THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

DEPARTMENT OF COMPUTER SCIENCE

Examining the Efficiency of Rule-Based Machine Translation

KYLE BURNS
Spring 2019

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Computer Science
with honors in Computer Science

Reviewed and approved* by the following:

Richard Zhao, Ph.D.
Assistant Professor of Computer Science and Software Engineering
Thesis Supervisor

Meng Su, Ph.D.
Chair, Computer Science and Software Engineering, Associate Professor of Computer Science
and Software Engineering
Honors Adviser

* Signatures are on file in the Schreyer Honors College.

ABSTRACT

Regardless of the advancements in technology, humans will always need language to communicate with one another. There are over 7,000 recognized languages around the world, and while not all of them are widely used, this still presents trouble for communicating across language barriers.[1] Machine Translation has existed for decades, and the field has split into different schools of translation, mainly statistical machine translation and rules-based machine translation. Statistical machine translation, a method based heavily on probabilities adapted from pre-compiled language texts, is a faster approach but is more prone to errors, while rule-based translations, which require native speakers to notate all the rules of a language in a computer digestible format, are traditionally more accurate, with the tradeoff of being slower. The purpose of this experiment is to examine if a rules-based machine translation system can be improved enough to match a statistical based translation system in the amount of time needed for translations, while still retaining the accuracy of a rules-based translation system. This experiment, using various modified versions of the Apertium English to Spanish translation system which were ran on a testing corpus of 200 sentences, shows that a rules-based system can be moderately improved in time, but not to the extent of matching or overtaking a statistical system.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to acknowledge the unwavering support and helpfulness of my two project mentors Dr. Zhao and Dr. Su. They both have been immensely helpful throughout the thesis project, as well as throughout my college career. I would also like to acknowledge all of the Computer Science professors I have had during my time at Penn State, for passing along their knowledge and expertise throughout the courses they have taught me.

Chapter 1

Introduction

1.1: History of Machine Translation

Machine translation as a field of research has been around since the 1950s in the area of computational linguistics, with research starting at many universities such as Georgetown in 1954. In an experiment using an IBM computer, sentences were translated from Russian to English, in a study sponsored by the CIA[2]. These systems were very structured and regimented for the original experiment, and the sentences selected for the experiment were carefully crafted and all were commonly related. In the 1950s, the United States asked a scientist named Yehoshua Bar-Hillel to investigate the merit of machine translation moving forward, and Bar-Hillel stated that this technology could be infeasible to use, due to the natural ambiguity in languages[3]. In the 1960s, computer translations were rough in translation, and more significant documents had to be reviewed by hand to fix the translations. At the start of the 1980s, mainframe computers were widely used such as Systran for translations[5]. Later in the 1980s, example-based machine translation or translation by analogy was developed, where a machine was given a list of sentences and used that to learn translations. The translation style worked by breaking down input sentences into smaller practical pieces, performing the translation, and then building the sentence in the target language[4]. The FALCon system, developed by the Air Force in the 1990s, required no prior knowledge of languages for use, and was used in Bosnia and the Pacific Rim in military applications[6]. In the modern day, applications such as Google Translate have taken center stage in language translation due to the availability of the software, as well as the speed of translation due to the statistical nature in Google's translation system. Open source systems are also in use in programs such as *Moses* and *Apertium*, and these systems are commonly used for research purposes.

1.2: Statistical Based Machine Translation

The style of statistical machine translation is rooted mainly in the mathematical field of probability. Statistical translation requires a large amount of data in the form of training sentences to create translation probabilities. A corpus, or a collection of prepared sentences in two separate languages, is required to accurately train a translation system, and a corpus can range from a few thousand sentences to over a million sentences. While training, the machine will read in the corpus of sentences, and determine the potential translations based upon the pairs of sentences in the source and target languages. In the case where words can be identified to two separate translations, this is where the probability part of the process becomes important. While sifting through the training sentences, the system will identify potential matches based on their frequencies throughout the text and resolve a probabilistic value to each possible match. For example, an expression such as "to be" in English can be matched to two separate Spanish verbs, "ser" and "estar". Ser is used for a permanent quality, whereas estar is used for a temporary state. The weights of the translation of "to be" will then be defined by the frequencies of ser and estar in the Spanish side of the prepared bilingual corpus.

The set of probabilities can then be compiled into a language model, which can then be evaluated for accuracy using different language training model evaluators. A common example of a language model evaluator is KenLM, which estimates language models using modified Kneser-Ney smoothing. This method of smoothing evaluates how likely a word is to appear relative to the words that either precede or follow the base word. An example of this would be evaluating how often York appears surrounded by New and City in the phrase New York City. This smoothing technique is viewed to be strong based on the ability it has, to round out low probability matches such as only translating York as opposed to the phrase New York City. This language model evaluation method is the default for Moses, one of the main open-source Statistical Machine Translation systems available, and a heavily used system in research in the field of machine translation[19].

A strength of this form of translation is the speed of translating text from the source to the target language. A statistical translation model being based largely in probability equates to a fast processing time, as the system only has to find the word to translate and evaluate which of the weighted possibilities to substitute in place. This also leads to an issue with statistical machine translation however. If a word has two matches, one with a 95% probability of being the accurate translation, and one with a 5% probability of being correct, there is a chance that the translation of 5% can be selected, despite being the incorrect selection. Another case where an issue like this can reside is if three different translations all carry weights between 30 and 40 percent, leading to a high probability of the incorrect translation being selected.

1.3: Rule-Based Machine Translation

An alternative style of translation, which is also widely adopted across translation systems, is the utilization of a rule-based system. While a system like this does not require a corpus of prepared bilingual data, it does require someone who speaks the language, usually a linguist, to manually craft a set of rules that the translation system will follow. These systems rely on some form of lexical analysis that occurs throughout sentences of a language, which can improve the overall accuracy of the translation. These systems can rely upon identifying and tagging sentences by their parts of speech to assist with the translations.

Rule-based translation systems can excel at adapting between languages with different orders or types. There are 6 major styles of language, where S stands for subject, O stands for object, and V stands for verb: SOV, SVO, VSO, VOS, OVS and OSV[7]. Out of the 402 languages examined by Russell Tomlin of Cambridge, 45% of them were SOV, including languages such as Japanese, and 42% of them were SVO, such as English or French[8]. When changing between the six language forms, despite some being more prevalent than others, if a rule set has been crafted for the translation, such as Japanese to English, a rule-based translation can handle the word reordering by applying translation rules. This can be an advantage in translation accuracy over statistical translation which can face difficulties reordering words.

An advantage of using this translation style as previously mentioned is the refined accuracy that comes with this translation, as well as the overall adaptability of the rule set. If a rule is being overlooked, or a rule has been left out, one can be easily added to continue to refine the translations. A disadvantage that stems from this style of translation system, however, is the requirement of pattern matching of some sort to identify which rules need to be applied. Another disadvantage from this method is the need for a bilingual speaker of the source and target language, with an extensive enough knowledge of the rules of both languages to craft the rule sets. An open-source software solution for this type of translation is Apertium, which was the primary system using for this experiment.

Chapter 2

Related Work

For the background research in this project, I wanted to examine different methods of optimizing the overall translation process. The improvement of the overall process can be achieved in many ways, from the accuracy of translation, to the speed of translation, to the algorithms used through the process.

A way of improving accuracy is by improving the identification of multiword expressions. A multiword expression is a phrase which cannot be broken down further to establish the meaning of the phrase. In the paper Integrating Specialized Bilingual Lexicons of Multiword Expressions for Domain Adaptation in Statistical Machine Translation[12], the authors explore the idea of using lexicons built with multiword expressions can improve accuracy of corpora data by adapting the translation data. The experimentation method identified likely expressions and used them in the translation process by testing the original lexicon versus the new lexicon of multiword expressions, to test for better results. When using pattern filtering to improve possible matches, the translation quality was improved by the measurement of BLEU or bilingual evaluation understudy[12]. Another paper addressing the same issue in a different way is Phrase-Level Grouping for Lexical Gap Resolution in Korean-Vietnamese SMT[13]. This paper examines two different methods of grouping morphemes to improve the translations between two different types of language, one being morphologically rich, or containing complex sentences, and one being morphologically poor. The two methods were to use frequency data from the source to determine multiword expressions to extract, and to abstract morpheme groups, which group together similar multiword expressions. After employing part of speech tagging, to reduce gaps in translation, the BLEU scores of the translation were vastly improved, on a scale of 5-10% per trial[13].

Another method of improvement is adapting statistical models to implement rule-based translation, which improves the speed over rule-based translations. In the paper A Classifier-Based Preordering Approach for English-Vietnamese Statistical Machine Translation[14], the authors address modifications to improve reordering in statistical translation models. The model discussed in the paper uses classifier-based preordering, in which a statistical model is trained using part of speech tagging, to determine the reorder for the target sentence. The models were built using different structures of sentences to extract the rules from the data for later use. The results of the experiment showed that passing the text through the automatically generated rules and then implementing manually made rules increased the BLEU score of the translation by 2.5% over baseline[14]. In Syntax-Based Pre-reordering for Chinese-to-Japanese Statistical Machine Translation[16], the paper addresses tackling word reordering in translation sentences with a Syntax-Based approach. This research uses two methods, one for matching the head of the phrases in each language, and a second to devise rules based on a dependency tree of the language. The evaluations of sentences translated by these two methods test above the benchmark BLEU from the baseline system. The two methods proposed in this paper, however, are both prone to parse errors because they rely on automatic parsers which can disrupt accuracy in translations[16].

Within the idea of using a different algorithm in translations, a genetic algorithm-based approach was suggested by Ameur D., David L., and Kamel S. Genetic-Based Decoder for

Statistical Machine Translation[15] examines a genetic based algorithm when decoding sentences, which would be used in place of Moses current searching algorithm, beam search. The genetic algorithm has a population of "chromosomes" holding a translation, segmentation and alignment of a phrase. These chromosomes are built by using random segments of the translation sentences, and then passed to the crossover part of the algorithm. Some of the chromosomes use crossover where a word or two from each alignment gets exchanged between two chromosomes, and then some of the chromosomes are also mutated, where segments of phrases are changed using a phrase table to introduce variation. Chromosomes are then evaluated for strength and selected to go through the model again, until a suitable model is found. The algorithm evaluates better than the baseline system ~35% of the time, and equivalent to the system ~29%[15].

The experiment discussed in this thesis will focus on improving the speed of a rules-based translation system while maintaining the accuracy can be accomplished with either the hybridization technique, where the time complexity can be improved by introducing partial statistical implementation in an aspect such as selecting the rules in a more effective way, or with the use of an alternative algorithm i.e. the genetic algorithm approach to improve an aspect of the process such as identifying which word to translate more efficiently.

Chapter 3

Methodology

3.1: Apertium

Apertium is an open source rules base machine translation system that is highly adaptable for adjusting translations between languages[18]. The process was designed to be modular to allow different steps of the process to be included or removed.

The process of Apertium starts with a sentence in the source language, in the case of this experiment English. The sentence then can be de-formatted into raw text if needed, and then the text is passed to the morphological analyzer. At the morphological analyzer, the words in the sentence are cross referenced against the dictionary for the source language that Apertium has to determine what their base form will be, referred to as a lemma. A lemma is the base English word without any adaptation, such as “go” being the base form of “went”, or “vents” being turned into “vent”. The lemmas are put into a stream with a set of tags that show the correct modifications for the final part of the process in the destination language. The stream of tagged words is then sent to the lexical analyzer portion of the process, where the lemmas are translated directly from the source language to the target, with words that have multiple different translations be placed one after another. A bilingual dictionary is used in this process, and any lemma that do not exist in the dictionary are tagged with an asterisk to display that the word is unrecognized. The stream then goes to the lexical selection module where the decision between which of the multiple translations is correct is made, based from surrounding words in the translation. The word stream is then moved to the chunker, which is a multi-phase step to determine if a word order needs to be reevaluated when making the translation between languages with different structures, such as going from a Subject-Object-Verb order to a Subject-Verb-Object order. The Chunker tags the chunks for reordering, and the Interchunker does the physical reordering of the words inside the chunk. The final step of the chunking process removes the textual tags that identify chunks from the stream and passes the final stream to the morphological generator for adaptation to the target language. The morphological generator reads the tags remaining from the original part of the translation process and adapts the lemma in the target language to the final form such as adding genders and plurals and removes the tags from the stream. The words are then looked over one final time for small technicalities, such as adapting *a* to *an* when preceding a word that begins with a vowel or transforming *de el* to *del* in languages such as Spanish. The stream of text can be reformatted if needed, but the result is a final translation in the target language of the translation[11].

Table 1 and Figure 1 show an example walking through the process of using Apertium to translate a sentence, with simplified tags for a clearer example.

Table 1: Apertium Stream Format

Part of System	Current Stream of Translation
Beginning	I ran to the park and saw trees
Morphological Analysis	I<proper noun> run<verb><past tense> to<preposition> the<descriptor> park<noun> and<conjunction> see<verb><past tense> tree<noun><plural>
Lexical Transfer Note: Between dollar signs(bolded for emphasis) are the different potential translations of a word	Yo<proper noun> **$**corer<verb><past tense> ir<past tense> repetirse<past tense> durar<past tense>…**$** a<preposition> la<descriptor> parquet<noun> y<conjunction> $ver<verb><past tense> entender<verb><past tense> árbol<noun><plural>
Lexical Selection	Yo<proper noun> corer<verb><past tense> a<preposition> la<descriptor> parquet<noun> y<conjunction> ver<verb><past tense> árbol<noun><plural>
Chunker	Note: Spanish is traditionally an SVO language, as is English, therefore the chunker stage would be omitted in this specific translation, as the word orders do not need to be adjusted.
Morphological generator	Yo corrí a l parque y vi árboles.
Post Generator	Corri al parque y vi árboles.

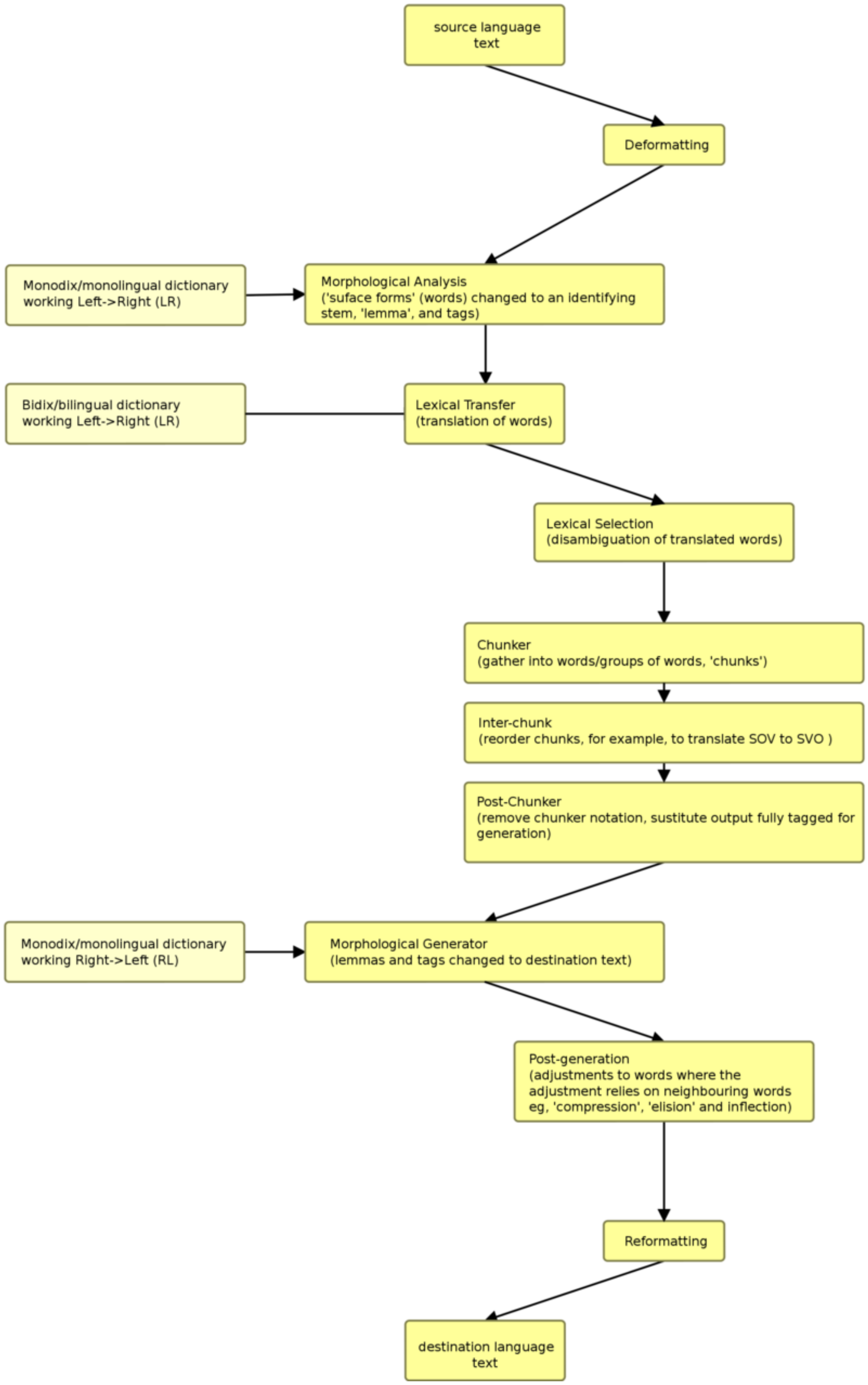

Figure 1: Apertium Work Flow Diagram[10]

3.2: Accuracy Evaluation

This experiment will use BLEU or the Bilingual Evaluation Understudy to determine the accuracy of the translation. A BLEU score is evaluated sentence by sentence throughout a translation, with a higher score corresponding with a more accurate translation. A perfect match of sentences, with exact wording, such as "The quick brown fox jumped over the lazy dog" and "The quick brown fox jumped over the lazy dog" returns a high score as opposed to matching with "Wheels turn around and around". The comparisons can be based on n-grams, where words can be matched in different phrase lengths to show different strengths of matching. Matching word for word does not necessarily prove the best translation was provided, as correct words could be included, but in a nonsensical or out of context order. Therefore, a BLEU score accounts for different lengths of n-grams, to show the quality of the overall translation, not just word to word. The simplified equation for evaluating a BLEU is to divide the matched number of n-grams over the total number of n-grams available to match.

A commonly cited example follows:

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

A traditional calculation is that there is a 7/7 match as the word the appears 7 times in the candidate. A modified calculation to determine the max count can be done, by evaluating the max number of times a word appears in each reference sentence. The appears twice in sentence 2, making the adjusted calculation 2/7 to show that the sentences are not just matching one word[9].

Chapter 4

Experiment

In this experiment, the various modules of Apertium were examined to determine if any modifications can be made to improve the overall process of the translation. Different modules throughout the process of Apertium can be adapted, to change the translation quality of the system. This experiment will use the Moses translation system as a baseline comparison as well as a standard build of an Apertium English to Spanish translation compared to an adapted version.

The first method of potentially improving the system was to adjust the rule set, to see if the style of matching affected the overall translation. A transfer rule is applied from a rules file inside of the Apertium translation system. Inside of that file, each rule is established with a specific pattern of tags such as noun-verb-preposition, to identify that the rule needs to be applied. These rules show how the words that match the pattern would need to be modified to adapt to the final form of the words in the target language, or language you are going to translate to. When the rule orders were adjusted, the overall quality of the translation was not affected, as different rule orders in a transfer file produced the same translation as the original unmodified rule file. This shows that rules can be reordered without dampening the quality of the translation.

The second method of improving the quality of the translation is to supplement the dictionary. The standard English to Spanish dictionary used in this experiment does not have the direct translation of vacuum, as in a standard household vacuum cleaner, to the Spanish word aspiradora. This translation exists in the dictionary as part of a multiword expression, but not alone, so when searching for the translation, if it does not find the expression, "pasar la aspiradora”, the correct translation of vacuum would not appear. When supplementing a dictionary, the translation must be added to both monolingual dictionaries, in this case the Spanish and the English versions, as well as the bidirectional dictionary that translates between the two languages. When adding to a dictionary where the words have a gender such as Spanish, tags must also be added to allow the machine to interpret the translation correctly, as shown in an example below. The tags aid the translation, as they get substituted into the word stream by the system during the morphological analysis phase, and used in the lexical transfer, as well as the morphological generation phase.

English example: <e lm =”vacuum”><i>vacuum</i><par n=”house__n”/></e>

Spanish example: <e lm=”aspiradora”><i>aspiradora</i><par n=”cuaresma__n”/></e>

Bidirectional Dictionary:

<e><p><l>vacuum<s n=”n”/></l><r>aspiradora<s n=”n”/><s n=”f”/></r></p></e>

Delving into what the above example shows, we will first break down the English and Spanish one language dictionary examples. The e tag designates a dictionary entry, with the word in quotations signifying what the input stream will appear to be to the reader. The i tag is the form of the word represented in the system. The par tag identifies the rules for modifying the word, such as how to make the word plural, i.e. vacuum becomes vacuums, in the same way house can become houses when pluralized. For the bidirectional dictionary, the p tag identifies a pair in a translation, with the l and r tags identifying the left and right sides of the pair in the translation, designating the separation of languages. The s tags are there to define the word in terms of parts of speech. On the English side of the expression, the word vacuum is tagged as a noun, whereas the Spanish side of the expression tags aspiradora as a feminine noun.

Adding the word to the dictionary did expand the understanding of the dictionary, but did not actually appear in translation testing samples, which leads to the third potential improvement addressed in this experiment: retraining the part of speech tagging.

Retraining the Part of Speech tagging module of the Apertium translation can be viewed as an overlap between the two main schools of translation. A corpus of hand-tagged sentences can be added to the system, which the system can use to learn from. The system examines the hand tagged data, which includes parts of speech, and generates a computer tagged file as it learns from the dataset. The computer then evaluates based on the orders of the parts of speech tags, a probability of which tags will be applied, namely when there is an ambiguous word. Returning to the vacuum example, vacuum can be a noun as in the absence of matter, an adjective used to describe the creation of a vacuum, or a verb as in to vacuum an area. On a sight read of a sentence, when the word vacuum comes across, the system has to determine based on the three word entries for vacuum which way to tag the system, as the Spanish words for different forms of vacuum are different: el vacío for the noun of an empty space, la aspiradora for the noun for vacuum cleaner, and pasar la aspiradora for the verb to vacuum. This poses an issue when the translation is going for the sentiment of cleaning and the translation decides upon an empty void.

4.1: Machine Learning with Hidden Markov Model

A potential solution to the problem of word ambiguity is to retrain the Part of Speech tagging section. During retraining, the system will scan a tagged file with parts of speech supplied and determine the orders that the parts of speech traditionally appear in. The training uses Hidden Markov Model (HMM), which uses two matrices to determine the probability of a part of speech appearing based on a set of potential ambiguities.

A Hidden Markov Model relies on Markov Chains, which can be used to predict the outcomes of events, based on events that have happened. The special case for a hidden model is that the intermediary states are not visible during the process. The parts of speech are not observed ordinarily in translation, but the tags are inferred from the words that comprise the sentence. Due to the fact that the transition states, as well as their probabilities are unknown, and calculating all possible state values is computationally complex, a forwarding algorithm is used to predict the potential state, by efficiently evaluating the probability of all potential paths that lead to the desired state. A decoding algorithm is used to evaluate the best path from the set of possible paths determined by the forwarding algorithm. The final stage of the process is using a forward-backward algorithm, to determine the probabilities of transitions based on a set of input data, which is re-evaluated as the states are processed[20].

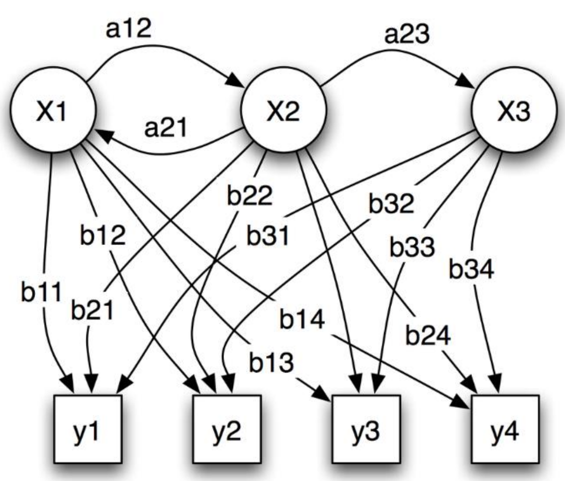

Figure 2: Hidden Markov Model Example[21]

The genericized model shown above in Figure 2, shows an example of a mapping of an HMM. The x stages are the hidden nodes in the model, where as the y stages are the final results of the model. The a-probabilities are the transition probabilities of moving between the various hidden states, whereas the b-probabilities are the resultant probabilities of moving from the hidden stage to the final results. How Apertium utilizes this model is explained in the next paragraph.

In the HMM for Apertium, the probabilities are determined by using word counts extracted from the corpus of tagged words supplied. The order of words in charted in the table by showing how many instances of a part of speech occur after another part of speech, such as how many verbs immediately follow nouns, as opposed to how many immediately follow prepositions. This is then turned into a probability table, where the likelihood of something such as a verb following a noun is calculated based upon the amount of times that this occurs, versus the number of times a verb appears after all parts of speech. For example, if there are 10 verbs in a data set, and 3 follow nouns, the probability of a verb following a noun in a translation is determined to be 0.3. The second matrix, referred to as an emission matrix, calculates probabilities in a similar fashion, but the emission matrix incorporates ambiguous tags. When a word can be found to be a noun or verb, it is originally tagged as a <noun>/<verb>, leading to an ambiguous tag. The emissions matrix follows the pattern of if the first tag is a <noun>/<verb> tag, what are the odds of the second tag being a preposition and completes the table accordingly. The system then uses a variation of the Viterbi algorithm to tag the sentences during translations, which evaluates the most likely outcome of tags given the set of probabilities and the surrounding known tags. For example, if a sentence is tagged as <noun>, <noun>/<verb> and then <preposition>, the Viterbi algorithm determines the likelihood of a noun given the tag is following a noun and preceding a preposition, versus the probability of the word being a verb. In this experiment, a set of approximately 13,000 words in various sentences that covered a wide array of topics were used as the training data. The retraining and recompilation of the language pairs resulted in a different tagging of similar sentences when compared to the default language set.

Chapter 5

Results

For the first trial of improving the system, the rule reordering, the overall system was not impacted. The sentences that were translated in both the baseline and modified rulesets translated to the exact same translations. The speed of the translation was also not changed, as the number of rules was not adjusted, just the order.

In the second portion of this experiment, where word translations were added to the dictionaries in the set did show minimal improvement on a sentence by sentence basis. This change was not reflected in an overall BLEU evaluation of a corpus, as the precompiled dictionaries cover large portions of the common English and Spanish lexicons. The quality of specific sentences were improved, such as identifying words that were not previously incorporated into the dictionaries, but the improvement upon words with multiple meanings came as a result of the third and final phase in retraining the Part of Speech tagging. The testing data for this experiment was a set of 200 sentences, in various sentence styles and structures, which was passed through each of the mentioned systems.

The part of speech tagging also improved a handful of sentences, mainly sentences with ambiguous word translations, on a sentence by sentence basis but did not improve the overall BLEU score of the corpus. As shown in table 2 below, Apertium modifications did not improve or modify the accuracy, but all 3 models improved the time taken to run the translation system. A note about the BLEU Score evaluations, the score of the Moses translation, did resolve to be higher than the Apertium score, which is not typical for these types of translations. This could be a result of an error on translation, or an error upon evaluation, or there could be missing words from the Apertium dictionary that were present in the Moses dictionary, that also showed up in the testing data.

Table 2: Results Table

Model	Time per Translation	BLEU Score
Moses	588.9 ms	28.92
Apertium Default Model	624.7 ms	26.29
Apertium With Words Added	597.9 ms	26.29
Apertium With Rules Reordered	602.1 ms	26.29
Apertium with Retrained Part of Speech Tagging	619.8 ms	26.29

The BLEU scores were evaluated using the Tilde Evaluator[17].

Chapter 6

Conclusion

While this experiment was not able to improve the time that it takes to evaluate with this system, there is still room to improve and better the way machines like this work. Starting with the trial of reordering the rules inside of a rule file. Reordering the rules was shown to not have a negative affect on the overall quality of the translation, showing that whatever order the rules are in the translation can return as a similar result. This means that if the pattern matching was improved, the time spent searching the rule files could be improved. If the part of the system that reads in the rule that the system is searching for was modified, as well as the file that contains the rules, the system could hypothetically use logic to zero in on possible rule matches faster. Searching for translations and rules appears to be the largest time consumer for translations.

The second potential enhancement investigated by this experiment, adding words to the dictionary, would not be a feasible solution to improving quality. Provided that the dictionary base is large enough that a wide percentage of the language is covered by the words in the dictionary, such as if the words in a dictionary encompass 90-95% of the common use of the language, adding edge-case words to the dictionary that are rarely used in everyday situations would most likely be beyond the point of diminishing returns given the rarity of the use of those words. This remains especially true with words such as be, which can be adjusted from be to either been, was, or are, depending on the situation. All the various forms count as separate words in English.

The part of speech tagging section of this experiment could be used to improve the overall accuracy, given an adequate dataset. A set of common sentences, tailored to show a proper distribution of parts of speech, as well as a mix of sentence structures, would make for a strong set of training data, which could improve the identification of words when changing between languages where the destination language can have multiple different translations from one source word.

This area warrants more research as the hybridization of statistical machine translation with a rules-based translation can ultimately be used to improve both the speed and translation accuracy of machine translation.

BIBLIOGRAPHY

1. “ISO 639-3.” *ISO 639 Code Tables | ISO 639-3*, iso639-3.sil.org/.
2. Nye, Mary Jo. “Speaking in Tongues.” *Science History Institute*, 5 Dec. 2017, www.sciencehistory.org/distillations/magazine/speaking-in-tongues.
3. Hillel, Yehoshua Bar. Report on the State of Machine Translation in the United States and Great Britain. Report on the State of Machine Translation in the United States and Great Britain, http://www.mt-archive.info/Bar-Hillel-1959.pdf.
4. Nagao, Makoto. A Framework of a Mechanical Translation Between Japanese and English by Analogy Principle. Elsevier Science Publishers, 1984, A Framework of a Mechanical Translation Between Japanese and English by Analogy Principle, http://www.mt-archive.info/Nagao-1984.pdf.
5. Hutchins, John. The History of Machine Translation in a Nutshell. 2005, The History of Machine Translation in a Nutshell, www.hutchinsweb.me.uk/Nutshell-2005.pdf.
6. Weisgerber J., Yang J., Fisher P. (2000) Pacific Rim Portable Translator. In: White J.S. (eds) Envisioning Machine Translation in the Information Future. AMTA 2000. Lecture Notes in Computer Science, vol 1934. Springer, Berlin, Heidelberg
7. “The Development of English.” *Introducing English Linguistics*, by Charles F. Meyer, Cambridge University Press, 2010, p. 36.
8. Blake, Barry B. “Russell S. Tomlin, Basic Word Order. Functional Principles. London: Croom Helm, 1986. Pp. 308.” Journal of Linguistics, vol. 24, no. 1, 1988, pp. 213–217., doi:10.1017/S0022226700011646.

9. Papineni, Kishore, et al. BLEU: a Method for Automatic Evaluation of Machine Translation. BLEU: a Method for Automatic Evaluation of Machine Translation, aclweb.org/anthology/P/P02/P02-1040.pdf.
10. “Workflow Diagram.” *Apertium*, wiki.apertium.org/wiki/Workflow_diagram.
11. “Workflow Reference.” *Apertium*, wiki.apertium.org/wiki/Workflow_reference.
12. Semmar N., Laib M. (2018) Integrating Specialized Bilingual Lexicons of Multiword Expressions for Domain Adaptation in Statistical Machine Translation. In: Hasida K., Pa W. (eds) Computational Linguistics. PACLING 2017. Communications in Computer and Information Science, vol 781. Springer, Singapore
13. Cho S.W., Lee EH., Lee JH. (2018) Phrase-Level Grouping for Lexical Gap Resolution in Korean-Vietnamese SMT. In: Hasida K., Pa W. (eds) Computational Linguistics. PACLING 2017. Communications in Computer and Information Science, vol 781. Springer, Singapore
14. Tran V.H., Vu H.T., Nguyen V.V., Nguyen M.L. (2018) A Classifier-Based Preordering Approach for English-Vietnamese Statistical Machine Translation. In: Gelbukh A. (eds) Computational Linguistics and Intelligent Text Processing. CICLing 2016. Lecture Notes in Computer Science, vol 9624. Springer, Cham
15. Ameur D., David L., Kamel S. (2018) Genetic-Based Decoder for Statistical Machine Translation. In: Gelbukh A. (eds) Computational Linguistics and Intelligent Text Processing. CICLing 2016. Lecture Notes in Computer Science, vol 9624. Springer, Cham
16. Han D., Martínez-Gómez P., Miyao Y. (2016) Syntax-Based Pre-reordering for Chinese-to-Japanese Statistical Machine Translation. In: Costa-jussà M., Rapp R.,

Lambert P., Eberle K., Banchs R., Babych B. (eds) Hybrid Approaches to Machine Translation. Theory and Applications of Natural Language Processing. Springer, Cham

17. "Tilde Custom Machine Translation." Tilde MT, www.letsmt.eu/Bleu.aspx.
18. Apertium. http://wiki.apertium.org/wiki/Main_Page
19. Moses. http://www.statmt.org/moses/
20. Jurafsky, Daniel, and James H. Martin. "Markov Models." Stanford.edu, Stanford University, 11 Sept. 2018, web.stanford.edu/~jurafsky/slp3/A.pdf.
21. Tdunning. "Hidden Markov Model." Wikimedia Commons, 2007, commons.wikimedia.org/wiki/File:HiddenMarkovModel.png, unmodified file, https://creativecommons.org/licenses/by/3.0/deed.en

ACADEMIC VITA

Kyle Burns

Education:

BACHELOR OF SCIENCE | PENNSYLVANIA STATE UNIVERSITY | MAY 2019

- Major: Computer Science
- Minor: Management Information Systems, Game Development
- Schreyer Honors College Member(June 2016-Present)

Academic Project:

- Drone 3D – Aerial mapping for business
    - Senior Project in development for Erie Insurance
    - Building a 3D Model of a house captured with a drone
    - Using machine learning to identify damage on home

Technical Experience:

- C++(1.5 Years), Java(1.5 Years), SQL(0.5 Years), Javascript(0.5 Years), Python(1 year)
- AWS(Intermediate), Visual Studio(Intermediate), UnReal Engine(Intermediate), Visio(Intermediate), Linux(Intermediate), TensorFlow(Basic)

Relevant Work Experience:

GRADER | DEPARTMENT OF COMPUTER SCIENCE | JANUARY 2018-PRESENT

- Grade for introductory and advanced Computer Science courses
- Coordinate closely with professors to ensure accurate grades

CONNECTED EXPERIENCES INTERN | AMWAY | MAY 2018-AUGUST 2018

- Worked with Continuous Integration for automated deployment
- Developed features for multiple Internet of Things services

STAFF LIBRARIAN | LILLEY LIBRARY | JANUARY 2016-DECEMBER 2017

- Used communication skills to complete orders and answer patron questions
- Responsible for closing procedures
- Processed transfers of materials between campuses

Extracurricular Experience:

- Resident Assistant at Penn State Erie, 2016-2018
- Founding member of Alpha Phi Omega Service Fraternity – Alpha Beta Nu Chapter(October 2015-Present)
- Staff Writer for Behrend Beacon(August 2015-February 2016)
- Silver, Silver Eagle Scout(Eagle Scout Award Attained: Oct 24, 2012, 6th Palm Attained: January 20, 2015)
    - Project was restoration of bird houses in Mingo Creek County Park in Washington County