MQP CDR#GXS1301

Automated Building of Sentence-Level Parallel Corpus and Chinese-Hungarian Dictionary

A Major Qualifying Project Report
submitted to the faculty of the
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements
for the Degree of Bachelor of Science by

Zhongxiu Liu

Yidi Zhang

April 23, 2013

Professor Gábor N. Sárközy, Major Advisor

Abstract

Decades of work have been conducted on automated building of parallel corpus and bilingual dictionary in the field of natural language processing. However, rarely have any studies been done between high-density character-based languages and medium-density word-based languages due to the lack of resources and fundamental linguistic differences. In this paper, we describe a methodology for creating a sentence-level parallel corpus and an automatic bilingual dictionary between Chinese (a high-density character-based language) and Hungarian (a medium-density word-based language). This method will possibly be applied to create Chinese-Hungarian bilingual dictionary for the Sztaki Dictionary project [26].

Key words: sentence-level parallel corpus, automatic bilingual dictionary, high/medium-density language, character/word-based language.

Acknowledgments

This project is supported by

- Worcester Polytechnic Institute (WPI)
- Computer and Automation Research Institute, Hungarian Academy of Sciences (MTA-SZTAKI)

We would like to thank WPI MQP Advisor ***Gábor Sárközy*** and MTA-SZTAKI Project Advisor ***András Kornai*** for their valuable support and guidance. We would also like to thank MTA-SZTAKI Colleague ***Attila Zséder, Judit*** **Ács** and **Katalin Pajkossy** for their generous help throughout the project.

Table of Contents

Tables and Figures

Summary

In this project, we automatically build a parallel corpus and a bilingual dictionary between Chinese (a high-density character-based language) and Hungarian (a medium-density word-based language).

Chapter 1 describes the background, related work, and our reasons for applying certain techniques in this project. Chapter 2 explains the detailed methodology and techniques that we used in this project. Chapter 3 presents our results and evaluations. Chapter 4 concludes this project and Chapter 5 discusses about potential future work. Appendix A describes the tools we created in this project. Appendix B is a simple user guide for creating one's own parallel corpus and bilingual dictionary in four steps using our methodology.

Chapter 1: Background and Introduction

In this chapter, we introduce the theory and techniques applied in this project. While explaining and reviewing the history of those theory and techniques, we explain our reasons for our applications.

1. Parallel Corpus in Natural Language Processing

1.1.1 Machine Translation and Parallel Corpus

Today's world is facing the challenge of unbalanced language distributions. According to [1], a dozen large languages account for 40% of the population whereas over 5000 small languages account for only about 4% of the population. Despite the information explosion, the uneven distribution hurts the population speaking medium or low-density languages in information accessing. Moreover, the expansion of digital information, the ever-growing online population, and the increase of international collaboration all call for a more efficient way of translating between languages. These factors have made machine translation a very active research area in the field of natural language processing.

One major approach for machine translation is developing mathematical models with machine learning techniques. The most ideal training data for mathematical translation models is parallel corpus -- a large collection of text paired with its translation. Parallel corpora have been successfully used to train various types of mathematical translation models such as phrase-based [3], [4], syntax-based [7], [8] and class-based [6] translation models. To be more specific, a parallel corpus helps mathematical models learn to create phrase-to-phrase pairs, syntactic unit pairs, and class pairs (which are groups of words with similar meanings such as "Friday" and

"Sunday") between two languages. The resulting pairs are applied to match one language to another during the machine translation process.

Therefore, the availability and quality of parallel corpora are crucial to the quality of mathematical translation models. Corpora with sufficient word coverage and accurate alignment are in high demand. To guarantee the quantity and quality of parallel corpora, many collecting and processing techniques are used. The selection and application of those techniques largely depend on the properties of the languages in the parallel corpus.

1.1.2 Techniques for Collecting and Processing Parallel Corpus

The two main techniques for collecting corpora are automatic collection from web pages (web mining) and manual collection.

Web Mining is a technique to automatically extract information from webpages. This technique takes advantage of the abundance of free digital information on the Internet. For example, STRAND [17] is one of the earliest parallel corpora created with web mining. Firstly, STRAND locates pages that have versions in other languages or pages that contain hypertext links to the same document in different languages. Secondly, pairs of potential translated pages are generated using automatic language identification, URL-matching and document length comparison. Thirdly, the pages' underlying HTML structures are analyzed to check whether the pages are real translations of each other.

Manual Collection means collecting corpora using human labor. This method can result in a parallel corpus with better diversity and better language coverage. A good example is [9], where a total of 11550 parallel documents between Hungarian and English were collected from literature, legal documents, captions, and six other categories.

Bringing down parallels corpus' alignment level from document to sentence improves the quality of the training data for mathematical translation models. The preprocessing step of this pairing is to find the word and sentence boundaries in the texts separately.

Segmentation (Tokenization) is the process of breaking the text up into sentences, words or other meaningful units.

Sentence-level segmentation can be tricky in western languages due to the ambiguousness of stopping punctuations. For example, the English period '.' does not only appear at the end of sentences, but also appears in abbreviations (e.g. 'Mr.'), internet URLs, or ellipsis marks. In fact, 47% of the English periods in Wall Street Journal Corpus [46] and 10% of the periods in Brown corpus [38] denote abbreviations instead of sentences' boundaries [37].

On the contrary, word-level segmentation it is much easier in western languages than in Chinese. In western languages, word-level segmentation only requires separating words from tokens that are not part of the word (e.g. periods that are not abbreviations or decimals). In the Chinese language, however, word-level segmentation requires combining or separating individual characters (Section 1.3). Decades of work has been put into Chinese word-level segmentation, resulting in many complicated approaches with the help of statistical models and machine learning models (usually constructed based on word frequency) [22], [23], dictionaries, or a mix of both [24].

Recent research suggests two main strategies for sentence-level and word-level segmentation: rule-based approach [20] and adaptive approach [21].

Rule-based approach segments sentence or word with a set of manually designed rules to detect sentences or words boundaries. Below is an example set of rules used in a rule-based approach for English sentences segmentation [39].

(a) If it is a period, it ends a sentence;

(b) If the preceding token is on my hand-compiled list of abbreviations (Mr. Dr. Ms., etc.), then it does not end a sentence;

(c) If the next token is capitalized, then it ends a sentence.

Unlike rule-based approach segmenters, adaptive approach segmenters automatically generate sentence segmentation rules using machine learning techniques. To be more specific, corpora with or without pre-marked sentence boundaries [40], [41] are used to train machine learning models. These models learn rules through finding the possibility of occurrences of the ending words and beginning words of sentences

In this project, for both Hungarian sentence-level and word-level segmentation, we use rule-based tool Huntoken [15].

For Chinese sentence-level segmentation, we created our own rule-based segmenter. Chinese sentence boundaries are less ambiguous then Hungarian sentence boundaries (Section 1.1.4). Therefore, the simple rule-based approach will be accurate enough to satisfy our needs. Moreover, Hunalign [9] (the tool we used in the subsequent step to create sentence-level parallel corpus) is able to combine or split sentences to achieve the best sentence alignment, which will correct some segmentation mistakes. For example, suppose a sentence A with its parallel translation B is incorrectly segmented to sentences A1 and A2. Then if Hunalign detects that both A1 and A2 match part of B, it will combine A1 and A2 back to one sentence. Therefore, our result is unlikely to be affected even with small amount of mistakenly segmented sentences.

For Chinese word-level segmentation, we used Stanford Word Segmenter [13] developed by the Stanford Natural Language Processing Group, which is explained in detail in Section 2.2.2.

Sentence Alignment is the process of pairing sentences from one language to another. Two sentence alignment approaches are suggested in previous research: length-based alignment that aligns sentences of similar length or token length [18], [10] and word-based alignment that pairs sentences based on the words' correspondences calculated by statistics information or by dictionary (dictionary-based) [5], [19].

1.1.3 Parallel Corpus and the Density of the Language

Language density is defined as the availability of digital stored materials written in the language. This language density may not be indicated by the population of speakers. For example, Indonesia has three times the population of Germany, but Indonesian is of lower density than German, which means that there are less digital materials available in Indonesian than in German. Chinese and English are high density languages, while many European languages such as Hungarian are medium density languages.

Due to the availability of resources, early research focused on parallel corpora between high-density languages, such as between English and French [10]. Later, with the improvement of text collecting techniques, more parallel corpora involved medium-density languages, such as between English and other European Languages [2], [12], [9]. Parallel corpora between two medium-density languages were also made available using a third high-density language such as English as the intermediary.

With regard to collecting parallel corpora involving medium-density languages, manual collection is more preferable than web mining. This is because the densities of languages significantly affect the quantity of machine-detectable parallel corpora. For example, a Hungarian corpus of 3.5 million unique pages yielded only 535 pages of bilingual parallel corpus

[14]. Therefore, we used manual collection in this project where we created a parallel corpus between Hungarian (medium-density) and Chinese (large-density).

1.1.4 Parallel Corpus between Chinese and Hungarian – Languages with Fundamental Linguistic Differences

Another key challenge of this project was the difficulties caused by the differences between Chinese and Hungarian linguistics, and the different writing systems. Firstly, the basic unit of a word is different in Chinese and Hungarian. In Hungarian (as well as in English and many western languages), the basic unit of a word is a letter from a limited set (alphabet). In Chinese, the basic unit of a word is a character (where each character has its own multiple meanings) from a much larger and still expanding set of approximately 80,000. Secondly, words in Hungarian vary from their root form depending on their functionality in the sentence (for example: "Budapest" is the capital of Hungary, but "Budapesten" is used when saying "at Budapest"), whereas words in Chinese do not have time tenses, plural forms, or other forms different than the original. Thirdly, unlike Chinese sentences, Hungarian sentences have spaces between words. Fourthly, the Hungarian period is used to indicate the end of a sentence, an abbreviation, or a decimal point, while Chinese has unambiguous sentence-ending markers. In the Chinese language, sentence-stopping punctuations such as "。？！" will only appear at the end of a sentence in normal text.

These fundamental linguistic differences make it clear that different preprocessing steps are crucial for both Chinese and Hungarian before performing sentence-alignment, especially when doing word-level segmentation, sentence-level segmentation, and when creating a bilingual dictionary. Extra techniques such as stemming must be used to guarantee the dictionary's quality and usefulness later in sentence alignment process.

1.2 Automatic Bilingual Dictionary

Bilingual dictionaries are of great importance in the field of natural language processing. Besides being widely applicable in machine translation (e.g. for creating sentence-level parallel corpora), bilingual dictionaries help in many other fields including cross-language information retrieval [30] and cross-language plagiarism detection [31].

Early research used a context-based approach for creating automatic bilingual dictionaries [32], [33]. A context-based approach constructs word pairs based on their similarity scores, which are calculated from words' occurrence frequencies in parallel documents. Later research suggested new approaches such as syntax-based [34] and character-similarity [35] approaches. In the syntax-based approach [34], the similarity score (dependency heterogeneity) measures the similarity between a word and its translation. In the character-similarity [35] example, word pairs are extracted based on the similarity between Japanese and Chinese characters, and are used to train statistical models for finding more word pairs.

In this project, we used a context-based approach to create an automatic bilingual dictionary. There are two reasons why we created our own Chinese-Hungarian dictionary. First, we failed to find available Chinese-Hungarian dictionaries online that have both good quality and good coverage. Second, creating automatic bilingual dictionaries is an important problem for machine translations involving medium-density languages (or potentially low-density languages).

Chapter 2: Methodology

In this chapter, we discuss the details of the methodology and techniques we used in our project.

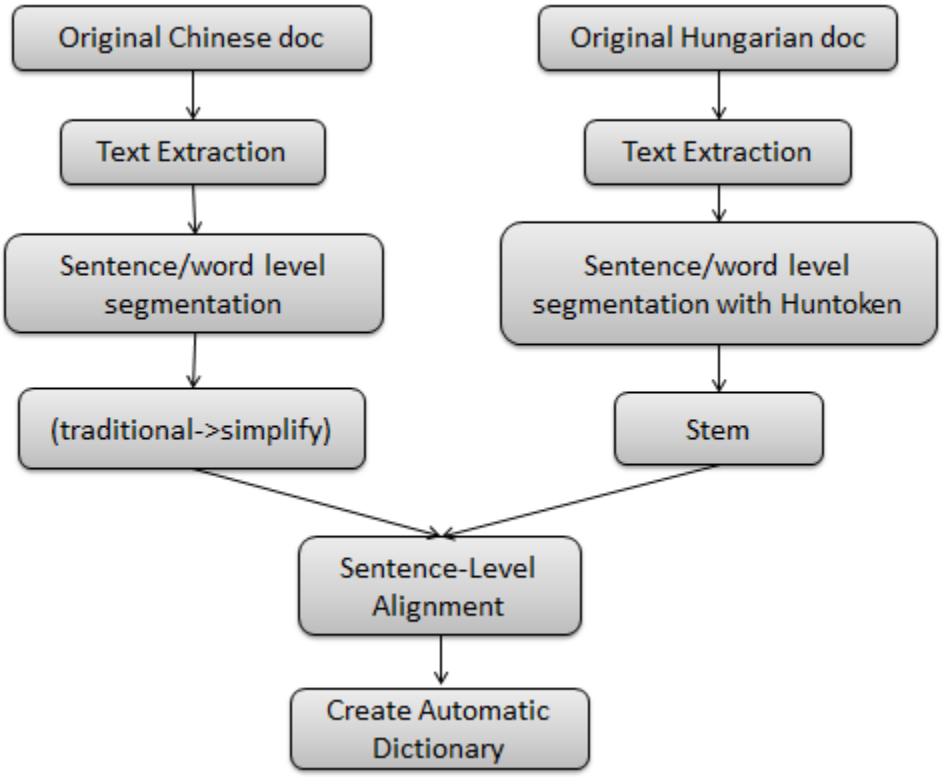

Figure 1: Methodology Flowchart of this Project

The flowchart [Figure 1] above summarizes the methodology used to create the sentence-level parallel corpus in this project. In general, parallel documents of Chinese and Hungarian are collected and preprocessed before passing into Hunalign [9]. Then, Hunalign [9] creates a sentence-level parallel corpus, which is later used by Hundict [42] (Section 2.6.1) to create an automatic dictionary.

2.1 Collecting Parallel Corpus

For medium density languages, parallel web pages exist only in small numbers (Section 1.1.3). Therefore, we collected parallel corpora manually from the web. After collection, we changed all the Hungarian and Chinese documents into *utf-8 without BOM* encoding and txt format.

Additionally, we filtered out certain parallel documents whose Chinese and Hungarian versions are significantly different in size. The difference in size is due to the incompleteness of the document in either one of the two languages. We filtered these documents out because they resulted in many incorrectly parallel sentences in our experiments. These incorrect sentence alignments will create noise in the final phase of creating bilingual dictionary out of the sentence-level parallel corpus.

After filtering and other quality checking processes, we collected a total of 60 literatures and religion books that had both Hungarian and Chinese versions.

2.2 Normalizing a Chinese Document

Normalizing a Chinese document involves sentence and word-level segmentation, and can sometimes involve traditional to simplified Chinese conversion.

2.2.1 Sentence-Level Segmentation

Because sentence-level segmentation in Chinese is much easier to handle, we created our own rule-based tool Chinese-sentencizor.py for the sentence-level segmentation of the Chinese language. The detail of this tool is described in Appendix A.

2.2.2 Word-Level Segmentation

Stanford Word Segmenter [13] is a tool for tokenizing Chinese sentences into words, developed by the Stanford Natural Language Processing Group.

The Stanford Word Segmenter uses a complicated statistical sequence modeling framework called conditional random field (CRF) [11]. The segmenter treats the segmentation task as a binary decision task, labeling each character as either the beginning or the continuation of a word. The segmenter also considers linguistic features such as n-grams, morphological and character reduplication. In the end, Stanford Word Segmenter selects label sequences with the largest probability.

As mentioned in Section 1.1.4, Chinese and Hungarian have fundamental linguistic differences. Before creating a parallel corpus, word-level tokenization must be done in Chinese. Hungarian to Chinese word matching is necessary in the dictionary-based alignment step of Hunalign. Furthermore, the number of shared words between sentences is a very important component of the sentence pairs' similarity scores, which is the criterion for aligning sentences.

The output of Stanford Word Segmenter is as below:

出去 (Outside)　散步 (Take a Walk)　　是 (Is)　不 (Not)　　可能 (Possible)

了 (Interjection no meaning)。

2.2.3 Traditional to Simplified Chinese Conversion [27]

Among all the Chinese documents used in the steps of collecting parallel corpora and creating dictionaries, very few are in traditional. Therefore, a traditional to simplified Chinese conversion tool is used to process some documents.

The only difference between traditional and simplified Chinese is the writing method of about 2000 characters. These characters have fewer strokes in simplified Chinese, but they have very similar shapes and the exact same meanings in both traditional and simplified Chinese.

Therefore, traditional to simplified Chinese conversion only concerns replacing certain characters with no changes on the word and sentence-level.

2.3 Normalizing a Hungarian Document

Normalizing a Hungarian document involves sentence-level segmentation, word-level segmentation, and stemming.

2.3.1 Word and Sentence-Level Segmentation

Huntoken [15] is a rule based boundary detector and word segmentation tool for Hungarian, English, and many other western languages, developed by the Budapest University of Technology and Economics. Huntoken segments a document on both word-level and sentence-levels, and outputs an XML file of the segmented document.

Huntoken detect paragraphs and sentence endings through recognizing blank lines and special characters (e.g. \r or \n). Huntoken separates each paragraph by xml tags and sentences by <s>. After paragraph and sentence segmentation, Huntoken wrap every word with <w> and each punctuation with <c>. For example, a paragraph with one sentence "I love science" will be output by Huntoken as:

<s><w>I</w><w>love</w><w>science</w><c>!<c/></s> "

In the output of Huntoken, there are some other formatting tags such as <?xml version="1.0" encoding="ISO-8859-2"?>. These tags do not offer useful information for our project.

After Huntoken output an XML file, a Python tool huntoken_parser.py was used to change the XML format back to the txt format. The detail of Huntoken_parser.py is explained in Appendix A.

2.3.2 Stemming

Hunmorph [25] is a tool for stemming Hungarian words developed by a group in the Media Research Center in Hungary. Stemming is the process of retrieving the root form of a word (for example, retrieving “tanul” from “tanulok”). The concept of a stem cannot be applied to certain languages such as Chinese, in which characters do not change their format based on their functionality in the sentence (Section 1.3).

Stemming is crucial in creating a parallel corpus between Hungarian and Chinese, especially when a bilingual dictionary is applied. After stemming, the same Hungarian word in different formats will be changed back to the same root form, which will then be matched with the same Chinese characters based on the input or the automatically created dictionary.

The output of Hunmorph is as below, where the first word of each line is the original word and the word before “||” is the stemmed work. A bash script was written to transform the original literature into Hunmorph’s input form and then pack stemmed words back into the original literature.

```
A a||ART K S
babonás babonás||ADJ K S
félelem félelem||NOUN K S
már már||ADV K S
megérintett, megérintett,||NOUN G U
de de||CONJ K S
még még||ADV K S
nem nem||ADV K S
lett van||VERB<PAST> K S
rajtam én||NOUN<PERS<1>><CAS<SUE>> K S
úrrá. úrrá.||NOUN G U
```

2.4 Creating Dictionaries as the Input of Hunalign

This section describes the three Chinese-Hungarian dictionaries we created as the original input of Hunalign. The results using the three dictionaries are compared and a final dictionary is decided as the input of Hunalign.

2.4.1 Dictionary1 – Combining Web-mined Dictionary with Upenn [36]

The steps involved in creating the first dictionary are explained below:

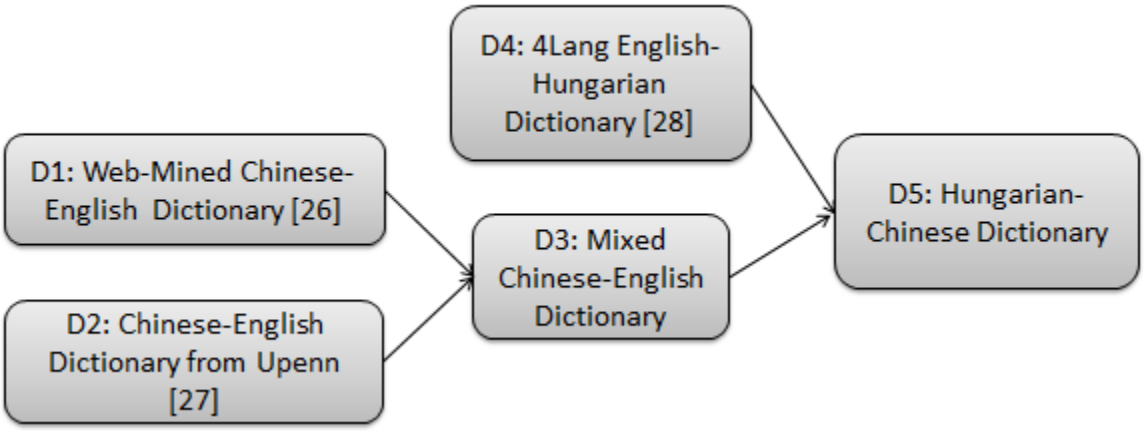

Figure 2: Flowchart of Creating Dictionary 1

First, a web-mined bilingual dictionary from Wiktionary (D1) was mixed with a UPenn dictionary (D2). To be more specific, all English words occurring in both D1 and D2 occur and only occur once in D3; if an English word occurs in both D1 and D2, its Chinese meaning from both dictionaries are mixed.

Second, the mixed Chinese-English dictionary (D3) was combined with an English-Hungarian dictionary 4Lang (D4). 4Lang is a seed dictionary of approximately 3500 entries composed of the most frequently used Hungarian words based on Longman Defining Vocabulary, Google unigram count [29], BNC, and the most frequent 2000 words from Polish and Hungarian from [9], [14]. Combining D3 and D4 result in a Hungarian-Chinese dictionary D5; this is used to pass to Hunalign.

2.4.2 Dictionary2 – OpenSubtitle.dict

OpenSubtitle.dict is created through running Hundict on Open-Subtitles [16]. OpenSubtitles is a sentence-level parallel corpus web-mined from Chinese and Hungarian movie subtitles. OpenSubtitles is originally in traditional Chinese, so a traditional to simplified conversion (Section 2.2.3) was applied to the original documents. OpenSubtitles is one of the open parallel corpora downloaded from OPUS – a growing collection of parallel corpus obtained from the web.

We experimented with multiple DICE values (Section 2.6.1) when applying Hundict [42] on OpenSubtitles. The largest dictionary resulted with DICE 0.01 has 10330 word pairs.

2.4.3 Dictionary3 – Upenn_Upann.dict

Upenn_Upann.dict was created by joining a Hungarian-English dictionary (82763 words pairs) from Attila Vonyó [47] and an English-Chinese dictionary (78585 words pairs) from Upenn LDC [36]. This resulted in a Hungarian-Chinese dictionary with 276965 word pairs. Both the Hungarian-English dictionary and English-Chinese dictionary contain many-to-many translations while the resulting Hungarian-Chinese dictionary contains only one-to-one translation (as required by Hunalign). This explains the size difference between the dictionaries.

Suppose we have the two below dictionaries with word pairs with the same English words E1 and E2, each English word is translated into Chinese (Z) or Hungarian word (H) in different dictionaries:

English-Chinese Dictionary: E1 -> Z1, Z2; E2 -> Z1

English-Hungarian Dictionary: E1 -> H1, H3; E2 -> H1, H2

Then the combined dictionary will be:

Z1->H1, Z1->H2, Z2->H1, Z2->H3 (E1=E1)

Z1->H1 (removed due to duplication), Z1->H2 (E2=E2)

2.4.4 Evaluating and Choosing Dictionaries

After obtaining the three dictionaries described above, we evaluated these dictionaries from different perspectives and decided to use Upenn_Upann.dict as the input for Hunalign.

Manual Checking

Our first approach to evaluate the dictionaries is manual checking. To our knowledge, this project will result in the first automatically created dictionary between Hungarian and Chinese languages. Therefore, we lack available quantified methodologies or tools that we can use to automatically evaluate the Hungarian-Chinese dictionary (and parallel corpus) we created.

For the only Hundict-created dictionary Opensubtitles.dict, we first sort word pairs by their similarity scores calculated by Hundict. Second, we divided similarity scores into ranges and randomly choose 10 word pairs with similarity scores from each range. Third, Hungarian and Chinese native speakers of proficient English level sit side by side and explain the meaning of Hungarian and Chinese word from the same word pair to each other in English. Fourth, Hungarian and Chinese speakers decide on whether the words in a word pair are the correct translations of each other. This manual process gives us information about the accuracy of the dictionary, both partly (based on similarity scores) and overall.

For dictionary 1 and Upenn_Upann.dict in which word pairs do not have similarity scores, we skipped the first and second step. Instead, we randomly selected word pairs to be examined and follow the same manual checking method as in the third and fourth steps.

Size Comparing

Another factor we used to evaluate the dictionaries is the size of each dictionary. A larger size dictionary is preferable because it offers more information to help Hunalign with aligning

sentences. The numbers of word pairs in each dictionary are shown in the table below. The table shows that Upenn_Upann.dict has an absolute advantage in quantity comparing with the other two dictionaries.

Dictionary	Number of Word Pairs
Base.dict	29,363
Opensubtitle.dict	10,330
Upenn_Upann.dict	276,965

Table 1: Numbers of Word Pairs in the Dictionaries We Created

Evaluating Opensubtitles.dict Using Upenn_Upann.dict

The approach described in this part was used for evaluating Opensubtitles.dict. Evaluating Opensubtitles.dict not only helps us make decision on which dictionary to use as the input of Hunalign, but also helps us evaluate the Hundict [42] tool, which is the main tool we will use to generate automatic dictionary out of sentence-level parallel corpus.

We evaluated Opensubtitle through extracting the common word pairs from Opensubtitle.dict and Upenn_Upann.dict. We chose Upenn_Upann.dict because: firstly, it is a big enough dictionary that will share enough common word pairs with Opensubtitle.dict; secondly, from the previous evaluations, we found that Upenn_Upann.dict has the best precision.

Evaluating Dictionaries Using Hunalign [9]

Moreover, we evaluated all three dictionaries through running Hunalign [9] (Hunalign is explained in detail in Section 2.5). First, we randomly selected 5 out of the 60 parallel documents as an experiment sample. Then, we ran Hunalign using each of the three dictionaries as input, resulting in three versions of sentence-level parallel corpus for each parallel documents. Lastly,

we checked the accuracy of all resulting parallel corpora using the same manual method we used before: randomly selected sentence pairs out of each corpus, and let Chinese and Hungarian native speakers communicate in English to decide the correctness of sentence pairs.

After checking the outputs of Hunalign, we all agreed that using Upenn_Upann.dict as input achieved the most accurate sentence-level alignment.

2.5 Sentence-Level Alignment using Hunalign [9]

After processing the original documents and creating dictionaries, sentence-level alignment was run.

2.5.1 about Hunalign [9]

Hunalign is a tool that takes in tokenized and sentence-segmented parallel text in two languages and outputs a sequence of bilingual sentence pairs, written by Hungarian scientist David Varga. Hunalign uses a hybrid algorithm of length-based and dictionary-based alignment: it can either use an input dictionary or an automatically-built dictionary built by Hunalign itself based on word frequency.

Hunalign starts with calculating sentence pairs' similarity scores using the below formula:

- Token-based score $= \frac{num\ shared\ words\ between\ the\ two\ sentences}{larger\ token\ count\ of\ the\ two\ sentences}$ + *award (for high proportion of shared tokens)*
- Length-based score $= \frac{length(longer\ sentence)+1}{Length(shorter\ sentence)+1}$

After computing similarity scores, Hunalign creates an alignment matrix with sentences from one language as rows and the other language as columns. With the alignment matrix, Hunalign goes through the matrix to find the best alignment rail using the score of skipping sentences and coalescing sentences learnt from the training corpus. Lastly, Hunalign iteratively

coalesce neighboring pairs if better scores are achieved, and thus discovers one-to-many and many-to-one sentence alignments as well.

Hunalign has the big advantage of discovering one-to-many and many-to-one sentence alignment. The reason is that a sentence with conjunctions is likely to be translated into more than one sentence in another language for the purpose of a clearer meaning. Moreover, a failure to discover many-to-one or one-to-many sentence alignment may lead to a failure of all sentence alignments afterwards. Thus, Hunalign is preferred over other available sentence alignment tools in this project.

2.5.2 Using Hunalign [9]

Hunalign is used with the following command:

hunalign-1.2/src/hunalign/hunalign [Dictionary] [Hungarian Document] [Chinese Document] > [Output]

In the command above, *[Dictionary]* is the Upenn_Upann.dict we discussed in Section 2.4; *[Hungarian Document]* is the normalized and stemmed Hungarian document; *[Chinese Document]* is the normalized Chinese document. The *[Output]* contains line-number pairs of the sentence pairs from the *[Hungarian Document]* and *[Chinese Document]*. *[Output]* is then passed to the Hunalign_textAligner tool (Appendix A), which transforms the line-number pairs back to the original sentence pairs, resulting in the sentence-level parallel corpus of which an example is shown in Figure 4 of Section 3.1.1.

2.6 Extracting an Dictionary out of Sentence-Level Parallel Corpus

2.6.1 about Hundict [42]

Hundict is a tool for extracting bilingual dictionaries from sentence-aligned parallel corpora, written by Hungarian scientist Attila Zseder from SZTAKI.

Hundict uses a context-based approach with the help of contingency table [43] and Dice's coefficient [44], [45]. Contingency table is a statistical matrix representing the distribution of the frequency of variables. Dice's coefficient is a statistical measurement for the similarity of two samples. The formula of Dice's coefficient that we used in this project is:

$$QS = \frac{2C}{A+B} = \frac{2|A \cap B|}{|A| + |B|}$$

Below is an example of how Hundict utilizes the contingency table and Dice's coefficient to generate dictionaries. Suppose we have a Chinese-English corpus of 1000 parallel sentences; one of the sentences pairs is:

I love science.

我 爱 科学 。

Then the resulting contingency table for this sentence pair will be as below:

	Co-Occurrence	En-Occurrence	Zh-Occurrence	No-Occurrence
I-我	300	100	200	400
love-爱	200	80	20	700
science-科学	50	40	10	900
I-爱	50	20	30	900
Love-科学	5	80	70	845
Science-我	5	70	90	835
I-科学	8	90	70	832
love-我	4	100	90	806
Science-爱	5	80	80	835

Table 2: Example of Contingency Table Created by Hundict

Taking the word pair “I-我” as an example: In the contingency table, Co-Occurrence = 300 means that there are 300 sentence pairs in which “I” occurs in the English sentence and “我” occurs in the corresponding Chinese sentence; the En-Occurrence = 100 means that there are 100 sentence pairs in which “I” occurs in the English sentence and “我” does not occur in the corresponding Chinese sentence; the Zh-Occurrence = 200 means that there are 200 sentence pairs in which “I” does not occur in the English sentence but “I” occurs in the corresponding Chinese sentence; No-Occurrence = 400 means there are 400 sentences pairs in which neither “I” nor “我” occurs in English or corresponding Chinese sentences. Notice that all the numbers in Table2 are made up for the purpose of explaining how Hundict works.

After completing the contingency table, Hunalign will use the Dice’s Coefficient function to calculate the two words similarity.

$$QS = \frac{2|A \cap B|}{|A| + |B|}$$

$$= \frac{2 * Co - Occurence}{(Zh - Occurence + Co - Occurence) + (En - Occurent + Co - Occurence)}$$

$$= \frac{2 * 300}{(200 + 300) + (100 + 300)}$$

= 66.7%

Therefore, the similarity score for “I-我” is 66.7%

2.6.2 Applying Hundict [42]

Hundict is used with the following command:

python hundict.py [Options] [Input_File] [Bound] [Scorer]

[Scorer] is the mathematical formula used to calculate the similarity scores of word pairs. In this project, we use DICE, which is explained in Section 2.6.1, as the *[Scorer]*. *[Bound]* is the minimum similarity score for accepting a word pair into the output dictionary. [Options] are the options offered by Hundict which can be seen by using the command "python hundict.py –h". In this project, we mainly experimented with the three options described below:

--iter is an option specifying the numbers of iterations that Hundict will go through to create the final dictionary. In every iteration, Hundict goes through all sentences to search qualified word pairs, with previously qualified word pairs being removed.

--ngram is an option helping detect the situation where one word is translated to n words in another language. For example, a single Chinese word "首相" is translated into two English words "Prime Minister". If -- ngram is set to be false, using Dice's Coefficient will only detect 1gram words translation. This means that instead of detecting word pair "首相-Prime Minister", hunDict will detect "首相-Prime" or "首相-Minister". If -- ngram is set to be true, then for word pair W1-W2, Hundict will check the words occurring before and after W2 in the original documents, and combine W2 with words that frequently occurring around it. Thus, one-to-many and many-to-one words translations can be detected.

-- **sets** is an option helps detect the situation where one word has multiple meanings and thus being translated into different words in another language. This situation frequently happens when translating western languages. For example, the English word "way" can be translated into both Chinese words "路 (road)" and "方法 (method)".

Recall that in Section 2.6.1, a word pairs' similarity score is calculated as

$$\frac{2 * Co - Occurence}{(Zh - Occurence + Co - Occurence) + (En - Occurence + Co - Occurence)}$$

If both word pairs “way-路” and “way-方法” occur with similar frequency, then the En-Occurrence and Zh-Occurrence will be much larger comparing with Co-Occurrence, which will result in very low similarity scores for both word pairs. Therefore, without using –sets to handle this situation, neither “way-路” nor “way-方法” is likely to be detected.

We ran Hundict using options of different values and their combinations on sample sentence-level parallel corpora. Empirically, setting -- *ngram* and/or -- *set* as true will require a much longer amount of time to create the dictionary. Moreover, almost all the extra word pairs created by -- *ngram* and -- *set* have very low similarity scores and low accuracy.

After several manual checking rounds, we found that running Hundict with *Dice = 0.2, --Iter = 5, --ngram = false and -- set = false* resulted in the most accurate Hungarian-Chinese dictionary. We ran Hundict with these options on each literature and combined their resulted dictionaries, with duplicated word-pairs and name pairs removed.

2.6.3 Postprocessing -- Filtering out Name Pairs

Hundict [42] generates word pairs based on their common occurrences. Therefore, the final dictionary generated contains some name pairs. A bash script is written to filter our all these meaningless name pairs out of the final dictionary. Basically, for the Hungarian word of each word pair in the final dictionary, if the frequency of the capitalized Hungarian word is 10 times bigger than the frequency of the not capitalized Hungarian word in the original document, then the word pair is listed as a potential name pair. For example, if “Henry” is used more frequently in the original documents then “henry”, then “Henry – 哈利” is probably a name pair.

Below is an excerpt of the filtered out name pairs from the dictionary created out of the literature “Jane Eyre”. The number in each row is the similarity score of the word pairs.

0.0171673819742	poole	那
0.0564652738566	adle	她
0.0833333333333	town	.
0.145454545455	jóisten	上帝
0.274914089347	eyre	爱
0.413793103448	blanche	布兰奇
0.454545454545	carter	卡特
0.470588235294	thornfield	桑菲尔德
0.5	rivers	里弗斯
0.5140562249	reed	里德
0.542372881356	eliza	伊丽莎
0.555555555556	scatcherd	斯卡查德
0.583333333333	sophie	索菲娅
0.584745762712	adle	阿黛勒
0.585365853659	edward	爱德华
0.611764705882	hannah	汉娜
0.615384615385	helen	海伦
0.615384615385	madame	皮埃罗
0.625	madeira	马德拉
0.627054361568	rochester	罗切斯特
0.627450980392	brocklehurst	布罗克赫斯特
0.634146341463	eshton	埃希顿
0.642857142857	briggs	布里格斯
0.666666666667	richard	理查德
0.68085106383	leah	莉娅
0.692307692308	oliver	奥利弗
0.714285714286	george	爵士
0.727272727273	lord	勋爵
0.756097560976	georgiana	乔治亚娜

Figure 3: Example of Name Pairs

Chapter 3: Results and Evaluation

3.1 Result1 -- Parallel Corpus

3.1.1 Description of Parallel Corpus

The final corpus we created contains 26427 parallel Chinese-Hungarian sentences.

The below graph an excerpt of the sentence-level parallel corpus we created out of the novel Jane Eyre. The expert shown is selected randomly from the whole corpus.

```
" 很 多 。 - Nagyon sokat . 0.13125
" 读 什么 ？ - Mivel foglalkozott ? 0.259091
" 印度斯坦语 。 - A hindusztáni nyelvet tanulta . 0.290323
" 那 时候 你 干什么 呢 ？ - És maga azalatt mit csinált ? 0.285714
" 起 初学 德语 。 - Eleinte németül tanultam . 0.8
" 他 教 你 吗 ？ - Ő tanította magát ? 0.292857
" 他 不 懂 德语 。 - Nem . ~~~ Németül nem tud . 1.03214
" 他 什么 也 没有 教 你 吗 ？ - Semmire se tanította magát ? 0.117391
" 教 了 一点儿 印度斯坦语 。 - Egy kicsit hindusztániul . 0.702101
" 里弗斯 教 你 印度斯坦语 ？ - Rivers hindusztániul tanította magát ? -0.342857
" 是 的 ， 先生 。 - Igen . 0.586466
" 也 教 他 妹妹们 吗 ？ - És a húgait is ? 0.245455
" 没有 。 - A húgait nem . 0.28
" 光 教 你 ？ - Csak magát ? 0.72
" 光 教 我 。 - Csak engem . 0.623333
" 是 你 要求 他 教 的 吗 ？ - Maga kérte őt , hogy tanítsa ? 0.3
" 没有 。 - Nem én kértem . 0.714935
" 他 希望 教 你 ？ - Ő akarta magát tanítani ? 0.0428571
" 是 的 。 - Igen . 0.85
他 又 停顿 了 一下 。 Újabb szünet . 0.607258
" 他 为什么 希望 教 你 ？ - Miért akarta magát hindusztániul tanítani ? 0.237209
印度斯坦语 对 你 会 有 什么 用处 ？ Mi hasznát vehette maga valaha a hindusztáni nyelvnek ? 0
" 他 要 我 同 他 一起 去 印度 。 - Azt akarta , hogy menjek vele Indiába . 0.1
" 呵 ！ - Aha . 0.3
这下 我 触到 要害 了 。 Végre kibújt a szög a zsákból . 0.276
他 要 你 嫁给 他 吗 ？ Szóval feleségül akarta venni magát . 0.1
" 他 求 我 嫁给 他 。 - Feleségül kért . 0.1375
" 那 是 虚构 的 — 胡编乱造 来 气气 我 。 - Ez nem igaz . ~~~ Azért találta ki , hogy ugrasson .
" 请 你 原谅 ， 这 是 千 真 万确 的 事实 。 320 ~~~ - Bocsánatot kérek , ez a színigazság . 0
他 不止 一 次 地 求 过 我 ， 而且 在 这 点 上 像 你 一样 寸步不让 。 Nem is egyszer , hanem többsz
" 爱 小姐 ， 我 再 说 一 遍 ， 你 可以 离开 我 了 。 - Már mondtam , Eyre kisasszony , hogy elmehe
```

Figure 4: Sample Results of the Chinese-Hungarian Sentence-level Parallel Corpus

Each row contains a Hungarian sentence and its corresponding Chinese sentence; the numerical score at the end of each line is meaningless as explained by the author of Hunalign himself.

3.1.2 Evaluation of Parallel Corpus

We randomly selected 10 pieces of literature out of the 60 pieces of literature and then manually checked the sentence-level parallel corpora created out of the 10 samples. We use the same manually checking method described in Section 2.4.4. All the Hungarian and Chinese native speakers who were involved in this checking process agree on the good quality of the sample parallel corpora. Therefore, we concluded that the whole corpus created out of the 60 literatures is in good quality.

The resulting Sentence-Level Parallel Corpora created out of each literature and all literatures are made available on nessi6.ilab.sztaki.hu, under /home/szuperAurora/Results.

3.2 Result II –Chinese-Hungarian Dictionary

3.2.1 Description of Chinese-Hungarian Dictionary

The final Chinese-Hungarian Dictionary is created by running Hundict on each of the 60 parallel documents collected and combined their results, with duplicated word pairs removed. The final Chinese-Hungarian dictionary has a total of 23932 word pairs after removing 613 name pairs.

The graph below is an excerpt of the final Chinese-Hungarian Dictionary we created. The expert is selected randomly.

Figure 5: Example of Word Pairs from Final Dictionary Created

The first column contains the similarities scores of the word-pairs. The second column contains the Hungarian word and the third column contains the Chinese word.

This resulting dictionary is made available on nessi6.ilab.sztaki.hu, under /home/szuperAurora/Results.

3.2.2 Evaluation of Chinese-Hungarian Dictionary

We evaluated our dictionary using the same manual checking method as described in the previous sections. We randomly selected 20 word pairs from different similarity score ranges and manually check their accuracy.

The chart below is the summary of our evaluation result. The chart contains six similarity score ranges, the percentage of the number of the word-pairs in each range out of the whole dictionary, and the estimated accuracy of word pairs in each range. An accuracy of x/20 means that we found x correctly translated pairs out of the 20 randomly selected word pairs.

Similarity Score Range	% Word Pairs	Accuracy
[0, 0.1)	33.2%	7/20 ≈ 35%
[0.1,0.2)	19%	12/20 ≈ 60%

(0.2-0.4]	20.5%	17/20≈ 85%
(0.4-0.6]	16.1%	18/20≈ 90%
(0.6-0.8]	8.6%	20/20≈ 100%
(0.8-1.0]	2.6%	19/20≈ 95%

Table 3: Evaluation Summary of the Final Dictionary Created

Additionally, this result shows that the similarity score computed by Hundict is a meaningful and accurate evaluation of word-pairs' quality. Word-pairs with higher similarity scores have larger possibilities of being accurate.

3.2.3 Common Errors

Below are the common errors we found when checking the dictionary manually.

Error I: Translating a Word Incompletely. For example, Hungrian word 'Sellő' is translated into Chinese word '鱼(fish)' with a similarity score of 0.58, instead of '人鱼（mermaid)'. However, the Chinese word '人鱼' is actually a combination of two words '人(human)' and '鱼(fish)'. This type of error is due to the defect of Hundict's '–ngram' option described in Section 2.6.1.

Error II: Translating a Word into Words that Frequently Occur Together with it. For example, Hungarian word 'Gyalázatos' is translated into Chinese word '行为(act)' with a similarity score of 0.087, instead of '可耻(shameful)'. However, '可耻行为 (shameful act)' is a commonly used phrase in the Chinese language. This error occurs because Hundict extracts word pairs based on their co-occurrence in sentence pairs, and word '可耻' and '行为' frequently occur together.

Error III: Including non-word in word-pairs. There are also very few roman numerals and tokens mixed in the resulting dictionary and are translated into Hungarian or Chinese words. For example: Hungarian word ‘a’ is translated to Chinese period ‘。’ with a similarity score of 0.58; Hungarian ‘p’ is translated to ‘P’ with a similarity score of 0.88; ‘900’ is translated to “九百 (900 in Chinese)” with a similarity score of 0.89. Therefore, a filtering tool should be developed to clean up the Hunalign’s output.

Chapter 4: Conclusion

In this project, we created a sentence-level parallel corpus and a bilingual dictionary between Hungarian and Chinese. We presented a methodology for creating corpora and dictionaries between high-density character-based languages and medium-density word-based languages. We pipelined existing tools with auxiliary self-created tools and made all the tools available to the public.

Our project is potentially an important contribution to the well-reputed Sztaki Szótár (‘dictionary’ in Hungarian) project [26]. Currently, Sztaki Szótár collected bilingual dictionaries in 7 languages: English, Hungarian, German, French, Italian, Polish, Dutch and Bulgarian. However, all the 7 languages are European word-based languages. The introduction of Chinese, a language spoken by about one fifth of the worlds’ population, would significantly increase the diversity and influence of SZTAKI dictionaries.

Chapter 5: Future Work

The next step for future research is to test the methodology we presented in this paper on other languages. A good example is Hungarian-Japanese. Japanese is a high-density character-based language that shares great similarities with Chinese in its characters. Therefore, creating sentence-level parallel corpora and bilingual dictionary between Hungarian and Japanese will ideally be accomplished with the same methodology presented in this paper, with a few minor modifications in the original documents' pre-processing steps. Similarly, sentence-level parallel corpora and dictionaries between Chinese and other medium density European languages such as Romanian may also be easily accomplished using the methodology we presented.

Another potential focus of future research is to collect larger and better parallel documents using more efficient approaches. Although Chinese is a high-density language, it is used in a limited numbers of geographical areas instead of being used worldwide, like English. This explains the fact that more documents are translated from western languages to English than to Chinese, making it harder to collect parallel corpora between Chinese and Hungarian. If more original text documents are collected, more sentence-level parallel corpora can be created, which will expand the quantity and coverage of automatically created dictionaries.

Moreover, future research should work on improving the evaluation method of the sentence-level parallel corpora and bilingual dictionaries. The manual comparison method used in this project guarantees accuracy. However, this method is inefficient and tedious, as the size of corpora and dictionaries would become larger in future research. Therefore, a better evaluation method is necessary. One suggestion is to test the accuracy of word-pairs using another high-density language (e.g.: English) as an intermediate, testing whether both words in word pairs are translated into the same or similar English words. Another suggestion is to utilize multiple online

translation tools by seeing whether a word is often translated to its correspondent word in the word pair. However, both of the suggested methods involve extra tools or dictionary resources, and the quality of these tools and dictionaries should be convincingly evaluated.

Appendix A: Tuning Tools

This chapter explains the auxiliary tools necessary to ensure the proper use of the main tools mentioned above.

All tools described below are made available on nessi6.ilab.sztaki.hu, under /home/szuperAurora/Hundict/python-tools or /home/szuperAurora/Hunalign/python-tools. The usage and detailed description of each tool can be found in its README file.

I. Format Processing Tools

i. encode_transform.py

encode_transform.py is a tool for changing the encoding of a file from certain encoding to a target encoding. At present, the tool can recognize the most popular encodings: UTF-8, GB2312, GBK, UTF16, BIG5, Latin1 and Latin2. Users can easily extend the source encodings or change the target encoding in the source code of encode_transform.py (explained in its README).

We made this tool because tools used in this project require different encodings for input files. For example, Huntoken and Hunmorph require Latin2 encoding, while Hunalign requires utf-8 encoding. Therefore, an automatic detection transformation for input files of various encoding is necessary and crucial to the proper usage of the tools we described in the above chapter.

ii. chinese-sentencizor.py

chinese-sentencizor.py is a tool for Chinese sentence-level segmentation. chinese-sentencizor.py uses regex to define rules for recognizing sentence-ending punctuation. Chinese-sentencizor.py processes a Chinese text in the following aspects:

1) Detecting sentence boundaries and outputting each sentence into a separate line;

2) Removing white spaces and blank lines;

3) Surrounding every punctuation mark with one space before and after the punctuation.

In addition, the input file of this tool must be in utf8 encoding. This is because only known encoding can be decoded into Unicode representing the Chinese punctuations.

iii. huntoken_parser.py

huntoken_parser.py is a tool for processing the output of Huntoken. Huntoken takes in a text file and outputs an XML file. However, the outputted XML format is not accepted by both Hunmorph and Hunalign. Therefore, we developed Huntoken_parser.py as a tool to change the XML format back to the normal text. The final output after Huntoken_parser.py is a plan text file containing only the contents in the original documents, with both words and sentences segmented.

iv. hunalign_textAligner.py

Hunalign_textAligner.py is a tool for processing the output of Hunalign. Hunalign_textAligner.py takes in line-number pairs and outputs the corresponding sentence pairs.

Hunalign_textAligner.py is crucial to the success of sentence-level alignment. Hunalign's input dictionary contains only the translation of stemmed Hungarian words, therefore, Hunalign's input file must be stemmed Hungarian documents. If we directly use Hunalign to output the sentence-level parallel corpus, the Hungarian sentences will be in the stemmed form

instead of the original form. This is why we need Hunalign_textAligner.py to pair up the original unstemmed Hungarian sentences and Chinese sentences using the output line-number pairs.

II. Dictionary Processing Tools

i. join_dict.py

join_dict.py is a tool for creating a combined bilingual dictionary out of two bilingual dictionaries: one bilingual dictionary in languages A and B, and the other in languages B and C. In this project, we used this tool to combine an English-Hungarian dictionary with an English-Chinese dictionary, which creates a Hungarian-Chinese dictionary.

We used this tool to combine the two dictionaries Upenn Chinese-English dictionary and Sztaki English-Hungarian dictionary as described in Section 2.4.3. Notice that this tool is not used in creating our final automatically created bilingual dictionary which only involves Chinese and Hungarian documents. Join_dict.py is simply used as the input dictionary when we first run Hunalign to create sentence-level parallel corpus.

III. Result Evaluating and Analyzing Tools

i. filter_dict.py

filter_dict.py is a tool for creating a new dictionary by extracting all the common word pairs from two bilingual dictionaries of the same languages. Filter_dict.py is used to evaluate Hundict and the Hundict-created Opensubtitles.dict through finding common word paris between Opensubtitles.dict and Upenn_Upann.dict.

ii. wordStats.py

wordStats.py is a tool for outputting the statistics of each Hungarian word, Chinese word, and Hungarian-Chinese word pair occurrence in multiple dictionaries. In the process of automatically creating dictionaries in this project, different Hundict option and option values were explored. Therefore, wordStats is needed to compare and evaluate these dictionaries.

Below is an example of the output of wordStats.py. The first column is the number of dictionaries (out of 12 in this example) that the Hungarian-Chinese word pair occurred in. The second column is the average score of the word pair from the dictionaries it occurred in. The third column is the word pair detected by Hundict. The rest of the columns are the detailed scores of the word pairs in each dictionary.

Notice that this example below is not part of the final resulting dictionary. This example is just an output file during an experimental process.

```
1   0.00    oxford-不过 6:0.00117288294628
3   0.03    környéki-附近    6:0.00982800982801  8:0.00982800982801  11:0.00982800982801
9   1.62    elemzés-分析     1:0.18045112782 2:0.18045112782 5:0.18045112782 6:0.18045112782 7:0.1804511278
6   0.52    hadtestparancsnok-纵队   2:0.0860215053763   5:0.0860215053763   6:0.0860215053763   7:0.086021
3   0.81    karenina-卡列宁  6:0.269230769231    8:0.269230769231    11:0.269230769231
9   5.67    planchet-普朗歇  1:0.62987012987 2:0.62987012987 5:0.62987012987 6:0.62987012987 7:0.6298701298
1   0.00    mondétour-男     6:0.00433839479393
3   0.07    gólszerzés-驱逐  6:0.0229885057471   8:0.0229885057471   11:0.0229885057471
6   2.82    kétdimenzió-二維     2:0.470588235294    5:0.470588235294    6:0.470588235294    7:0.4705882352
3   0.68    városháza-市政府 6:0.22641509434 8:0.22641509434 11:0.22641509434
3   1.59    szláv-斯拉夫 6:0.529411764706    8:0.529411764706    11:0.529411764706
6   4.05    jupiter-木星     2:0.674418604651    5:0.674418604651    6:0.674418604651    7:0.674418604651
9   7.71    ílosz-伊洛斯 1:0.857142857143    2:0.857142857143    5:0.857142857143    6:0.857142857143    7:
3   0.22    elsőéves-年级    6:0.0729927007299   8:0.0729927007299   11:0.0729927007299
1   0.01    sophie-hoz-仪式  6:0.00714285714286
3   0.41    megáld-祝福  6:0.137931034483    8:0.137931034483    11:0.137931034483
2   0.03    akkora-大    6:0.0160817717206   8:0.0160817717206
9   5.89    suellen-苏伦     1:0.654353562005    2:0.654353562005    5:0.654353562005    6:0.654353562005
3   0.19    biztonsági-安全  6:0.0645933014354   8:0.0645933014354   11:0.0645933014354
5   0.14    jóslástanórá-课  2:0.028 6:0.028 7:0.028 8:0.028 11:0.028
6   1.49    esküszik-发誓    2:0.248772504092    5:0.248772504092    6:0.248772504092    7:0.248772504092
```

Figure 6: Sample of the wordStats.py Output

Appendix B: User Guide -- Create Your Own Sentence-Level Parallel Corpus and Bilingual Dictionary in Four Steps

This appendix describes the fastest way to create your own Chinese-Hungarian sentence-level parallel corpus and dictionary based on our project. The four steps required are shown in the below graph:

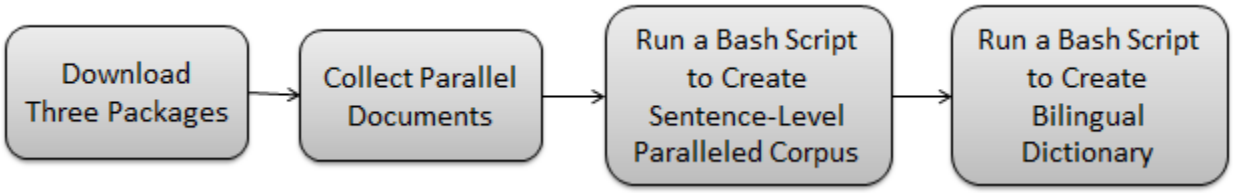

Figure 7: User Guide Flowchart

Step 1: Download the packages listed below from nessi6.ilab.sztaki.hu, under /home/szuperAurora

Package Name	Package Function
Hunalign	Create Sentence-level Parallel Corpus
Hundict	Create Bilingual Dictionary Automatically
CorpusTXT	Parallel Hungarian-Chinese Corpus we collected

Table 4: Packages to Download in User guide

Each package has a README folder contains the documents for all the tools and bash scripts inside the package. However, to create your sentence-level parallel corpus or bilingual dictionary, you DO NOT need to learn about using each tool inside each package. You ONLY

need to run the bash script in Step 3 and 4 to create sentence-level parallel corpus and a bilingual dictionary, respectively.

Step2: Collect your Parallel documents

All Chinese and Hungarian documents collected should be in *utf-8 without BOM* encoding and *txt* form. The proper format of the documents is necessary for the success of the next steps.

You can write bash script to transform your documents' encoding. Use *file –bi [Input File]* to find out the encoding of your documents. Use *iconv –c –f [Current Encoding] –t [Target Encoding] [Input File] > [Output File]* to change file encodings. You can also use our python tool encode-transform.py following the instructions in its readme file. From our experience, the Macs system is the most convenient operating system to find and edit encoding.

To avoid editing the bash script and tools we created, we suggest you to organize documents in the same structure as in CorpusTXT. The structure is shown in the figure below.

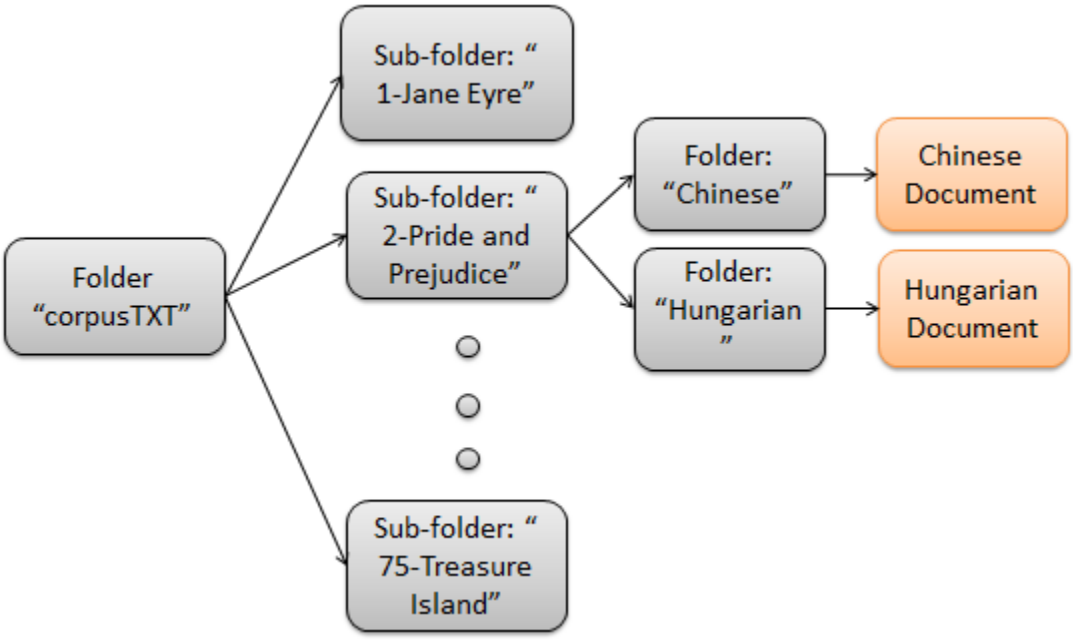

Figure 8: Documents' Organization in User Guide

Step3: Create Sentence-Level Parallel Corpus

Your only task in this step is to run bash script runHunalign.sh using the command

bash runHunalign.sh [location of folder corpusTXT]

The bash script runHunalign.sh normalizes (e.g. segment, stem…) the parallel documents automatically, and creates sentence-level parallel corpus for each parallel documents. This bash script outputs the eight files in each sub-folder, listed in the table below.

File Name	File Content
zh-sen	Chinese Doc segmented in word-level and sentence-level
hu-sen	Hungarian Doc segmented in word-level and sentence-level
hu-sen-stemmed	Hungarian Doc stemmed and segmented in word-level and sentence-level
Sentence-level-parallelDoc.txt	Chinese-Hungarian Sentence-level parallel corpus. The Hungarian sentences are in the original unstemmed form.
Sentence-level-parallelDoc.stemmed.txt	Chinese-Hungarian Sentence-level parallel corpus. The Hungarian sentences are in stemmed form.
align.txt	Hunalign's output. Line-number pairs of parallel sentences.
zh.aligned	The Chinese sentences from Sentence-level parallelDoc
hu.aligned	The unstemmed Hungarian sentences from Sentence-level parallelDoc.txt
hu.stemmed.aligned	The stemmed Hungarian sentences from Sentence-level parallelDoc.stemmed.txt

Table 5: Output Files after Running runHunalign.sh

Step4: Create Chinese-Hungarian Bilingual Dictionary

Your only task in this step is to run bash script runHundict.sh using the command

bash runHundict.sh [location of folder corpusTXT] [your choice of output location]

This bash script creates a bilingual dictionary for each parallel document, and combines the dictionaries to produce the final dictionary. The outputs of this bash script are listed in the table below. File names with * are output in [your choice of output location]. File names without * are output in each subfolder.

File Name	File Content
*HundictSummary.txt	For each parallel documents: name of documents, word count of Hungarian/Chinese Documents, number of word pairs resulted,
*Zh_hu_dict_DICE0.2_iter5 .sort.filtered.dup	The final dictionary (after combining dictionaries for all parallel Documents). May contain duplicated word pairs.
*Zh_hu_dict_DICE0.2_iter5 .sort.filtered	The final dictionary without duplicated word pairs, but no similarity scores.
Zh_hu_dict_DICE0.2_iter5 .sort	The sorted dictionary created by Hundict out of each parallel documents
Potential_name_dice0.2_ln11 .sort	Potential name pairs in zh_hu_dict_DICE0.2_iter5.sort
Zh_hu_dict_DICE0.2_iter5 .sort.filtered	The sorted dictionary created by Hundict out of each parallel documents, which potential name pairs removed

Table 6: Output Files after Running runHundict.sh

Notice that the –iter and dice are set as 5 and 0.2 respectively. You are free to go into the bash file and change the options.

References

[1] Grime, B. The Ethnologue (14th Edition). *SIL*. Print. 2003.

[2] Koehn, P. Europarl: A Parallel Corpus for Statistical Machine Translation. *In Proceedings of Machine Translation Summit.* Web. 2005. < http://mt-archive.info/MTS-2005-Koehn.pdf>.

[3] Chiang, D. A Hierarchical Phrase-Based Model for Statistical Machine Translation. *In Proceedings of Association for Computational Linguistics.* Web. 2005. <http://acl.ldc.upenn.edu/P/P05/P05-1033.pdf>.

[4] Marcu, D., Wong, W. A Phrase-Based, Joint Probability Model for Statistical Machine Translation. *In Proceedings of the Conference on Empirical Methods in Natural Language Processing.* July 2002. Web. <http://acl.ldc.upenn.edu/W/W02/W02-1018.pdf>.

[5] Haruno, M., Yamazaki, T. High-Performance Bilingual Text Alignment Using Statistical and Dictionary Information. *In Proceedings of Association of Computational Linguistics.* Web. 1996. <http://acl.ldc.upenn.edu/P/P96/P96-1018.pdf>

[6] Guo, J., Liu, J., Walsh, M., Schmid, H. Class-Based Language Models for Chinese-English Parallel Corpus. *In Proceedings of the 14th Computational Linguistics and Intelligent Text Processing.* Web. Mar. 2013. <http://link.springer.com/chapter/10.1007/978-3-642-37256-8_22#page-2>

[7] Yamada, K., Knight, K. A Syntax-based Statistical Translation Model. *In Proceedings of the 39th Conference of the Association for Computational Linguistics.* Web. 2001. <http://acl.ldc.upenn.edu/P/P01/P01-1067.pdf>

[8] Charniak, E., Knight, K., Yamada, K. Syntax-based Language Models for Statistics Machine Translation. *In Proceedings of the Machine Translation Summit IX.* Web. 2003. <http://cs.brown.edu/research/pubs/pdfs/2003/Charniak-2003-SBL.pdf>

[9] Varga, D., Halacsy, P., Kornai, A., Nagy, V., Nemeth, L., Tron, V. Parallel Corpora for Medium Density Languages. *Proceedings of Recent Advances in Natural Language Processing.* Web. 2005. <http://www.kornai.com/Papers/ranlp05parallel.pdf>.

[10] Brown, P., Lai, J., Mercer, R. Aligning Sentences in Parallel Corpora. *In Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics.* Web. 1991. <http://delivery.acm.org/10.1145/990000/981366/p169-brown.pdf?ip=94.27.196.104&acc=OPEN&key=1B55DF923F77674F55057ED4F3766CA0&CFID=307853574&CFTOKEN=30548227&__acm__=1365034480_6fd22ca14fec5bab312c8fed43f64762>

[11] Lafferty. J., McCallum, A., Pereira, F. Conditional Random Field: Probalilistic Models for Segmenting and Labeling Sequence Data. In Proceedings of the 18th International Conference on Machine Learning. Web. 2001. < http://www.cis.upenn.edu/~pereira/papers/crf.pdf>

[12] Tadic, M. Building the Croatian-English Parallel Corpus. *In Proceedings of the Second International Language Resources and Evaluation Conference*. Web. 2000. <http://gandalf.aksis.uib.no/non/lrec2000/pdf/119.pdf>

[13] Tseng, P., Chang, G., Andrew, G., Jurafsky, D., Manning, C. A Conditional Random Field Word Segmenter. *SIGHAN of the Association for Computational Linguistics.* Web. 2005. <http://nlp.stanford.edu/pubs/sighan2005.pdf>

[14] Halacsy, P., Kornai, A.,Nemeth, L., Rung, A., Szakadat, I., Tron, V. Creating open Language Resources for Hungarian. *In Proceedings of Language Resources and Evaluation Conference.* Web. 2004. < ftp://ftp.mokk.bme.hu/LDC/doc/lrec04szsz.pdf>

[15] http://mokk.bme.hu/resources/huntoken/

[16] Tiedemann, J. Parallel Data, Tools and Interfaces in OPUS. *In Proceedings of the 9th International Conference on Language Resources and Evaluation.* Web. 2012. <http://opus.lingfil.uu.se/>

[17] Resnik, P., Smith, N. The Web as a Parallel Corpus. *Computational Linguistics*, 29(3). Print. Sep. 2003.

[18] Moore, R. Fast and Accurate Sentence Alignment of Bilingual Corpora. *In Proceedings of the 5th Conference of the Association for Machine Translation in the Americas.* Web. 2002. <http://research.microsoft.com/pubs/68886/sent-align2-amta-final.pdf>

[19] Menezes, A., Richardson, S. A Best-first Alignment Algorithm for Automatic Extraction of Transfer Mapping from Bilingual Corpora. *DDMT Workshop, Association of Computational Linguistics.* Web. 2003. <http://acl.ldc.upenn.edu/W/W01/W01-1406.pdf>

[20] Grefenstette, G., Tapanainen, P. What is a Word, What is a Sentense? Problems of Tokenization. *In Proceedings of the 3rd Conference on Computational Lexicography and Text Research.* Web. April 22nd, 1994. <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.31.8947>

[21] Palmer, D., Hearst, M. Adaptive Sentense Boundary Diambiguation. *In Proceedings of Neuro Linguistic Programming*. Web. 1994. <http://www.aclweb.org/anthology-new/A/A94/A94-1013.pdf>

[22] Sproat, R., Shih, C. A Statistical Method for Finding Word Boundaries in Chinese Text. *Computer Processing of Chinese and Oriental Languages*, Vol4, No.4. Print. Mar, 1990.

[23] Xue, N. Chinese Word Segmentation as Character Tagging. International Journal of Computational Linguistics and Chinese Language Processing. Web. 2003. <http://www.aclweb.org/anthology-new/O/O03/O03-4002>

[24] Sun, W. A Stacked Sub-Word Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging. *In Proceedings of the Association of Computational Linguistics.* Web. 2011. <https://www.aclweb.org/anthology-new/P/P11/P11-1139.pdf>

[25] Tron, V., Gyepesi, G., Halacsy, P., Kornai, A., Nemeth, L., Varga, D. Hunmorph: Open Source Word Analysis. *In Proceedings of the Association of Computational Linguistics.* Web. 2005. <ftp://ftp.mokk.bme.hu/LDC/doc/acl05software.pdf>

[26] Sztaki Dictionary. Computer and Automation Research Institute, Hungarian Academy of Sciences (MTA-SZTAKI). Budapest, Hungary.

[27] Zhang, Y. Python Tool: Traditional to Simplified Chinese Converter. Web. Mar. 2008. <http://frcchang.blogspot.hu/2008/03/python-tool-traditional-chinese-to.html>

[28] Kornai, A., Makrai, M. A 4lang Fogalmi Szotar. *In Proceedings of the Hungarian Conference on Computational Linguistics.* Pp. 62-70. Print. 2013.

[29] Brants, T., Franz, A. Web 1T 5-gram Version 1. *Linguistic Data Consortium.* Web. 2006. <A. http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2006T13>

[30] Nie, J., Simard, M., Isabelle, P., Durand, R. Cross-Language Information Retrieval Based on Parallel Texts and Automatic Mining of Parallel Texts from the Web. *In Proceedings of the 22nd Annual International ACM-SIGIR*. Web. 1999. <http://www.iro.umontreal.ca/~nie/Publication/nie-sigir99.pdf>

[31] Potthast, M., Barron-Cedeno, A., Stein B., Rosso, P. Cross-Language Plagiarism Detection. Language Resources and Evaluation. Pp. 1-18.Web. Jan., 2010. <http://www.uni-weimar.de/medien/webis/publications/papers/stein_2011b.pdf>

[32] Kay, M., Roscheisn, M. Text-Translation Alignment. *Technical Report P90-00143, Xerox Palo Alto Research Center.* Web.1988. < http://acl.ldc.upenn.edu/J/J93/J93-1006.pdf>

[33] Fung, P. A Statistical View on Bilingual Lexicon Extraction – From Parallel Corpa to Non-parallel Corpora. In Proceedings of the 3rd Conference of the Association for Machine Translation in the Americas. Web. 1998. <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.54.5787>

[34] Yu, K., Tsujii, J. Extracting Bilingual Dictionary from Comparable Corpora with Dependency Heterogeneity. In Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics. Web. 2009. <http://www.newdesign.aclweb.org/anthology-new/N/N09/N09-2031.pdf>

[35] Yasuda, K., Sumita, E. Building a Bilingual Dictionary from a Japanese- Chinese Patent Corpus. In Proceedings of the Computational Linguistics and Natural Language Processing Conference. 2013. Web. < http://link.springer.com/content/pdf/10.1007%2F978-3-642-37256-8_23>

[36] Linguistic Data Consortium, University of Pennsylvania < http://projects.ldc.upenn.edu/Chinese/LDC_ch.htm>

[37] Stamatatos, E., Fakotakis, N., Kokkinakis, G. Automatic Extraction of Rules for Sentence Boundary Disambiguation. In Proceedings of the Workshop in Machine Learning in Human Language Technology, Advance Course on Artificial Intelligence. Web. 1999. < http://www.ling.gu.se/~lager/Mutbl/Papers/sent_bound.ps>

[38] Francis, W., Kucera, H. Brown Corpus Manual. *Department of Linguistics, Brown University*. Web. 1979. < http://khnt.aksis.uib.no/icame/manuals/brown/ >

[39] O'Neil, J. Do Things with Words. Web. Oct 29, 2008. <http://www.attivio.com/blog/57-unified-information-access/263-doing-things-with-words-part-two-sentence-boundary-detection.html>

[40] Reynar, J., Ratnaparkhi, A. A Maximum Entropy Approach to Identifying Sentence Boundaries. *In Proceedings of the Fifth Conference on Applied Natural Language Processing. Web.* 1997. <http://www.aclweb.org/anthology-new/A/A97/A97-1004.pdf>

[41] Palmer, D. An Adaptive Sentence Segmentation System. *UC Berkley Master thesis*. Web. Dec, 1994. <http://digitalassets.lib.berkeley.edu/techreports/ucb/text/CSD-94-846.pdf>

[42] Zséder, A. SZTAKI. <https://github.com/zseder/hundict>

[43] Pearson, K. On the Theory of Contingency and Its Relation to Association and Normal Correlation; On the general Theory of Skew Correlation and Non-Linear Regression. Cambridge University Press. Print. 1904.

[44] Sørensen, T. A Method of Establishing Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species and its Application to Analyses of the Vegetation on Danish Commons. 1957.

[45] Lee, D. Measures of the Amount of Ecologic Association Between Species. *Ecology, Vol. 26, No. 3. pp. 297-302.* Print. Jul. 1945.

[46] Charniak, E., Blaheta, D., Ge, N., Hall, K., Hale, J., Johnson, M. BLLIP 1987-89 WSJ Corpus Release 1. 2000. Linguistic Data Consortium, Philadelphia. Web. 2000.
< http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2000T43>

[47] Vonyó. A. University of Pannonia. Web. Oct. 1999.
< http://almos.vein.hu/~vonyoa/SZOTAR.HTM >