THE ENHANCEMENT OF MACHINE TRANSLATION FOR LOW-DENSITY LANGUAGES USING WEB-GATHERED PARALLEL TEXTS

Michael Augustine Gaylord Mohler

Thesis Prepared for the Degree of

MASTER OF SCIENCE

UNIVERSITY OF NORTH TEXAS

December 2007

Mohler, Michael Augustine Gaylord, The enhancement of machine translation for low-density languages using Web-gathered parallel texts. Master of Science (Computer Science), December 2007, 61 pp., 12 tables, 9 illustrations, bibliography, 25 titles.

The majority of the world's languages are poorly represented in informational media like radio, television, newspapers, and the Internet. Translation into and out of these languages may offer a way for speakers of these languages to interact with the wider world, but current statistical machine translation models are only effective with a large corpus of parallel texts – texts in two languages that are translations of one another – which most languages lack.

This thesis describes the Babylon project which attempts to alleviate this shortage by supplementing existing parallel texts with texts gathered automatically from the Web -- specifically targeting pages that contain text in a pair of languages. Results indicate that parallel texts gathered from the Web can be effectively used as a source of training data for machine translation and can significantly improve the translation quality for text in a similar domain. However, the small quantity of high-quality low-density language parallel texts on the Web remains a significant obstacle.

ACKNOWLEDGMENTS

I would like to thank everyone who has contributed to the Babylon project, to the creation of this thesis, and to the progress of my life in general.

To Rada, whose patience, suggestions, and time have most contributed to the completion of this project and to my continued adequacy in the graduate department. I would never have completed this thesis without your encouragement and the gentle guilt that comes from the fear of letting down someone whom you respect and admire. Thank you!

To Kevin Scannell, Christian Loza, and Cameron Palmer for contributing to this project by providing monolingual seed data, Bible translations, code samples, brainstorming sessions, and late nights fighting with Moses. Thank you!

To Drs. Tarau and Chen, for taking the time to attend my defense, serve on my committee, and rake me over the coals. Thank you!

To my colleagues in the Language and Information Technologies groups at both UNT and IPN, for your ideas, support, camaraderie, and inspiration as I try to work my way into the research community. Thank you!

To my parents, Artie and Gaylord Mohler, for always believing that I could accomplish anything. For being there to bail me out when I get in over my head. For giving me free reign to explore the wonders this life has to offer. Thank you!

To all my friends and family, for the countless contributions you have all made to my life – the ones that I often overlook. For never giving up on me. For freeing me from myself. Thank you!

"May the road rise up to meet you; may the wind be always at your back; may the sun shine warm upon your face and the rains fall softly on your fields; and until we meet again, may God hold you in the palm of his hand."

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1. The Problem

1.1.1. Low-Density Languages

Every language in use today can be placed into one of three categories according to the exposure the language has in the realms of business, entertainment, and technology, and for my purposes, the strongest emphasis is placed on the relationship the language has with technology. The languages with the highest exposure (such as English, Spanish, Chinese, Standard Arabic, etc.) are considered high-density languages. These languages are used globally, have an extensive presence on the Web, and are well-integrated into technology-related tools. Medium-density languages (such as Hungarian or Dutch) have far less exposure than high-density languages, but enough resources are available that they are not in danger of being smothered by the higher-density languages. Finally, there are low-density languages which are of the subject of my current interest. The overwhelming majority of the world's thousands of living languages fall into this category. I here define low-density languages (alternatively called minority languages) to be languages that do not have a large presence on the Web or in other high-tech media. Let it be granted that these categories are somewhat loosely defined but I propose that they remain useful in defining the problems facing many of the world's languages [13].

Put simply, languages without technological resources are in danger of extinction. For instance, there are regions of the world in which all news is transmitted through English-only media by way of radio, television, or Internet. In these areas, anyone with an interest in world news is required to learn English. Whatever the relative merits of learning a global language like English, situations such as these make the disappearance of a indigenous languages much easier. It is often less advantageous financially, politically, or socially to learn and pass on an indigenous language than to learn and pass on a more global language. The best method at our disposal for preventing a monopoly of high-density language usage is to make tools and resources available to speakers of low-density languages in order to reduce their societal dependence upon high-density languages. With the right resources, they can encounter the global village on their own terms and in their own tongue [5]. Providing these resources is an ideal task for automatic machine translation.

1.1.2. Machine Translation

Machine translation is one of the oldest subfields of computer science. From the outset, the benefit of applying computational power to machine translation was obvious. It is a largely repetitive task (e.g. dictionary lookup), it is time consuming for human annotators, and it has been a necessary work since the very advent of language. In the 1950s, machine translation received a lot of attention due to early successes in automatically translating Russian interceptions. However, time and money spent on rule-based machine translation failed to yield the results researchers expected, and until the advent of more powerful computers, machine translation remained quietly on the shelf. With more powerful computers, a new machine translation paradigm emerged in which translation models could be built for a language pair with little to no *a priori* knowledge about the languages. The models were built by analyzing existing translations, and using statistical characteristics of these translation pairs, it became possible to translate text from one language to another with a high degree of confidence [7].

While there is still room for improvement in machine translation between high-density languages like English and French, the task of building a human-quality machine translation system for low-density languages is even further from being fully realized. The bottleneck in this case is a lack of sample translations in the form of sentence-aligned texts [12]. It is in the creation of these parallel texts that I am primarily concerned.

1.1.3. Parallel Texts

Sample translations are the building blocks of natural language translation. Let us consider the case of a classical translator – a person who encounters a group of people who speak a language with which he is unfamiliar. This translator might begin to converse with an individual who speaks the unfamiliar language by gesturing to physical objects and using the terms associated with them - *man*, *woman*, *tree*, *rock*. Once these simple terms are understood, more robust language features can be observed. Perhaps the translator discovers that whenever the word *man* is said in a sentence, it always occurs at the beginning. Eventually, this may lead him to an understanding of how subjects and objects are related in the unfamiliar language and later to still more advanced features of the language.

In a statistical machine translation system, the basic needs are the same. The system needs to see how a familiar idea is expressed in an unfamiliar way by observing the same terms, phrases, and sentences in both languages. This is accomplished in a machine translation environment by training the system on properly-aligned parallel texts. When building a machine translation system for a language with few speakers and even fewer machine-readable resources, it becomes even more crucial to find parallel texts because the system is starting from the same position as the classical translator - learning which word is used for rock and how that word is used in sentences.

1.2. My Solution

1.2.1. Existing Parallel Texts

I would like to have as many translation examples as possible covering as wide a range of topics as possible. Learning about the word *rock* is not very useful when the topic of conversation moves to communism, music, or love. Fortunately, there is often a starting point in building parallel texts. Missionary zeal has unknowingly furthered the cause of machine translation by producing religious texts in translation for a sizable portion of the world's languages; these texts even have the fortuitous side-effect of being verse-aligned.

However, Biblical texts remain somewhat dated in language, and their subject matter is narrowly focused. For this reason, Biblical texts alone are insufficient for a robust statistical translation model. I have attempted to supplement Biblical parallel texts by finding parallel texts written or compiled on the Web that explore more modern themes and use more modern terminology.

1.2.2. Automatically Discovered Texts Found by Crawling the Web

In the discussion by Varga et al. [25] regarding parallel texts for medium-density languages, it was pointed out that many of the best resources for building parallel texts – literary texts, religious texts, corporate reports, movie captioning, international laws, and software manuals – are much easier than Web data to acquire, align, and use as parallel texts. For low-density languages, the difficulty in obtaining clean and usable Web data is just as severe, but there is, in addition, a lack of these other resources. This is often due to the existence of more prevalent languages being used in law courts, in classrooms, on the radio, and on television. Therefore, parallel texts from the gathered Web remain one of the few sources of parallel texts that automatic and inexpensive methods can hope to harvest.

For languages with scarce resources, the Web pages written in that language are few, and the pages in that language with translations into a more widely spoken language are even more limited. We must ask ourselves what types of pages are likely to have translations that are usable by this system? Perhaps a business or agency will target a local community and provide information in the local language alongside a high-density language. Perhaps some blogger will translate a poem, song, or story from a low-density language original in order to share it with friends or colleagues who do not speak the language. Perhaps a stray quote is used and immediately translated. The content of the text is not my present concern. I am concerned with how I may find this text.

It appears that I am looking for two basic types of translations - Web sites with pages translated into multiple languages and pages with small snippets of text in a low-density language. To the best of my knowledge, all Web-based parallel text discovery systems to date have focused entirely on the first category of pages which I believe to be ill-suited to building texts for languages with few speakers or scarce resources.

1.3. My Experiment - the Babylon System

To meet to goal of enhancing the quality of machine translations for low-density languages, the Babylon System was commissioned. The Babylon System searches the Web for parallel texts in a manner specifically suited to a low-density/high-density language pair. In the interest of adaptability to a wide variety of low-density languages, the system was designed to need very few resources – namely a short sample of monolingual text. A bilingual dictionary is useful in assuring the quality of the resultant parallel texts but may be omitted if none exists. Once a parallel text is discovered, Babylon uses it to improve the quality of translation by supplementing it with existing parallel texts.

For my experiment in building parallel texts for languages with scarce resources, I have selected two languages spoken in Latin America - Quechua, which is spoken by around 10 million people in Peru, Bolivia, Ecuador and a few other regions of South America, and Nahuatl, spoken by around 1.5 million people in Mexico and central America. As with many languages that are not in heavy modern use, there are many dialects of these two languages. For the purpose of these experiments, I have not targeted any particular language dialect, but some bias may have necessarily occurred during the language identification step. As the areas where these languages are spoken are majority Spanish-speaking, Spanish is the high-density language chosen for these experiments. The experiment was first run for Spanish-Quechua and then rerun in its entirety for Spanish-Nahuatl. For more information on building linguistic tools in Quechua, refer to Monson et al. [14] in which a team of computational linguists in tandem with native Quechua speakers manually produced a parallel corpus for Quechua in addition to several morphological tools and a rule-based translation system.

CHAPTER 2

RELATED WORK

Much work has been carried out in an effort to create parallel texts using data mined from the Web. Before exploring the methods used in the Babylon system, let us first highlight several of the systems that have come before and which have influenced the evolution of Babylon. The systems discussed here are STRAND, BITS, PTMiner, the Parallel Text Identification System, WebMining, and a Document Object Model-based system. Additionally, I describe the Web crawling methodology that most influenced Babylon - the Language Specific Web Crawl.

2.1. STRAND

STRAND [17] is arguably the seminal work on building parallel texts using resources gathered from the Web. The fundamental hypothesis of the work is that parallel Web pages can be found with little or no *a priori* knowledge about the language pair. In fact, Resnik proposed that the structure of the Web page is the most useful feature in determining whether or not a pair of pages constitute a translation.

The original STRAND implementation queries AltaVista to find *parent* and *sibling* pages by searching for hyperlinked text that would indicate a translation. Sibling pages are pages in one language that themselves contain a link to the same page in another language. Parent pages are pages that contain links to both translations of the same text. For instance, a sibling page might have the text "Spanish Version" and a parent might have links tagged with "Spanish" and "English" to indicate a translation. Later this method for finding candidate pages was supplemented with a crawler [19]. Sibling and parent pages link to their candidate matches directly, so the original STRAND system considers only those few pages, but with the introduction of the crawler, every page on the Web site is a potential match.

In the original implementation, no language identification takes place, and Web sites with neither of the target languages often produced false positives. With the addition of language identification, STRAND reduces the number of candidate pairs from $(N+M+k)^2$ to (N*M) where N is the number of pages in language L_1, M is the number of pages in language L_2, and k is the number of pages in neither language.

Candidate generation was further improved in some implementations [19] by performing a URL matching algorithm which takes into account the language features in URLs. For instance, a pair of translated texts might appear as "index_en.htm" and "index_sp.htm" or "/spanish/file.html" and "/english/file.html". In one experiment, English-related features of a URL (e.g. "en", "english", etc.) were replaced with corresponding features for the other language in hopes of generating a valid URL among the candidate pages. In another larger-scale experiment, language-related features were removed entirely from the URL string, and pages whose resulting strings were equivalent were considered likely candidates.

Once the candidate generation stage is complete, STRAND passes each pair through a structure filter. A common practice among webmasters of multilingual sites is to reuse Web page templates when maintaining multiple translations of the same content. Frequently, the same header tags, tables, images, and display tags can be found on both pages and in roughly the same order. STRAND makes use of this structural similarity to find translation pairs without the need for any content analysis. The system converts a Web page into a sequence of tags, where opening HTML tags, closing HTML tags, and text are mapped to STRAND-specific tags (e.g. $< HTML >$ maps to [BEGIN:HTML] and text is converted to [CHUNK:length] where length is the non-white-space length of the string). Attributes of the markup tags are converted to text chunks.

Given this sequence of tags for a pair of Web pages, the original version of STRAND runs the *diff* command (common in most *nix environments) which is a simple dynamic programming algorithm for finding an optimal line by line alignment between two texts. Any pairs that have too many unmatched lines (>20%) are removed from consideration. Later versions include a more robust dynamic programming alignment algorithm that measures the amount of both the non-shared material and the aligned non-markup chunks and takes into account the lengths of the aligned text chunks.

In three experiments searching for English-Spanish, English-French, and English-Chinese parallel texts, Resnik [19] reported near perfect precision (>= 98%) for all three and recall values of around 60%, 68.6% and 61% respectively. The Spanish experiment was evaluated informally using the judgment of the author. In the French experiment, 326 candidate pairs were selected randomly from a much larger set and were judged *good* or *bad*. Evaluation was conducted on the pairs for which the annotators were in agreement. The Chinese experiment was conducted similarly to the French experiment. When the system was enhanced by supplementing the manually set thresholds using decision tree-based machine learning and rerun on the English-French task, recall was raised to 84.1% while precision dropped to 95.8%.

2.2. BITS

The BITS system [12] differs from the STRAND system in being primarily interested in the content of the text rather than its structure. Additionally, the BITS system generates candidate pairs differently. The system is fed a few top level domains that are expected to have a high percentage of Web sites bilingual in the target languages. Since their experiment was for German-English, Ma and Liberman released their crawler onto the .de (Germany), .au (Austria) and .lu (Luxembourg) domains and found all Web sites within those domains. Given a list of Web sites, each site is crawled to a distance of 3 or 4 links from the entry page in order to filter out monolingual Web sites. After that, each Web site is divided into sets by language.

For each potential pair of Web pages, BITS computes a similarity value based upon the pages' content. For candidate pages A and B the similarity score is

$$sim(A, B) \leftarrow \frac{\#\ translation\ token\ pairs}{\#\ tokens\ in\ A}$$

A token in A that when translated is found in B is called a translation token pair. Cognates, anchors (i.e. non-changing text and symbols like numbers), and entries in an optional lexicon are the sources of translation token pairs for the BITS system. For each page A in language L_1, a page B in L_2 is found that has the highest similarity metric of any page in L_2. If this similarity metric is above a threshold (not defined in the literature), the pair passes the filter.

In their experiment, they manually selected 300 German and 300 English pages from 10 Web sites and found 240 translations. Ma and Liberman reported a precision and recall for this experiment of 99.1% and 97.1% respectively.

2.3. PTMiner

PTMiner [2] was designed to build a parallel text from the Web in order to contribute to the task of cross-language information retrieval. Like STRAND, PTMiner queries existing search engines (specifically AltaVista and Northern Light) for anchor text in order to find parent and sibling pages (see Section 2.1). Once these pages are found, they become starting points for a host crawler which visits every document on the Web site.

Once the PTMiner system has produced a list of all document URLs on a Web site, it performs a "pair scan" which begins with the assumption that webmasters tend to name bilingual files according to some predictable naming convention. For each language, the system uses a manually created list of language-specific features (as described in Section 2.1) to find likely parallel pages. When any feature in the list is found, it is replaced with the complementary feature (e.g. "en" with "fr"), and the system determines if any file on the Web site matches the new URL. If so, the pair are kept as candidates for parallelism. PTMiner then filters out any pairs that are not in the target languages or whose difference in file length is above a certain threshold. Since this varies by language pair (i.e. English-French texts are closer in length than English-Chinese), this threshold is set on a case-by-case basis.

PTMiner was used to build a corpus of about 14,000 English-French texts and about 15,000 English-Chinese texts. The parallel texts were aligned using a combination of length similarity and cognate matching. These parallel texts were then used for statistical translation model training and in cross-language information retrieval. In the machine translation experiment for the English-Chinese corpus, two hundred words were selected at random from the training source and the most probable translation (in a lexicon) was matched with the output translation for that word. PTMiner showed a precision of 77% and recall of 81.5% for the machine translation task.

2.4. Parallel Text Identification System

The Parallel Text Identification System (PTI) [3] was developed by Chen, Chau, and Yeh to detect whether a pair of pages are meant to be translations of one another. PTI makes use of two modules to identify parallel texts. Like other systems described in this chapter, PTI uses a URL comparison module in which parallel sets of language-related features are used in an attempt to map a URL to another URL by substitution. To find pages in which the URL naming convention breaks down, PTI relies upon a content matching module.

The content module makes use of a bilingual word list to generate a vector of term features. Terms which are translations of one another (according to the word list) are mapped to the same term feature. The resulting vectors (one for each document) are then compared and a similarity score is calculated for the pair. The literature [3] proposed several similarity metrics including Euclidian distance, inner product, cosine similarity, Dice coefficient, and Jaccard coefficient. Eventually, Chen, Chau and Yeh selected the Jaccard coefficient for use in their experiments, because it ignores joint absences in the feature vector and is concerned with the proportion of features in either vector that occur in both vectors. The equation for Jaccard coefficient is reproduced below where p is the vector length, and X and Y are the term frequency vectors:

$$Jaccardcoefficient = \frac{\sum_{i=1}^{p} x_i y_i}{\sum_{i=1}^{p} x_i^2 + \sum_{i=1}^{p} y_i^2 - \sum_{i=1}^{p} x_i y_i}$$

PTI accepts as valid any pair with a Jaccard coefficient of 0.9 or greater.

In their experiment, they made use of the commercially available crawler WebZip to download 427 documents from the Hong Kong governmental Web site which contains pages in both Chinese and English. The site contains a high proportion of language-related outlinks (e.g. "English version") which makes manual evaluation tractable. Of their 427 documents, they manually detected 187 parallel pairs using the page titles and link features. They reported a precision and recall of 93% and 96% percent respectively using the PTI system.

2.5. WebMining

The WebMining project [24] set out to build parallel corpora using very few resources and no supervision. They manually create a list of Web pages by searching AltaVista for low-density language content (Basque in one case) as well as by using all servers in the region's .edu domain (Catalan in the other case). For each Web site, their system downloads every page on the site and compares it to all previously downloaded pages.

Each pair is passed through four filters. First, the two must be of similar length (file size, number of paragraphs, and number of tags). Second, they must be in different languages as well as in one of the languages of interest. Third, the edit distance between the two pages (when converted to a format similar to that mentioned by Resnik [17] must be below a certain threshold (not defined in the literature). Finally, the content must be similar. Any pair that fails any filter is discarded as a potential match.

The content similarity filter is performed without the use of a lexicon, by attempting a sentence alignment and accepting or rejecting the pair based upon the alignment probability metric. Each word to word alignment probability is given some initial value. Accepted pairs are fed back into the system to improve the probability estimation.

In their first experiment, they selected eight Web sites with multilingual content (the languages were English, French, Spanish, German, Italian, Portuguese, and Catalan). They manually found 652 parallel Web pages of the 1,087 total Web pages on the sites. Their system, using only the structural filters, found the parallel pages with an overall precision and recall of 89% and 78% respectively. In their second experiment, six English-Spanish Web sites (out of the original eight) were selected, and utilizing the full system (both the structural and content-based filters) improved the precision from 85% to 96% and the recall from 82% to 89%.

2.6. DOM Tree Alignment

Shi et al. [20] describe a system for creating a parallel corpus using a DOM tree alignment model. A DOM (Document Object Model) tree is a representation of a properly structured Web page in the form of a tree. Like STRAND, the primary concern of the DOM alignment model system is the structure of the Web page.

They intended to improve upon the model in two ways: by improving throughput by downloading fewer Web pages and by improving the quality of the mined data by aligning parallel text chunks rather than aligning text at the page level.

Given a root Web page, the hyperlinks are searched for associated text containing language-specific features (as described in the URL matching paragraph of Section 2.1). If a language-specific hyperlink is found for both target languages, the two Web pages they are referring to are downloaded and added to the candidate pair list. The links of these two pages are then analyzed for parallel language links and followed iteratively until no more pages are found. Parallel text chunks are discovered by optimally aligning the trees using a dynamic programming algorithm that begins by finding optimally aligned subtrees. These parallel chunks are later aligned at the sentence level using more conventional alignment tools.

Each candidate pair must pass a filter for file length, HTML tag similarity, and sentence alignment score. The HTML tag similarity is the edit distance between the tags of both pages when concatenated. The sentence alignment score is the ratio of aligned sentences to total number of sentences. A threshold must be passed for each of these filters (not defined in the literature).

In their experiment, Shi et al. downloaded 300,000 Chinese URLs from Yahoo Web directories as starting points for their system. This resulted in 63,214 parallel English-Chinese Web documents. They reported that DOM tree alignment model yielded an improvement from 93.5% to 97.2% over a simple URL pattern matching approach. Additionally, for the alignment task, they reported an increase of precision from 86.9% to 93.4% and an increase in recall from 79.4% to 86.6%.

2.7. Language-Specific Web Crawl

Somboonviwat et al. [23][22] describe a study in purposeful Web crawling. Their task was to find Web pages in the Thai language for the purpose of archiving a snapshot of Thai culture at a moment in time. They observed that searching Web sites in the top level domain .th was useful, but by limiting their search to this domain, they missed a sizable number of relevant Web pages, and so they needed additional methods to find Thai Web pages in other domains (e.g. .com or .edu or .jp). They predicted that the Web exhibits language locality when viewed as a graph. In other words, a page in the Thai language is more likely than an English-language page to link to other Thai Web pages. Resting on this assumption, they proposed a method for efficiently finding Thai Web pages in foreign Web space by a process they call a language specific Web crawl (LSWC).

They made three observations. First, following links from Thai pages resulted in an increased likelihood of finding more Thai Web pages. Second, it may be necessary to follow up to N non-Thai Web pages in order to find an additional Thai page. Finally, a server with a Thai Web page is likely to have other Thai Web pages. They propose the following strategy. A page with majority Thai content is considered a relevant page. Links that are further away from a relevant page than a certain threshold T are discarded. Links on irrelevant servers (i.e. a server that has not found a relevant page within a threshold S downloads) are discarded as well. URLs on relevant servers (servers with at least one relevant page) are given the highest priority. Then, the URLs with the shortest hyperlink distance from a relevant parent page are downloaded.

In their experiment, they performed a virtual Web crawl (on data downloaded from a conventional crawl at some point in time) so that the content would remain consistent for all crawls. They began their crawls from three popular Thai Web sites[1], and used the tool text_cat (see Appendix A.1) to perform language identification. The LSWC crawler was run with two different threshold levels (S=1, T=1 and S=3, T=5) and compared with several other crawling strategies – specifically those proposed in

[1]http://www.sanook.com, http://www.siamguru.com, and http://www.matichon.co.th

their 2005 study [22]. Their technique was shown to perform comparably with the other focused crawling techniques mentioned in the study and consistently outperformed a standard breadth-first search. After the first 4 million pages downloaded, both LSWC crawls achieved a coverage of approximately 93% of the total Thai pages in the virtual set, while breadth-first search found only around 55%.

CHAPTER 3

BABYLON PARALLEL TEXT BUILDER

3.1. Overview

Let us first define some terminology. I will refer to two categories of languages throughout - the minor language and the major language. The major language is, in general, a language that is widely spoken and has a relatively large presence on the Web. The minor language on the other hand is less common - usually a low- or medium-density language (cf. Section 1.1.1). In these experiments, the major language is Spanish and the minor language is Quechua for the first experiment and Nahuatl for the second.

Figure 3.1 describes the overall system flow of the Babylon Parallel Text Builder. Given a short monolingual text in the minor language as a starting point, I eventually produce a parallel text composed of sentence-aligned text in the minor and major languages. The process is described in detail below. In Section 3.2.1, I describe the method by which the monolingual text for the minor language is used to select a set of words with which to query Google. These queries will supply us with a collection of starting points from which to run the crawler. In Section 3.2.2, I describe how the crawler, starting at these seed URLs, searches for pages in the minor language and categorizes them as either having a large proportion of the language, being a candidate for finding an on-page translation, or being irrelevant to the search. In Section 3.2.3, I describe how the Web pages found by the crawler are then used as starting points on a second, shorter crawl to find their major-language counterparts. Each remaining minor language page has one or more major-language candidate pages. As described in Section 3.3, these pairs are run through a set of filters to determine which are most likely to be translations of one another based upon the URL, the structure of the page, and the content of the text. Finally in Section 3.4, the remaining file pairs are aligned to be used in the translation system. Throughout the system, thresholds have been defined that were selected before the experiment began and were not altered.

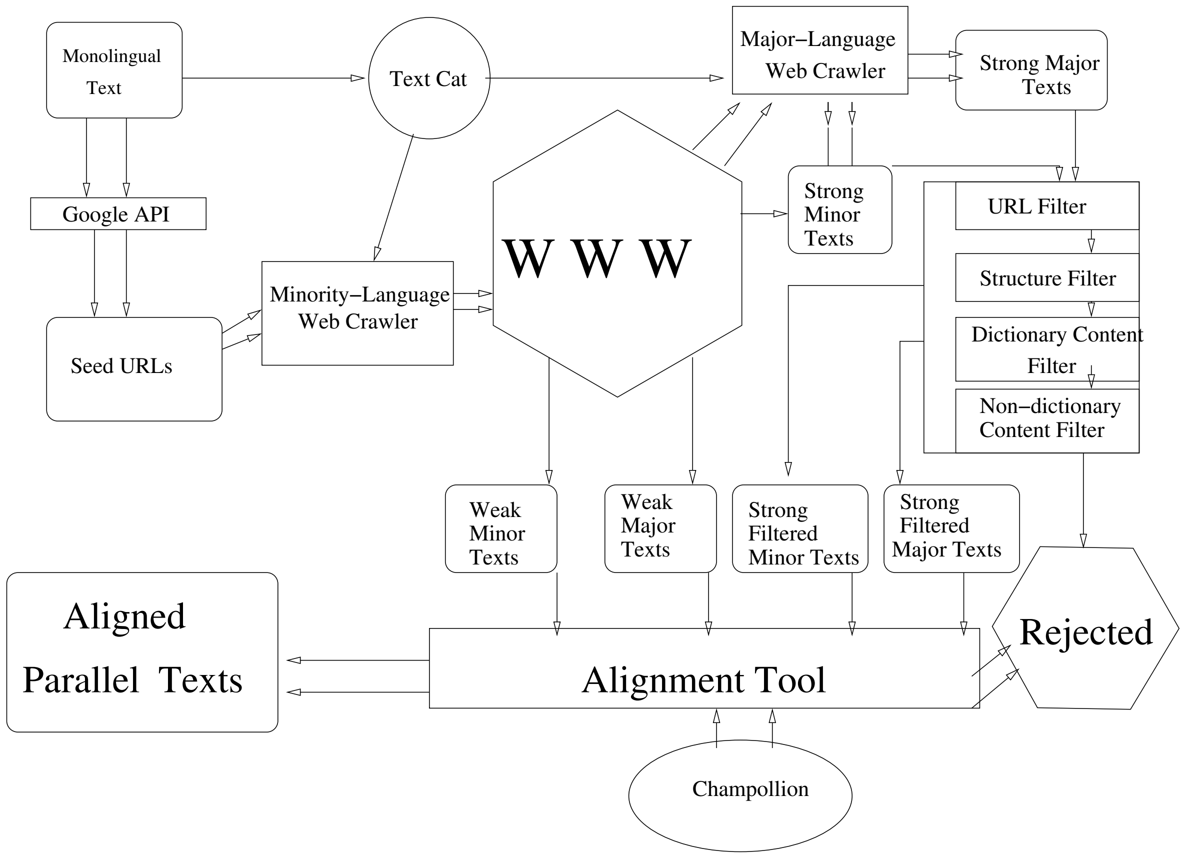

Figure 3.1. Parallel Text Builder Flow

3.2. Crawling

3.2.1. Google Seeding

The initial seeds are found by utilizing a (potentially very short) sample of text in the minor language as well as the Google API. The Google API allows a script to query Google and access the results. It has been provided by Google to researchers as

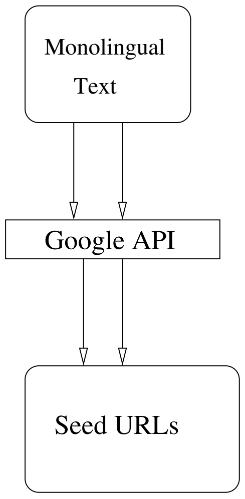

Figure 3.2. Seed URL Generation

a convenience. Given the monolingual sample, I clean the text (remove punctuation, add white-space, etc.), generate a list of all word tokens, and sort them in decreasing order by frequency. The word list is then divided into thirds – most common third, least common third, and middle third. A total of 1,000 words are selected randomly with an equal distribution from each set in hopes of seeding the crawler with a wide variety of domains. These 1,000 words are then fed individually to the Google API which return a listing of the top 10 Web pages that match that query. Any Web page that is found for any of the 1,000 words is included as a starting point for the crawler. For the Quechua experiment, around 5,200 starting pages were found. For the Nahuatl experiment, approximately 3,100 were found. A similar approach was used by Ghani et al. [8] to produce monolingual corpora for low-density languages.

3.2.2. Minor Language-Specific Crawl

There are relatively few Web pages that contain data in low-density languages like Quechua and Nahuatl. Therefore, standard crawling methods like a pure breadth-first search (BFS) will spend a lot of time and resources in parts of the Web that have

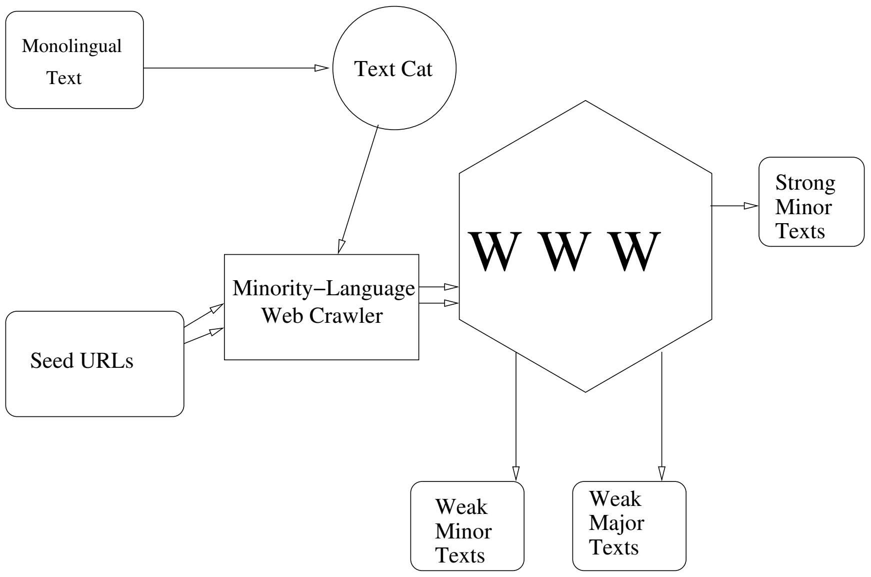

Figure 3.3. Minor Language Web Crawl

nothing to do with the targeted minor language. Based upon this observation and the recommendation by Somboonviwat et al. [23], I chose to tailor my crawler to the task of finding Web pages written in languages that will not be frequently encountered. In order to accomplish this, I have added to the crawler a memory of how many links have been followed since a page in the target language has been found. If a certain threshold has been met (a link distance of 5), then new links are not added to the queue.

The crawler performs a modified BFS where the seeds produced from the previous step are the initial contents of the queue. The crawler analyzes the language of each page and classifies it one of three ways - not relevant, *weak*, or *strong*. Weak pages are meant to describe the pages that contain some of the minor language, but are primarily written in the major language. These will be assumed to have a translation of the minor language on the same page. Strong pages are those that are primarily written in the minor language and so may be assumed to have a major-language translation on some other page. The crawler terminates after one million pages have been downloaded.

A page is broken into segments where non-display HTML tags and multiple space characters are the boundaries of the segments. A page is classified as weak if it contains at least 40% major-language segments and the minor-to-major language ratio is at least 0.15. If a page has at least 40% minor-language segments, it is classified as a strong page. Any pages that are too small (i.e. contain fewer than 200 characters or fewer than 50 tokens), or fail to fall into either of the above categories are rejected. The language identifier used in these experiments is text_cat (see Appendix A.1 for a description).

3.2.3. Major Language-Specific Crawl

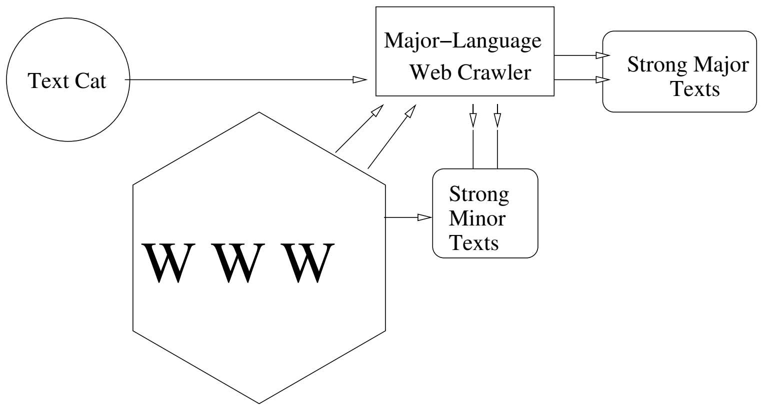

Figure 3.4. Major Language Web Crawl

Given a set of Web pages that appear to be primarily in the minor language (i.e. strong pages), I must now find all potential major-language pages that may be direct translations - henceforth called candidate pages. At this point, I operate under the assumption of locality – if a Web site contains two or more translations of the same text, they are usually close together on a graph of the site's hyperlink structure.

Recall the two norms followed by webmasters regarding multilingual page structure [18][19] (cf. Section 2.1) – sibling pages and parent pages. In these experiments, I focus primarily on finding candidate pages that are organized according to the sibling page/parent page paradigms. In order to find major-language counterparts to the strong minor-language pages already found in the minor-language crawl, I assume that all true translations are within *N* hyperlink steps from either the minor-language page itself or from the parent of the minor-language page (i.e. the page that linked to it in the previous crawl). I perform a BFS on the effective directed graph up to a total depth of *N*. In these experiments, I used *N*=2. At this point, the only criterion for being considered a candidate for translation is the distance from the starting minor-language page and the language identified. Therefore, a lot of candidate pages are generally found at this stage and later filtered out.

One enhancement was attempted at this point in the Quechua experiment, but abandoned in the Nahuatl experiment. I attempted to map the entire domain of each Web site of interest (i.e. Web sites on which have been found strong minor-language pages) so that rather than a directed graph (where the hyperlink contains directionality) I might be able to traverse the site as an undirected graph. When starting the Nahuatl experiment, I decided that the improvement observed was not worth the cost in bandwidth, disk space, and time spent crawling the site. One possible means of improving the efficiency of this step is to utilize a precomputed Web site map from some larger organization with crawlers dedicated to the task.

3.3. Filtering Pairs

After creating a collection of candidate pairs (strong minor-language pages paired with major-language pages), I attempt to filter out candidates that do not appear to be translations. It may be more appropriate to describe the process as filtering *in* candidate pairs, since I chose to accept for the next stage any pair that passes at least one of the following four filters: URL matching, structure matching, content matching without a dictionary, and content matching with a dictionary.

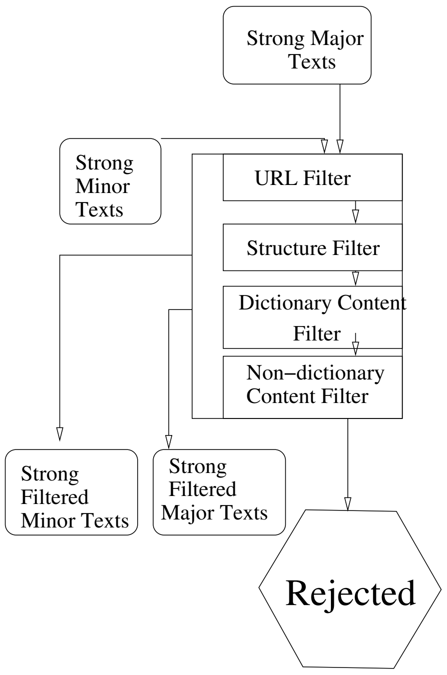
Strong Major Texts
Strong Minor Texts
URL Filter
Structure Filter
Dictionary Content Filter
Non–dictionary Content Filter
Strong Filtered Minor Texts
Strong Filtered Major Texts
Rejected

Figure 3.5. Filtering in Pairs

3.3.1. URL Matching

It has been observed in the development of many previous systems (cf. [15], [12], [3], [19], and [20]) that bilingual Web sites frequently make use of URL and directory structure naming conventions to reduce the difficulty of maintaining a large Web site. When searching for parallel data, I can utilize this tendency by looking at the similarity of URL names from among the candidate pairs.

Previous work [18] used a list of language-related strings (e.g. "english", ".de", "español") to generate alternative versions of the URL. For instance, a URL ending in "index.html" might match with the URL "index_sp.html" (see chapter 2). While this may work well when the target and source languages are known *a priori* and common abbreviations and language strings are known, I believe that it would become too unwieldy in the general case of searching for arbitrary languages. Therefore, other methods of exploiting URL parallelism must be sought.

In these experiments following the recommendation of Ma [12], I find the edit distance between the page's URL and that of its translation candidate. Two pages are considered to pass this stage of the filter if their edit distance is less than 5 characters (chosen arbitrarily) or is less than the length of the shorter language name (i.e. "quechua", "spanish", "nahuatl") and is less than 5% of the length of the URL itself. The result of this is a low recall and high precision since a lot of potentially valid naming changes are missed, and any URL pair that passes must have a very low edit distance.

3.3.2. Structure Matching

Our experiments utilize the fundamental idea of the STRAND structure matching algorithm with some slight modifications (see Section 2.1). Rather than using *diff*, I have implemented an edit distance algorithm that accepts any pair with an edit distance less than a threshold (in this case 4 characters and at least 1/4 of the total length of the page). The original STRAND system [17] did not take into account the length of the text chunks when attempting to find an alignment. In my edit distance implementation, I consider two lengths to be a match if the smaller is greater than 2/3 the length of the larger. A pair passes the structure filter if its edit distance is less than 4 and less than 1/4 of the smaller token sequence.

3.3.3. Content Matching

Despite the relative success of the STRAND project, Resnik [19] concedes that it is not desirable to ignore a Web page's content entirely. Structure- and URL-based approaches are useful only on the Web and, even then, only in the specific context of a markup language. A lot of useful parallel data may be found in .txt files, .pdf files,

.doc files, .ps files and so forth which would be missed entirely by systems that only consider markup language. This is supplemented in many systems with some type of content based filter (cf. [12], [19], [24]).

In the Babylon system, this is accomplished in two steps – one step matching content with a dictionary and one step matching content without the use of a dictionary. One of the stated goals of the Babylon project is to target languages with scarce resources, and since a machine-readable, robust, bilingual dictionary is a resource (and perhaps a costly one), some level of functionality must be possible without one. Without the dictionary, the content matching is, of course, limited, but it is still able to match some non-translated tokens (e.g. numbers, punctuation, proper names) and, to a lesser extent, some cognates.

The content filter is modeled on the PTI system [3]. The dictionary (if available) is read and a hash is created mapping each entry in the dictionary to a list of possible translations. Additionally, each word is always mapped to itself in case no other translation exists. Additionally, a stemmed version of this dictionary hash is created which contains the same entries and mappings, but only the first four characters of each token. For instance if "apple" maps to "manzilla" and "apple" in the first hash table, "appl" will map to "manz" and "appl" in the stemmed hash table.

For each minor-language document, a term frequency (tf) vector is created. The term frequency is the number of times that a term appears in a document, and so a tf vector is a representation of the number of times that each word appears in the document. Any terms found in the minor-language text are added to the hash table and mapped only to themselves. This allows the filter to find non-translated proper nouns in the texts. Then, each major-language candidate is read, translated one term at a time into the minor language (using the hash tables), and converted into two additional tf vectors (one for both the stemmed and non-stemmed hash tables). For two tf vectors representing two documents, it is possible to find a similarity score between the two documents. In this experiment, I calculated the Jaccard Coefficient (see Section 2.4) to indicate the content similarity between the two texts. The pseudocode below produces the Jaccard Coefficient where V and U are the tf vectors for the two documents. The Jaccard Coefficient is calculated twice – once with the stemmed tf vector and once with the unstemmed vector.

```
$prod, a, b \leftarrow 0$
for $i = 0\ to\ N - 1$ do
    $prod \leftarrow prod + V_i * U_i$
    $a \leftarrow a + V_i^2$
    $b \leftarrow b + U_i^2$
end for
if $a + b = prod$ then
    $answer \leftarrow 1$
else
    $answer \leftarrow prod/(a + b - prod)$
end if
```

The non-stemmed similarity is given a weight of 0.75 while the stemmed coefficient is weighted with a value of 0.25. The sum of these two values is the final effective score for the content filter. In this experiment, any pair with a similarity score greater than 0.2 is passed on to the next stage.

3.4. Alignment

The final stage of the Babylon pipeline is the alignment of the weak pages and strong minor- and major-language pages. All of the texts involved are separated into segments (based upon HTML tags and text spacing), and any segments that are classified by the language identifier text_cat (see Appendix A.1) as being neither in

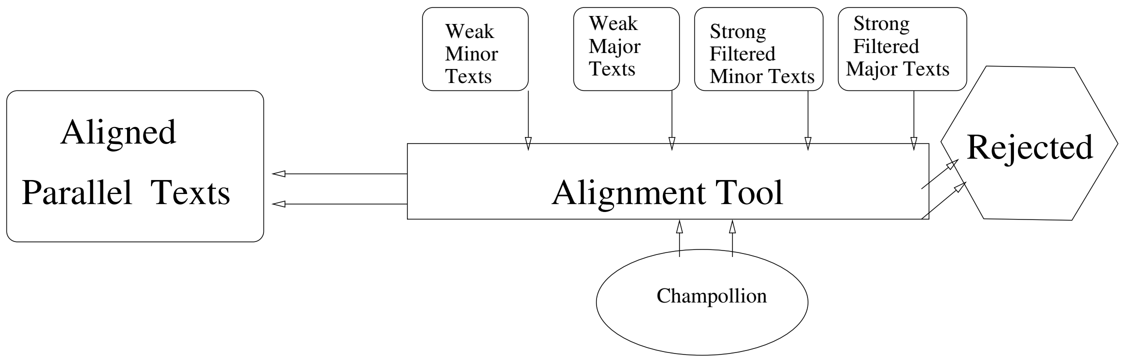

Figure 3.6. Final Parallel Text Alignment

the major language nor in the minor language are discarded. The weak pages are divided into two separate files. From this point on, the weak and strong file pairs are treated the same. The pairs of files are aligned by the alignment tool before being combined into a single parallel text.

Given the nature of this experiment (specifically the quality of text found on the Web), I decided that it was important to research available alignment tools and to select the tool that most closely met my needs. As shown by Singh and Husain [21], noise (such as adding or dropping sentences and paragraphs), extraneous markup, and split or merged sentences affect the performance of the alignment tools in different ways. Additionally, some of the tools evaluated performed significantly worse for some languages pairs than for others [4]. A perfect tool would need to perform well with noisy data and also perform similarly for any arbitrary pair of languages. Many of the candidate texts contained sentences that had no corresponding translation - particularly int the weak pages where only a small portion of the Web page was in translation. Eventually, I decided to use the tool Champollion [11] described in more detail in Appendix A.2.

Given the alignment output of Champollion, I perform one final filtering to account for high Web noise [24]. I give each alignment a score in the interval [0..1] based upon the number of one-to-one matchings and omissions returned. The following pseudocode defines the alignment score:

```
**if** no omissions **then**
    $answer \leftarrow 1$
**else**
    $answer \leftarrow one\_to\_one + (total - one\_to\_one - omissions)/(2 * total)$
**end if**
```

Alignments with no omissions are given the highest score. Beyond that, one-to-one alignments are preferred to one-to-many or many-to-many alignments. When more than one candidate major-language page is still present (as may be the case with strong minor-language pages), only the page with the highest score is matched with the minor-language page. All pages that do not meet a certain threshold are discarded. For these experiments, I used two thresholds - 0.3 and 0.65 labeled in Section 4.2 as Crawled and Hi-Crawled respectively.

CHAPTER 4

BABYLON MACHINE TRANSLATION EXPERIMENT

The Babylon Translation System takes as a starting point a collection of human translated texts. In this case, these texts are Biblical. They are high quality translations with over 30,000 verses, and they are aligned making them ideal for my task. Once additional parallel texts have been discovered by the Web crawl, filtered, and properly aligned, they can be used to enhance the performance of the system. After some cleaning (separating words from punctuation, removing special characters, etc.), the parallel texts are used to build a translation model with Moses (see Appendix A.3 for more information on Moses). The first step of my experiment was to generate language models for Spanish, Quechua, and Nahuatl. A language model is used to instruct the text generation stage of the translation system in rearranging an effective bag of translated words into a more natural sounding sentence in the target language. The word order is probabilistically determined based upon word orders observed in the sample texts. I generated language models using the full Bible for Spanish and Quechua and the New Testament for Nahuatl. N-gram counts were included in the language model for all n-grams up to 4.

Table 4.1. Experiment Size

	$Google$	$Min-Crawl(st/wk)$	$Maj-Crawl^1$	$Filtering$	$Alignment$
Quechua Pages	5,249	1,433/507	4,927	2,720	364
Nahuatl Pages	3,104	148 / 5,482	490	338	251

The Maj-Crawl and Filtering columns describe pairs of pages - the strong Quechua page paired with all candidate Spanish pages.

Table 4.2. Parallel Text Size

		Bible	NT	Crawled	Hi-Crawled
Quechua	Lines	31,095	7,949	5,485	2,034
	Words	484,638/747,448	119,490/191,645	87,398/99,618	22,593/24,711
(QU/ES)	Size	4.6MB/4.2MB	1.1MB/1.1MB	550KB/540KB	160KB/160KB
Nahuatl	Lines	N/A	7,949	3,774	2,054
	Words	N/A	187,744/191,645	28,917/84,791	14,202/34,824
(NAH/ES)	Size	N/A	1.5MB/1.1MB	190KB/480KB	110KB/210KB

4.1. Experimental Setup

I then built translation models using Moses, by combining the available Biblical texts with the texts automatically discovered by the crawler. The stated goal of the Babylon project was to determine how much improvement can be gained by adding crawled parallel texts to existing hand-translated texts (i.e. the Biblical texts). To answer this question, I built translation models for both language pairs, in both directions, for three different model categories – models trained on crawled texts only, on Biblical texts only, and on combinations of the two. I then used each of these models to translate both a crawled sample and one or more Biblical sample in order to test the effect of domain on translation models. Evaluation of the translation quality was performed using the implementation of BLEU included with the Moses toolkit. A discussion of BLEU can be found in Section 5.2, Appendix A.4, and Appendix B.

The translation models can be described as an $N*M$ cross product of the N types of crawled parallel texts, and M types of Biblical texts. The N crawled types are as follows: none, Crawled, and Hi-Crawled. The Crawled training set is the text that passed through the alignment stage with a score (see Section 3.4) above the threshold of 0.3, and the Hi-Crawled set is the resulting corpus when the threshold is raised to 0.65. The M hand-translated Biblical texts are: none, New Testament, Bible, Four New Testaments, and Four Bibles. The Four New Testaments and Four Bibles texts

Table 4.3. Spanish to Quechua Translation Results

SPANISH to QUECHUA	Evaluation Set		
Training Set	Bible	NT	Crawled
Baseline	0.39	0.00	3.80
Crawled	0.62	0.22	6.42
Hi-Crawled	0.47	0.00	3.23
NT	2.23	2.98	2.59
NT + Crawled	2.07	2.68	5.20
NT + Hi-Crawled	2.11	2.53	2.90
Bible	2.89	2.80	2.65
Bible + Crawled	3.32	3.27	5.16
Bible + Hi-Crawled	4.53	3.28	4.00
4 NTs	2.46	2.99	2.63
4 NTs + Crawled	2.52	3.12	5.03
4 NTs + Hi-Crawled	2.48	2.97	3.41
4 Bibles	4.70	3.33	2.66
4 Bibles + Crawled	4.55	3.64	5.70
4 Bibles + Hi-Crawled	4.85	3.42	3.23

were used to improve the system by making use of the multiple Spanish translations of the Bible that were available to us. The resulting parallel text was the cross product of the single Quechua or Nahuatl translation and the four Spanish translations. Only the New Testament was available in Nahuatl. The total number of translation models for the Quechua experiment was N*M-1 or 14 (none/none being an invalid combination) in both directions for a total of 28 models. For Nahuatl, only 16 translation models were generated because the Bible and Four Bibles training texts were not available.

Table 4.4. Spanish to Nahuatl Translation Results

SPANISH to NAHUATL	Evaluation Set	
Training Set	NT	Crawled
Baseline	0.00	0.93
Crawled	1.31	2.46
Hi-Crawled	0.00	2.59
NT	10.98	0.80
NT + Crawled	11.04	2.04
NT + Hi-Crawled	10.89	1.94
4 NTs	11.26	0.78
4 NTs + Crawled	10.96	1.71
4 NTs + Hi-Crawled	11.12	1.80

All Bibles and New Testament translations except for the Quechua Bible were downloaded from Bible Gateway.[3]. The Quechua Bible was scanned using OCR for the purposes of this experiment by Christian Loza. The Spanish translations were: Dios Habla Hoy, Nueva Versión International, Biblia de las Américas, and the Reina-Valera 1995 translation. The Nahuatl translation was the 1987 Nahuatl de Guerrero New Testament. All Bibles and New Testaments were converted to a normalized form – not including the Deutero-Canonicals sections and following the numbering in the Latin Vulgate when a disagreement arose between the translations.

For the purposes of evaluation, three books were removed from the Bibles – Exodus, Proverbs, and Hebrews. From the New Testament, only the book of Hebrews was removed. All other books were used for training the Bible-based translation model. For

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

Table 4.5. Quechua to Spanish Translation Results

QUECHUA to SPANISH	Evaluation Set		
Training Set	Bible	NT	Crawled
Baseline	0.38 (0.57)[2]	0.00 (0.00)	3.81
Crawled	0.70 (1.20)	0.44 (0.73)	7.17
Hi-Crawled	0.27 (0.50)	0.00 (0.00)	3.79
NT	3.67 (5.95)	3.73 (6.01)	3.19
NT + Crawled	3.75 (5.93)	3.43 (5.33)	7.02
NT + Hi-Crawled	3.71 (6.01)	4.02 (5.99)	3.03
Bible	4.82 (7.37)	5.80 (8.82)	3.56
Bible + Crawled	4.79 (7.27)	5.68 (8.57)	6.26
Bible + Hi-Crawled	8.15 (12.78)	5.85 (9.05)	4.25
4 NTs	4.18 (7.38)	4.24 (7.79)	3.20
4 NTs + Crawled	4.15 (7.18)	4.28 (7.75)	6.51
4 NTs + Hi-Crawled	4.05 (7.10)	4.39 (7.90)	4.37
4 Bibles	7.99 (15.35)	6.35 (10.66)	3.32
4 Bibles + Crawled	8.02 (15.20)	6.21 (10.67)	6.46
4 Bibles + Hi-Crawled	7.84 (14.95)	5.99 (10.43)	3.76
Upper-Bound	(38.25)	(33.86)	N/A

Parenthesis indicate the score when evaluated across multiple reference translations.

the crawled text evaluation experiment, every tenth line of the crawled data was removed for testing, while the other nine lines were used for training. In this experiment, the default values for Moses were used in all cases.

The baseline used in these experiments is the result of not performing any translation. In other words, if A is the source text and B is the gold standard target, the

Table 4.6. Nahuatl to Spanish Translation Results

NAHUATL to SPANISH	Evaluation Set	
Training Set	NT	Crawled
Baseline	0.00 (0.80)	0.42
Crawled	0.55 (1.36)	0.78
Hi-Crawled	0.44 (0.58)	0.82
NT	10.58 (14.12)	0.24
NT + Crawled	10.07 (13.67)	1.53
NT + Hi-Crawled	10.17 (13.72)	1.04
4 NTs	10.32 (14.93)	0.23
4 NTs + Crawled	9.91 (14.47)	0.87
4 NTs + Hi-Crawled	10.04 (14.65)	1.04
Upper-Bound	(33.86)	N/A

baseline is the score associated with comparing text A to the gold standard - despite its being in a different language. This is an informative baseline, because it indicates the degree of overlap irrelevant of any translation. Numerals, proper nouns, some punctuation, and some cognates may be reflected in this score. The upper bound (when available) is the result of comparing the Dios Habla Hoy human translation of the Bible (or New Testament) with the other available human translations. Since these are all hand translations, they represent a perfect translation and therefore this is the highest score that can be expected. Note that this upper bound is only available when translating into Spanish and that the value is not 100%.

4.2. Results

The scores associated with each translation may be found in Tables 4.3 through 4.6. From these scores a few observations can be made. It was predicted that the best translation model would be the model that best balances quality and quantity. The Four Bibles plus Hi-Crawled data appears to meet this criterion, but based upon the results, this model performed best on only one evaluation set: converting a sample from the Bible from Spanish to Quechua. In the experiment converting the New Testament from Spanish to Quechua, the Four Bibles plus Crawled model performed better. In many experiments, a higher score was achieved on the Biblical evaluation sets by ignoring the crawled data completely. However, the opposite was true when evaluating on the crawled data. In the experiment converting the crawled evaluation set from Quechua to Spanish, the score with low-quality crawled data alone was 7.17. When the New Testament training data was added it degraded to 7.02 and with the Four New Testaments data, it degraded further to 6.51. With the Bible training data, it degraded to 6.26 and improved somewhat to 6.46 by adding the Four Bibles data. Some translation models appeared to do worse than the baseline according to the BLEU metric. In the experiment translating the crawled evaluation set from Quechua to Spanish, the baseline was 3.81 while the score for the model trained on Four New Testaments was only 3.20. In short, no clear trend is visible from this data.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1. Practical Concerns

During the process of discovering Web-crawled Quechua-Spanish and Nahuatl-Spanish parallel texts, I encountered several issues that should be addressed in any future work. First of all, the Web is a messy place. There is no better way to describe it. There are spelling errors, sentence fragments, punctuation errors, and formatting issues. Each page is created using some character encoding, and it is not always obvious which one is being used. During the Quechua-Spanish experiment, I did not anticipate a problem with character encodings and assumed that gathering pages in UTF-8 would be sufficient. Frequently encoding and decoding into and out of UTF-8 resulted in some malformed data. At the end of the process, I made an effort to re-crawl the pages that had passed through the filters, convert them to UTF-8 uniformly, and re-align them. In the Nahuatl experiment, all files were downloaded locally and converted into UTF-8 immediately in order to avoid this issue.

Second, supplementing an existing system with parallel texts gathered automatically from the Web may be problematic. There is no guarantee that the texts are high-quality, and so it is possible that the texts may degrade rather than improve system performance. Texts found on the Web that are meant to be valid translations may be suspect. The nature of the Web is such that a person of any language proficiency level can post content, so it may be that the quality of some translations is poor.

5.2. Conclusions

5.2.1. The Importance of Domain

While the effects of translating Biblical texts using crawled training data were less than impressive, it is nonetheless apparent from these results that crawled texts are crucial when translating texts in a non-Biblical domain. By evaluating the translation models against a crawled testing set, I was able to observe that supplementing the models with crawled training always improved the results over Biblical training data alone. This is most clear in the Quechua->Spanish experiment where adding crawled training data to the New Testament texts raised the score from 3.19 to 7.02. It may be beneficial to test this assertion on a better-trusted evaluation set.

5.2.2. Multiple Bibles/New Testaments Improve Results Significantly

In all cases, the addition of multiple translations in the training sets improved results over using a single translation. While the Law of Diminishing Returns applies, the number of translation pairs available increases significantly as new translations are added. For instance, the addition of another Quechua or Nahuatl translation to the current system (1 minor- x 4 major-language translations) would result in a total of 8 pairwise translations expressing the same idea in multiple forms.

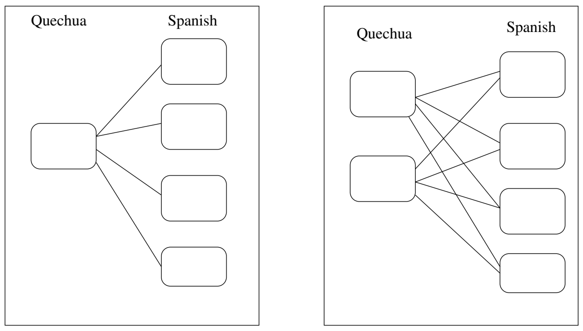

Figure 5.1. A Single Additional Quechua Translation Results in Four Additional Language Pairs

5.2.3. Drawbacks of the BLEU Metric Used by Moses

It may be disadvantageous for the BLEU algorithm to be applied in such a way that if no K-grams are found the overall score is zero even if there are (K-1)-grams. In Appendix B, I have proposed a modification to the BLEU n-gram combination which takes into account the possibility of no matching k-grams. I have also evaluated all of the output translations with the modified BLEU algorithm and have compared the results of the two metrics.

5.2.4. Translating Nahuatl Versus Translating Quechua

Based upon the results of the Nahuatl experiment (specifically the NT evaluation), it appears that the quality of the translations are better than they are using the same training data and evaluation data for the Quechua experiment. For instance, the Four New Testaments translation model translated the Spanish New Testament data into Quechua with a score of 2.98. The corresponding Nahuatl translation model produced a translation score of 11.26. Other translation models behaved comparably. It is not clear why this is the case. The New Testament evaluation set was specifically applied to the Spanish-Quechua translation models to determine whether the reduced size of the evaluation set led to better results, but it appears that this is not the case.

5.2.5. Highly-Aligned Crawled Texts Do Not Correlate to Better Quality Translations

In general, the Hi-Crawled training sets either performed roughly the same as the Crawled sets or performed significantly worse. Based on this observation, I can theorize that high-recall parallel texts may be more important than high-precision texts in the development of low-density language machine translation systems. In other words, having a lot of bad translations may be more helpful in building a translation model than having one or two good translations. Again, further analysis is recommended.

5.2.6. The Quality of the Nahuatl- and Quechua-Language Texts

When the crawled evaluation set is analyzed comparatively for the two languages, it is apparent that the Nahuatl-Spanish parallel text yields significantly less improvement than the Quechua-Spanish text. Anecdotally, the Nahuatl text appears to be of noticeably lower quality than that of the Quechua. The aligned phrases are shorter on average, many lines in the Nahuatl text consist only of proper nouns, and the size ratio of the two sets (in KB) indicates a discrepancy (cf. Figure 4): the Nahuatl to Spanish ratio is 190KB/480KB while the Quechua to Spanish ratio is 550KB/540KB. For both language pairs, the Spanish translations of the New Testament are shorter, so for us to observe such a large size differential between the crawled Nahuatl and Spanish texts suggests that the results of the Nahuatl experiment may have been tainted.

5.3. Future Work

Candidate generation is the stage of the project most in need of improvement. It may be beneficial to supplement the language specific Web crawl with candidate generation methods such as top level domain crawling [12] and utilizing a search engine to find language-specific anchor tags as in STRAND [17]. These methods could be used in conjunction with the language specific Web crawl as well as with a host crawl as done in BITS [12] and PTMiner [2] to improve the quantity of candidate pairs. It may also be beneficial to utilize archival data as proposed by Resnik [19].

The URL matching module is arguably the most-effective and widely used parallel text filter described by previous systems, but was given little attention in Babylon due to its inability to find on-page translations and its inflexibility when dealing with languages with uncertain naming conventions. Future work, should attempt to alleviate the need for *a priori* language knowledge in order to increase the utility of URL pattern matching components.

Throughout these experiments, preset values were used to determine language content, filter thresholds, alignment scores, and translation settings. They were all set before the experiment began based upon my intuition, and were not changed at any time in the experiment. Future work should explore the effectiveness of these threshold values in a formal attempt to maximize discovered text quality through experimentation and machine learning.

Finally, resources other than Web data and Biblical text should be used to improve translation models. Other sources of this data include governmental texts, literature, language community involvement, and parallel texts created by other researchers in the field such as that produced by Monson et al. for the Quechua language[14].

APPENDIX A

TOOLS

A.1. Text_cat

text_cat[1] [1] is an N-Gram based text categorization system that can be used to determine the language of a given text. Out of the box, the implementation provided around 80 language models (including Quechua and Spanish, but not Nahuatl). The script included a routine to build language models for additional languages using a short sample of monolingual text. The Nahuatl language model was built using a subset of the monolingual Web crawl provided to us by Kevin Scannell (personal correspondence).

text_cat compares the character-based n-gram counts of an input text to the language models in its repertoire and indicates which language model has the minimum distance from the input text. The language model is simply an ordering of the 300 most common n-grams in the language for n-grams up to and including trigrams. The distance between a text and a language model is the out-of-place measure - the distance of an n-gram ranking in the text from its ranking in the language model. n-grams not in the language model are given a score of some maximum distance. See Figure A.1 for clarification.

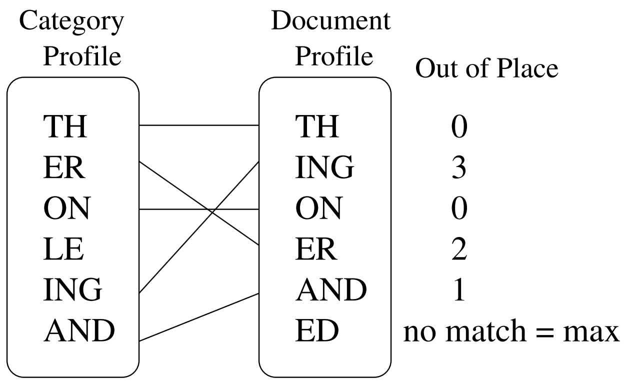

Figure A.2. Sample N-gram Frequency Statistics

[1] http://www.let.rug.nl/~vannoord/TextCat/

The actual result of text_cat is either a listing of the most likely language models in decreasing order of confidence down to some threshold (default 1.05 * best_score). If more than X (default 10) language models match the sample text within that threshold, text_cat returns the string "I don't know".

There is an option in the text_cat script to allow a text to be read line by line rather than as a single entity. This seemed well suited to the task of finding small snippets of minor-language data on a page primarily in another language. However, the script as provided reloaded all of the language models each time a line was categorized, and so it was too inefficient to be useful as a component in a larger system. I modified the script to load the language models only once per file rather than once per line. The modified version of text_cat is available for download.[2]

A.2. Champollion

Champollion[3] [11] is a sentence alignment system that begins with the assumption that the input texts are noisy (i.e. contain a large percentage of alignments that are not 1-1) and that a large portion of the text is either additional or missing. This differs from most existing alignment tools which have been optimized for clean data sets or for a specific language pair. As data gathered from the Web is often noisy, Champollion appears to meet my needs.

According to the literature, Champollion assumes that an alignment does not exist unless lexical evidence indicates otherwise. The tool accepts a translation lexicon to determine if there are parallel lexical features in the texts that support an alignment. However, in the absence of a lexicon, Champollion makes use of non-translated tokens such as punctuation, abbreviations, numbers, etc. to find lexical parallelism. Length-based metrics are used to supplement lexical information and to weed out

[2]http://lit.csci.unt.edu/~babylon/tools/text_cat_new

[3]http://champollion.sourceforge.net

poor alignments. In an experiment with a noisy manually-aligned English-Chinese corpus, Champollion earned a precision of recall of 97.0% and 96.9% respectively with a translation lexicon of 58,000 words and precision and recall of 88.1% and 90.8% respectively with no lexicon at all[11]. This indicates the utility of Champollion in dealing with language pairs for which no lexicon exists at all.

Champollion also differs from other alignment systems by treating various translated terms differently. A term's tf-idf (term frequency-inverse document frequency) is a measure commonly used in information retrieval tasks to weight a term's importance in document by preferring terms that are in the document many times (i.e. they have a high term frequency) but are not in many documents in the collection (i.e. they have a low document frequency). In English texts, this measure disadvantages common terms like *the*, *and*, and *he* which are very common in most contexts. Champollion builds upon the tf-idf score by treating segments of a text as individual documents. The similarity function for Champollion is based upon the stf-idtf, the ratio of segment term frequency (stf) - the importance of a term within a given segment - times the inverse document term frequency (idtf) - the inverse frequency of the term in all segments of the document. An alignment penalty (preferring 1-1 matchings) and a length penalty (preferring matchings of similar length) are also used to compute the alignment. The final alignment is performed using a dynamic programming algorithm similar to that used by Gale and Church [6].

A.3. Moses

Moses [9] is an open-source collection of machine translation tools designed as a "drop-in" replacement for the closed-source, proprietary toolkit Pharaoh whose conception is described in an earlier paper by Koehn [10]. It includes tools to build language models, tools to build a translation model, tools to perform a translation using the language and translations models, and tools to evaluate the quality of a translation.

A language model is a representation of the word-level n-gram frequencies found in a representative monolingual sample. The language model is used in the text generation step of the translation so that, given an effective bag of words produced by translating into the target language, the word order can be made to match that of similar sentences as defined in the language model.

Moses builds a translation model by taking as input a pair of sentence-aligned parallel texts. By training on this pair of texts, Moses is able to build an effective probabilistic phrase-based translation table in both directions. The phrase-based table is constructed [10] by building upon word alignments generated bidirectionally and taking the intersection of the two alignments. Phrase alignment points are added iteratively. For each word *w* in the phrase alignment table, all words aligned with *w* in the union of the bidirectional word alignments are considered and are added if they connect some previously unaligned word. In addition to word-based phrase generation, Moses allows for multiple factors in its training such as part of speech, tense, etc. which can then be combined with the words to improve the quality of the translation.

Once the translation model is built, Moses is able to translate an arbitrary source language text into the target language using a Bayesian decoding algorithm. For an input sentence *s*, the system attempts to translate it into some sentence *t* with the maximum probability given *s* or, in other words, the maximal p(t|s). The Bayesian relationship is shown in the following equation:

$$argmax_e p(t|s) = argmax_e p(s|t)p(t)$$

This equation then depends upon a language model to find p(t). The input sentence in language *s* is segmented into N phrases and a uniform probability distribution is assumed for all possible segmentations. The probability p(s|t) for a given segmentation is found by finding the translations of the phrases in s into phrases in the target language and finding the overall probability for that segmentation of *s*. The maximal segmentation probability is used along with the language model to produce a final sentence *t* of maximal likelihood.

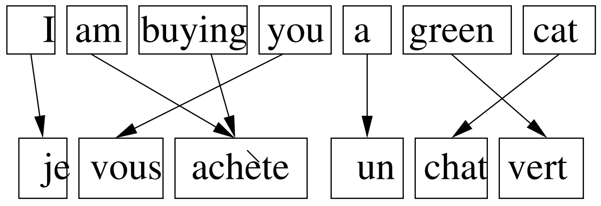

Figure A.3. Non-Factored Translation Using Moses

Once an output translation is created, it can be quantitatively compared to other translations using Moses' bundled implementation of BLEU described below. More information on the Moses toolkit can be found on the Moses Web site[4] as well as in Koehn's literature [9] [10].

A.4. BLEU

BLEU [16] is a tool for evaluating the quality of a machine translation. It begins with the assumption that "the closer a machine translation is to a professional human translation, the better it is." Therefore, it requires both a human translation of the text being translated and a measure of how similar the machine translation is to the gold standard human translation. Since the usual reason for producing a machine translation is because a human translation is unavailable would be too expensive or time-consuming to create, BLEU's usefulness in determining the quality of a machine translation is limited to tuning a machine translation system in which a closed environment can be assured. Out in the wild, it is assumed that a machine translation system

[4]http://www.statmt.org/moses/

tuned using the BLEU metric will perform better than the same system without the training.

The translation comparison metric proposed for BLEU is a modified n-gram precision for a machine translation and one or more reference translations. The base n-gram precision of two translations is the number of n-grams of the machine translation that also appear in the reference translation divided by the total number of n-grams in the machine translation. This is modified by observing that an n-gram should appear no more frequently in a machine translation than in a human translation. An n-gram that appears multiple times in the machine translation only counts as a match (for the purpose of finding precision) the maximum number of times that the n-gram appears in any given reference translation. For example, consider the following candidate and reference solutions:

Candidate: the the the the the the the

Reference 1: The cat is on the mat.

Reference 2: There is a cat on the mat.

The standard unigram precision would be 7/7 since all seven *the* tokens appear in one or more reference solution. However, the modified unigram precision is only 2/7 since the token *the* appears at most twice in any reference translation.

While Papineni [16] reports that BLEU corresponds well to human evaluation by measuring any single n-gram alone, a combination of n-grams was chosen in order to provide a more fine-grained score. The n-gram scores are combined using a geometric sum as described in the following pseudocode: ‘

```
brevity_penalty ← 1
if translation_length < reference_length then
    brevity_penalty ← $e^{(1-reference\_length/translation\_length)}$
```

```
end if
answer ← brevity_penalty*$e^{(log(B_1)+log(B_2)+log(B_3)+log(B_4))/4}$
```

where the length variables are measured in number of tokens, reference_length in a multi-reference context is generated by adding at each step the length closest to the translation length, and B_k is the precision for all k-grams. The implementation provided uses n-grams up to n=4. One side effect of this implementation is that any translation in which the 4-gram precision is zero will have a total score of zero because log(0) is negative infinity. Even if there are substantial unigrams, bigrams, and trigrams, a lack of 4-grams will nullify the entire score – which occurred in several of the experiments. An alternative combination for the n-gram scores as well as the results of a rerun of the new algorithm on the output translations may be found in Appendix B.

APPENDIX B

RE-EVALUATION USING ENHANCED BLEU METRIC

The tables below show the results of applying the modified BLEU algorithm to the output translations already measured in Section 4.2. The algorithm is similar to the one bundled with Moses (and proposed in the BLEU literature[16]). The Moses implementation produces a score which I will call the (1 to 4)-gram score based on the psuedocode from Appendix A.4 and abridged below for convenience:

answer $\leftarrow$ 100*e$^{(log(B_1)+log(B_2)+log(B_3)+log(B_4))/4}$

Here B_x is the ratio of machine translated x-grams that were found in the reference translations to those that were not found. The output produces four scores in the same manner – one for unigrams, one for bigrams, one for trigrams, and one for 4-grams – and combines them as shown below for some weighting value A:

$S_1 \leftarrow 100 * B_1$

$S_2 \leftarrow 100 * e^{(log(B_1)+log(B_2))/2}$

$S_3 \leftarrow 100 * e^{(log(B_1)+log(B_2)+log(B_3))/3}$

$S_4 \leftarrow 100 * e^{(log(B_1)+log(B_2)+log(B_3)+log(B_4))/4}$

$answer \leftarrow A^8 * S_1 + A^4 * S_2 + A^2 * S_3 + A * S_4$

In general, where S_x is the score defined above as a (1 to x)-gram score, the resulting score for my algorithm can be defined for arbitrary N-grams as:

$$\sum_{i=1}^{N} 100 * A^{2^{N-i}} S_i \ where \ \sum_{i=0}^{N-1} A^{2^i} = 1$$

For N=4, this implies that A=0.56616081. This backoff ensures that translations without 4-grams will not automatically be given a score of zero as is the case in the original Moses implementation. Rather, the score will consider trigrams, bigrams, and unigrams. Additionally, this implementation maintains a preference for higher-order N-grams through an exponential weighting system. That is, bigrams are twice as important as unigrams, trigrams are twice as important as bigrams, and 4-grams are twice as important as trigrams. Papineni [16] stated that one of the goals of BLEU was

Table B.1. N-gram Overlap Example

N-gram	N	Count for S1	Count for S2
de	1	1	1
hermanos	1	1	1
los	1	1	1
sus	1	1	1
todos	1	1	1
y	1	1	2
sus hermanos	2	1	1
todos los	2	1	1

S1) josepas sus hermanos a egipto y de todos los que por masinkunapas wañukapurqankun .

S2) josé y sus hermanos , y todos los de esa generación , murieron ;

to provide fine-grained feedback for machine translation systems, and it is in keeping with this goal that I propose this modified algorithm for use in machine translation systems. When working with languages with very few resources, it is vital to have a fine grained translation quality metric that distinguishes between a true 0.0% precision and a quality translation that merely lacks 4-grams.

Consider the sentences associated with Table B.1 above (Exodus 1:6). Sentence 1 is taken from my system's translation from Quechua into Spanish using the Bible + Hi-Crawled translation model. Sentence 2 is the reference Dios Habla Hoy human translation. Both sentences have a total of 14 unigrams, 13 bigrams, 12 trigrams, and 11 4-grams. In the same table, observe which n-grams are common to both sentences. Based on these n-gram counts, the unigram score (B_1) for the machine translation is 6/14 or 42.9%, the bigram score (B_2) is 2/13 or 15.4%, and both the trigram and 4-gram score is 0% since no trigrams or 4-grams were common to both translations.

Below are final scores for the old and new algorithms as described above. Observe that since log(0) is $-\infty$ and for all x, $x + -\infty = -\infty$, the result of $e^{-\infty}$=0 when any n-gram score is not represented.

$A \leftarrow 0.56616081$

$S_1 \leftarrow 100 * e^{log(.429)} = 42.90$

$S_2 \leftarrow 100 * e^{(log(.429)+log(.154))/2} = 6.61$

$S_3 \leftarrow 100 * e^{(log(.429)+log(.154)+log(0))/3} = 0.00$

$S_4 \leftarrow 100 * e^{(log(.429)+log(.154)+log(0)+log(0))/4} = 0.00$

$score_{new} \leftarrow A^8 * (42.90) + A^4 * (6.61) + A^2 * (0) + A * (0) = 1.13$

$score_{old} \leftarrow 100 * e^{(log(.429)+log(.154)+log(0)+log(0))/4} = 0.00$

Keep in mind that the raw numerical values themselves (for the new algorithm and for the original algorithm) are not useful out of context. They only reflect relative quality within an implementation, so the numbers reported in the tables below are not directly comparable to the numbers in Section 4.2 except insomuch as they reflect a relative ranking among the translations. My modified implementation of BLEU is available for download.[1] I have attempted to show a correlation between the two metrics in Figure B.6. Figure B.6 compares the two metrics for a Quechua to Spanish translation of the Bible evaluation set. Other experiments showed similar results.

[1]http://lit.csci.unt.edu/~babylon/tools/multi-bleu-new.perl

Table B.2. Spanish to Quechua Experiment

Spanish to Quechua Training Set	Evaluation Set		
	Bible	NT	Crawled
Baseline	0.84	0.45	4.80
Crawled	1.30	0.70	7.70
Hi-Crawled	1.03	0.57	4.28
NT	3.88	4.97	3.50
NT + Crawled	3.64	4.75	6.40
NT + Hi-Crawled	3.71	4.54	4.11
Bible	4.73	5.04	3.57
Bible + Crawled	5.17	5.46	6.41
Bible + Hi-Crawled	6.66	5.57	5.26
4 NT	4.19	5.15	3.54
4 NT + Crawled	4.26	5.31	6.32
4 NT + Hi-Crawled	4.28	5.18	4.58
4 Bibles	6.98	5.72	3.58
4 Bibles + Crawled	6.81	6.08	6.91
4 Bibles + Hi-Crawled	7.08	5.87	4.45

Table B.3. Spanish to Nahuatl Experiment

Spanish to Nahuatl	Evaluation Set	
Training Set	NT	Crawled
Baseline	0.88	1.04
Crawled	2.75	2.72
Hi-Crawled	1.64	2.89
NT	15.29	0.91
NT + Crawled	15.38	2.29
NT + Hi-Crawled	15.16	2.23
4 NT	15.56	0.88
4 NT + Crawled	15.31	1.94
4 NT + Hi-Crawled	15.42	2.06

Table B.4. Quechua to Spanish Experiment

Quechua to Spanish Training Set	Evaluation Set		
	Bible	NT	Crawled
Baseline	0.80 (1.17)[2]	0.44 (0.61)	4.80
Crawled	1.54 (2.55)	1.46 (2.28)	8.68
Hi-Crawled	0.69 (1.25)	0.37 (0.86)	5.21
NT	5.69 (9.07)	6.24 (9.67)	4.26
NT + Crawled	5.80 (9.12)	5.91 (9.07)	8.53
NT + Hi-Crawled	5.77 (9.18)	6.58 (9.75)	4.34
Bible	7.02 (10.77)	8.64 (13.03)	4.75
Bible + Crawled	7.01 (10.70)	8.46 (12.64)	7.85
Bible + Hi-Crawled	10.96 (16.93)	8.65 (13.28)	5.65
4 NT	6.33 (10.80)	6.79 (11.71)	4.32
4 NT + Crawled	6.30 (10.61)	6.87 (11.76)	8.07
4 NT + Hi-Crawled	6.22 (10.53)	6.96 (11.86)	5.80
4 Bibles	10.91 (19.74)	9.22 (15.06)	4.47
4 Bibles + Crawled	10.95 (19.62)	9.14 (15.14)	7.87
4 Bibles + Hi-Crawled	10.73 (19.34)	8.84 (14.80)	5.07
Upper Bound	(43.27)	(38.55)	N/A

Parenthesis indicate the score when evaluated across multiple reference translations.

Table B.5. Nahuatl to Spanish Experiment

Nahuatl to Spanish	Evaluation Set	
Training Set	NT	Crawled
Baseline	0.88 (1.67)	0.47
Crawled	1.80 (2.92)	0.83
Hi-Crawled	1.19 (1.69)	0.87
NT	14.27 (18.76)	0.26
NT + Crawled	13.79 (18.41)	1.61
NT + Hi-Crawled	13.88 (18.44)	1.12
4 NT	14.00 (19.73)	0.25
4 NT + Crawled	13.62 (19.24)	0.93
4 NT + Hi-Crawled	13.69 (19.42)	1.11
Upper Bound	(38.55)	N/A

Table B.6. Comparison of Old and New BLEU Metrics for Quechua to Spanish Translation of the Bible Evaluation Set[3]

Training Set	New BLEU algorithm	Moses-bundled algorithm
Baseline	0.80 / 1.85%	0.38 / 0.99%
Crawled	1.54 / 3.56%	0.70 / 1.83%
Hi-Crawled	0.69 / 1.59%	0.27 / 0.71%
NT	5.69 / 13.15%	3.67 / 9.59%
NT + Crawled	5.80 / 13.40%	3.75 / 9.80%
NT + Hi-Crawled	5.77 / 13.33%	3.71 / 9.70%
Bible	7.02 / 16.22%	4.82 / 12.60%
Bible + Crawled	7.01 / 16.20%	4.79 / 12.52%
Bible + Hi-Crawled	10.96 / 25.33%	8.15 / 21.31%
4 NT	6.33 / 14.63%	4.18 / 10.93%
4 NT + Crawled	6.30 / 14.56%	4.15 / 10.85%
4 NT + Hi-Crawled	6.22 / 14.37%	4.05 / 10.59%
4 Bibles	10.91 / 25.21%	7.99 / 20.89%
4 Bibles + Crawled	10.95 / 25.31%	8.02 / 20.97%
4 Bibles + Hi-Crawled	10.73 / 24.80%	7.84 / 20.50%
Upper Bound	43.27 / 100.00%	38.25 / 100.00%

Output translations are compared with the Dios Habla Hoy Spanish translation only. The percentage is the ratio of the raw score to the upper bound score.

BIBLIOGRAPHY

[1] William B. Cavnar and John M. Trenkle, *N-gram-based Text Categorization*, Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval (Las Vegas, US), 1994, pp. 161–175.

[2] Jiang Chen and Jian-Yun Nie, *Parallel Web Text Mining for Cross-Language IR*, Proceedings of RIAO-2000: Content-Based Multimedia Information Access, 2000.

[3] Jisong Chen, Rowena Chau, and Chung-Hsing Yeh, *Discovering Parallel Text from the World Wide Web*, ACSW Frontiers '04: Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalization (Darlinghurst, Australia, Australia), Australian Computer Society, Inc., 2004, pp. 157–161.

[4] Yun-Chuang Chiao, Olivier Kraif, Dominique Laurent, Thi Minh Huyen Nguyen, Nasredine Semmar, François Stuck, Jean Véronis, and Wajdi Zaghouani, *Evaluation of Multilingual Text Alignment systems: the ARCADE II Project*, 2006 International Conference on Language Resources and Evaluation, 2006.

[5] Daniel Cunliffe and Susan C. Herring, *Introduction to Minority Languages, Multimedia and the Web*, New Review of Hypermedia and Multimedia 11 (2005), no. 2, 131–137.

[6] William A. Gale and Kenneth W. Church, *A Program for Aligning Sentences in Bilingual Corpora*, Compututational Linguistics 19 (1993), no. 1, 75–102.

[7] David Geer, *Statistical Machine Translation Gains Respect*, Computer 38 (2005), no. 10, 18–21.

[8] Rayid Ghani, Rosie Jones, and Dunja Mladenić, *Mining the Web to Create Minority Language Corpora*, CIKM '01: Proceedings of the tenth international conference on Information and knowledge management (New York, NY, USA), ACM Press, 2001, pp. 279–286.

[9] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst, *Moses: Open Source Toolkit for Statistical Machine Translation*, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions (Prague, Czech Republic), Association for Computational Linguistics, June 2007, pp. 177–180.

[10] Philipp Koehn, Franz Josef Och, and Daniel Marcu, *Statistical Phrase-Based Translation*, NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (Morristown, NJ, USA), Association for Computational Linguistics, 2003, pp. 48–54.

[11] Xiaoyi Ma, *Champollion: A Robust Parallel Text Sentence Aligner*, 2006.

[12] Xiaoyi Ma and Mark Y. Liberman, *BITS: A Method for Bilingual Text Search over the Web*, 1999.

[13] Mike Maxwell and Baden Hughes, *Frontiers in Linguistic Annotation for Lower-Density Languages*, Proceedings of the Workshop on Frontiers in Linguistically Annotated Corpora 2006, Association for Computational Linguistics, 2006, pp. 29–37.

[14] Christian Monson, Ariadna Font Llitjós, Roberto Aranovich, Lori Levin, Ralf Brown, Eric Peterson, Jaime Carbonell, and Alon Lavie, *Building NLP systems for Two Resource-Scarce Indigenous Languages: Mapudungun and Quechua*, LREC

2006: Fifth International Conference on Language Resources and Evaluation, 2006.

[15] Jian-Yun Nie, Michel Simard, Pierre Isabelle, and Richard Durand, *Cross-Language Information Retrieval Based on Parallel Texts and Automatic Mining of Parallel Texts from the Web*, SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval (New York, NY, USA), ACM Press, 1999, pp. 74–81.

[16] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu, *BLEU: A Method for Automatic Evaluation of Machine Translation*, ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (Morristown, NJ, USA), Association for Computational Linguistics, 2001, pp. 311–318.

[17] Philip Resnik, *Parallel Strands: A Preliminary Investigation into Mining the Web for Bilingual Text*, AMTA '98: Proceedings of the Third Conference of the Association for Machine Translation in the Americas on Machine Translation and the Information Soup (London, UK), Springer-Verlag, 1998, pp. 72–82.

[18] ______, *Mining the Web for Bilingual Text*, Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics (Morristown, NJ, USA), Association for Computational Linguistics, 1999, pp. 527–534.

[19] Philip Resnik and Noah A. Smith, *The Web as a Parallel Corpus*, Computational Linguistics 29 (2003), no. 3, 349–380.

[20] Lei Shi, Cheng Niu, Ming Zhou, and Jianfeng Gao, *A DOM Tree Alignment Model for Mining Parallel Data from the Web*, ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL (Morristown, NJ, USA), Association for Computational Linguistics, 2006, pp. 489–496.

[21] Anil Kumar Singh and Samar Husain, *Comparison, Selection, and Use of Sentence Alignment Algorithms for New Language Pairs*, 2005, pp. 99–106.

[22] Kulwadee Somboonviwat, Masaru Kitsuregawa, and Takayuki Tamura, *Simulation Study of Language Specific Web Crawling*, ICDEW '05: Proceedings of the 21st International Conference on Data Engineering Workshops (Washington, DC, USA), IEEE Computer Society, 2005, p. 1254.

[23] Kulwadee Somboonviwat, Takayuki Tamura, and Masaru Kitsuregawa, *Finding Thai Web Pages in Foreign Web Spaces*, ICDEW '06: Proceedings of the 22nd International Conference on Data Engineering Workshops (ICDEW'06) (Washington, DC, USA), IEEE Computer Society, 2006, p. 135.

[24] J. Tomás, E. Sánchez-Villamil, L. Lloret, and F. Casacuberta, *WebMining: An Unsupervised Parallel Corpora Web Retrieval System*, Proceedings from the Corpus Linguistics Conference (Birmingham, U.K.), July 14th -17th 2005.

[25] Daniel Varga, Peter Halacsy, Andras Kornai, Viktor Nagy, Laszlo Nemeth, and Viktor Tron, *Parallel Corpora for Medium Density Languages*, 2005.