA, 10 p.