Automatic Bilingual Corpus Collection from Wikipedia

Mark Unitt
Capita Translation & Interpreting
Huddersfield Road
Delph

Simon Tite
Capita Translation & Interpreting
Huddersfield Road
Delph

Pejman Saeghe
Capita Translation & Interpreting
Huddersfield Road
Delph

{Mark.Unitt, Simon.Tite, Pejman.Saeghe}@capita.co.uk

Abstract

This is a study to combine a number of existing technologies with newly developed tools to create an automatic tool to assist with corpus collection for machine translation. This study aims to combine technologies for domain classification, domain source identification, and comparable file alignment into a unified tool. The unified tool will be used to make the corpora collection process more focused and efficient and enable a wider variety of sources to be used.

1 Introduction

The existing method for bilingual corpus collection is a slow manual process. It focuses on identifying suitable material in the source language using search engines and then locating "matching" material in the target language. If the matching material is absolutely parallel conventional tools such as Hunalign[1] tend to yield satisfactory results. However, for non-parallel resources, their output is far from useable, since these tools have been design for the purpose of aligning parallel sentences. This presents a problem and reduces the pool of material available for creating bilingual corpora.

Our goal is to create topic specific machine translation engines. In order to create these engines, there is need for large quantities of bilingual corpora on the given topic. In this paper we examine different technologies and tools that are already available and combine them with our in-house tools to create an automatic pipeline to generate these bilingual corpora.

2 Section

This section describes the steps involved in generating parallel segments from a set of keywords.

1) Describe the topic: The topic is delineated using keywords. Initially these keywords are identified manually from field experts. This process is then automated using topic modelling techniques (Wallach 2006).

2) Locate source material: Using Wiki API[2] each keyword is searched and a list of wiki pages related to that topic are collected in the source language.

3) Locate matching material: Each of the web pages from the previous step are checked to see if they contain a link to the same article in the target language. Only the pages that have a link to the target language are kept.

4) Download and pre-processing: Source and target articles are downloaded. The text content is extracted and tokenized.

5) Alignment: Pairs of matching pages are aligned using Yalign[3].

[1] http://mokk.bme.hu/en/resources/hunalign/
[2] https://www.mediawiki.org/w/api.php
[3] https://github.com/machinalis/yalign/tree/master

Proceedings of the 38th Conference Translating and the Computer, pages 159–163,
London, UK, November 17-18, 2016.

6) Cleansing: The aligned sentences are passed through a series of filters which use heuristic methods to discover bad alignments.

7) Evaluation: The final result is evaluated with help of linguists.

2.1 Topic classification

This will be a process using domain specific material to generate lists of domain specific terms. The domain specific material will simply be a set of plain text files which are known to have content related to the domain. The basis for this section of the tool will be topic classification technologies, which will be used to provided a list of terms that have been ranked by their compatibility to the domain. Inspiration for this stage of the process came from the very useful tool BootCat (Baroni and Bernardini, 2004). This tool takes a list of terms and generates an output file containing the content of web pages that match the term list, although this is limited to monolingual data.

For the purposes of the proof of concept we limited ourselves to a short initial set of manually generated seed terms saved in plain text file. An enhancement would be to automatically generate these terms using utilities such as NLTK[4] or MALLET[5]. The terms need to be specific to the topic and not be generic or open to a number of interpretations.

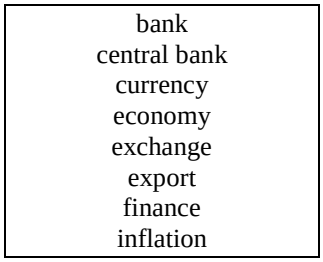

Figure 1Example term list

For a more substantive test a longer list would be extracted from a set of sample documents using MALLET. To simplify the use of MALLET a web based front end was created to control the parameters required by MALLET.

Once the term list is created, it will be converted into a number of multiple word tuples. The terms in the tuples are randomly generated with the limitation that no term is duplicated.

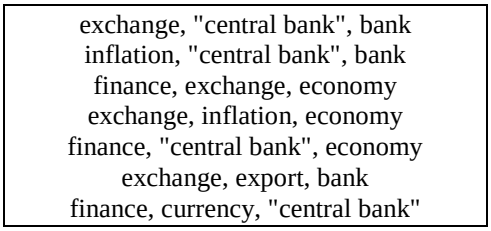

Figure 2 Example tuple list

For the purpose of providing a proof of concept we limited ourselves to seven tuples of three randomly selected terms each.

[4] http://www.nltk.org

[5] http://mallet.cs.umass.edu/topics.php

2.2 Domain source identification

Using the "tuples" generated by the previous process searches were made for web pages that match these tuples. For the initial proof of concept, the search was limited to Wikipedia and used the wiki API. For a wider search a search engine API, such as Google or Microsoft will have to be utilised. The search generated a list of Wikipedia page titles, using the search engine's API the process generated a list of URLs. A search using the seven sample tuples the search generated a candidate list of 3500 page titles. This list was then filtered to remove duplicate entries, reducing the list to 1938 titles.

2.3 Candidate URL pairing

The candidate list of page titles were processed individually to determine if a matching page existed in the target language. For the purposes of this sample exercise Spanish was chosen. Each candidate page title was processed with the Wiki API to determine if an "interlink" to a page with the Spanish language code existed. If such a page, existed it was added to the target page list. In order to test the system this search was limited to the first 50 source language titles and resulted in a target list of 22 page titles.

2.4 Candidate File downloading

Each of the files identified as having a matching target file was downloaded together with its Spanish equivalent. These files were then processed to leave just the main paragraph copy in a plain text format. Additionally any Wikipedia link and reference HTML codes were also removed. Every file was also given an additional two letter language code as a prefix to its filename.

2.5 Comparable file alignment:

The comparable alignment tool identified was Yalign. The concept behind comparable alignment is that each segment in the source file is compared against the segments in the target file. The best matching segment pairs being returned as output.

The content of the paired URLs is aligned using comparable file alignment techniques to produce segment based corpora. This process will divide the content, both source and target, into individual segments using standard text segmentation structures. The segments will then be aligned based on a simple language model and a number of string comparison techniques. The end result will be a comparability rating to each segment combination. The segment combinations that exceed the threshold will be exported for further processing, the remainder will be discarded. This is explained in more detail see (Wolk and Marasek, 2015).

In order to improve the processing of batches of files, some of the file handling routines in Yalign were separated into individual processes. These included the routine to segment the text into individual segments.

Comparing alignment technologies.

Before committing completely to a single choice of comparable alignment tool an alternative was investigated. An alignment tool was written based on the method outlined by (Mohammadi and QasemAghaee, 2010). This tool, known internally as Palign, was used as a comparison to Yalign. The two tools were tested on the same data sets and the results compared. The data was taken from a number of previously translated files. The test "source" files contained a randomised sequence of text segments from a number of test files, whilst the "target" files were those returned by human translators.

Sample	Method	# correctly aligned segments	sample size#	% of segments aligned
1	Yalign	34	84	40.47619
	Palign	23	84	27.38095
2	Yalign	32	85	37.64706
	Palign	10	85	11.76471

Figure 3. Comparative examples of different alignment tools

Although the test was limited these results demonstrated that the Yalign tool would produce the best results.

2.6 Combining the tools into one processing

Each of the components of this process were developed into a standalone module, these modules were then joined into a pipeline. In this set up the output of one module becomes the input of the next module in the pipeline. The output of the individual modules were also recorded for reporting and error checking purposes.

2.7 Corpora evaluation and clean up:

The produced corpora are evaluated both mechanically and through the use of human linguists. Poor quality segments will be discarded. The file format sent to the linguists will be in a simple excel spreadsheet. The linguists will be asked to rate the segments on a basic scale. As there may be very large numbers of segments to be rated, the segments will be divided into batches based on the source URLs. The linguists will be asked to rate the segments at a batch level.

Human evaluation

A set of 800 segment pairs were sent to a human linguist for scoring. The linguist was asked to rate each aligned pair as 1: not acceptable; 2: just acceptable and 3: acceptable. The initial result are as follows:

Rating	**# segments**	**%**
1	285	35.54%
2	50	6.23%
3	467	58.23%

Figure 4. Human evaluation of initial alignment

A 36% "noise" ratio is not good enough for the data to be used as a source for building an engine. To improve on this, a series of filters will have to be created to reduce this noise ratio.

Clean up and noise reduction

We have developed some rough filters for removing what appear to be misaligned sentences. These checks currently include

Mismatched segment length:	Translating a sentence from one language into anther will change the length of that sentence, however both the source and translated sentences will generally be of a comparable length. The filter will remove sentences where the relative length exceeds the parameter.
Mismatched numeric tokens:	The filter will count the number of groups of digits and will remove any segments where the number of digit groups differ.
Mismatched non-alphanumeric characters:	This group of characters includes punctuation, brackets and similar characters. The filter will remove those segment pairs where the ratio between the source and target non-alphanumeric characters exceeds a pre-set parameter.

Human Evaluation			**Automatic checks to filter out misaligned segments**					
			Matching Number tokens		**Matching Number tokens and strict length ratio**		**Matching Number tokens length ratio, non-alphanumeric characters**	
Rating	**# segments**	**%**	**# human rated segments left**	**%**	**# human rated segments left**	**%**	**# human rated segments left**	**%**
1	285	35.54%	132	21.85%	72	17.87%	77	16.67%
2	50	6.23%	40	6.62%	8	1.99%	13	2.81%
3	467	58.23%	432	71.52%	323	80.15%	372	80.52%

Figure 5. Effects of different filers

Applying these filters reduces the proportion of "noisy" to a more acceptable level. Further filters may be considered in the future. The goal will be to reduce this noise proportion to ideally below 10% and ultimately below 5%.

2.8 Conclusion

The work done so far has indicated that the concept is workable and will produce aligned material that can be used for MT engine creation. There are areas for improvement and further experimentation. There are for instance a number of parameters within Yalign that need to be explored that have an effect on the number and quality of the outputs provided.

We will also want to conduct a large scale test to generate a sufficiently large output set to be able to generate a machine translation engine. The output from this engine would then be compared against the output of an engine created by more "traditional" methods.

References

H.M. Wallach. 2006. ‘Topic modeling’, Proceedings of the 23rd international conference on Machine learning - ICML ’06, . doi: 10.1145/1143844.1143967

M. Baroni and S. Bernardini. 2004. BootCaT: Bootstrapping corpora and terms from the web. Proceedings of LREC 2004.

K. Wołk and K. Marasek. 2015. "Unsupervised comparable corpora preparation and exploration for bi-lingual translation equivalents".

M. Mohammadi and N. QasemAghaee. 2010. "Building Bilingual Parallel Corpora based on Wikipedia".