2023

Simulating the Machine Translation of Low-Resource Languages by Designing a Translator Between English and an Artificially Constructed Language

Michaela Snyder

SIMULATING THE MACHINE TRANSLATION OF LOW-RESOURCE LANGUAGES BY DESIGNING A TRANSLATOR BETWEEN ENGLISH AND AN ARTIFICIALLY CONSTRUCTED LANGUAGE

A Capstone Experience/Thesis Project Presented in Partial Fulfillment

of the Requirements for the Degree Bachelor of Science

with Mahurin Honors College Graduate Distinction

at Western Kentucky University

By

Michaela Noel Snyder

May 2023

*****

CE/T Committee:

Dr. Trini Stickle, Chair

Dr. Elizabeth Winkler

Prof. Susann Davis

ABSTRACT

Natural language processing (NLP), or the use of computers to analyze natural language, is a field that relies heavily on syntax. It would seem intuitive that computers would thrive in this area due to their strict syntax requirements, but the syntax of natural languages leaves them unable to properly parse and generate sentences that seem normal to the average speaker. A subfield of NLP, machine translation, works mainly to computerize translation between different languages. Unfortunately, such translation is not without its weaknesses; language documentation is not created equal, and many low-resource languages—languages with relatively few kinds of documentation, most often written—are left with no way to effectively benefit from machine translation. As a step toward better translation processors for low-resource languages, this thesis examined the possibility of machine translation between high resource languages and low resource languages through an analysis of different machine learning techniques, and ultimately constructing a simple translator between English and an artificially constructed language using a context-free grammar (CFG).

Keywords: natural language processing, machine translation, context-free grammars, syntax, corpora, artificially constructed languages, low-resource languages

Word count: 13,510

I dedicate this thesis to my parents, Avis and John, who have given me everything I've needed to get to where I am today. I also dedicate it to my cat, Indy, who gave me much comfort throughout my life, and who I wish was here with me as I finish this accomplishment. Finally, I dedicate it to my partner, Lizzy, who helped me see the positives as I fell into the lows while completing this.

ACKNOWLEDGEMENTS

I'm extremely grateful for Dr. Trini Stickle for putting up with me for not only the year I was working on this thesis, but also throughout my undergraduate career as my self-designed studies advisor. You were always there to help me with whatever problems I came across. Things certainly became very difficult for me throughout the process, but you were always there to provide patience and guidance.

I'm also extremely grateful to Dr. Elizabeth Winkler for getting me started with my linguistics major at the beginning of my college career. Without you, I don't think I would have known exactly what I wanted to do, nor would I know what I wanted to study. You gave me a huge boost of confidence when I needed it most, and I'll always be grateful for that.

To Professor Susann Davis, thank you so much for helping me throughout this entire process. When I first began designing my linguistics major, you were there to help me understand the process, to answer my questions, and I'm very grateful for that.

Lastly, to all of my loved ones, thank you all for providing me with time to breathe when I was stressed and tired. I'm sure I would have suffered burn out without you all, and I wouldn't have been able to enjoy this project for what it is: my passion.

VITA

EDUCATION

Western Kentucky University, Bowling Green, KY — May 2023
- B.S. in Computer Science – Mahurin Honors College Graduate
- B.S in Semantics-Based Computational Linguistics – Mahurin Honors College Graduate
- Honors Capstone/Thesis: *Simulating the Machine Translation of Low-Resource Languages By Designing a Translator between English and an Artificially Constructed Language*

North Hardin High School, Radcliff, KY — May 2019

PROFESSIONAL EXPERIENCE

Technology Department, Union Pacific Railroad — May 2022-Present
- Hybrid Intern

Kentucky Mesonet, WKU — Sept. 2021-Aug. 2022
- Student Programmer

AWARDS & HONORS

Magna Cum Laude, WKU — May 2023
Computer Science Outstanding Sophomore Award, WKU — May 2021

PROFESSIONAL MEMBERSHIPS

Linguistic Society of America (LSA)

CONTENTS

List of Figures

List of Tables

SECTION ONE: LANGUAGE, COMPUTER PROCESSES, AND COMPUTER LANGUAGE

Introduction

Natural language processing (NLP), or the use of computers to analyze natural language, is a field that relies heavily on syntax. Syntax, or the study of grammatical structures, breaks down the internal structure of text-based sentences, focusing heavily on how the separate components interact with each other. Computers, as a rule, rely on such interactions to work. Take, for example, assembly language, an incredibly basic programming language, for which each seemingly simple command requires numerous lines of code. For instance, printing the classic "Hello World" to the console takes twelve lines of code, while in a computer-coding language like Python, a more modern programming language, it takes only one line of code. The main reason for this discrepancy in length is that assembly language interacts with each individual part of a computer's memory, moving data from memory register to memory register. The comparison, while relatively loose, is fairly accurate to how a computer must parse natural language; while not explicitly interacting with computer memory, each individual part of each sentence must be broken down and shifted into different structures in order to extract any number of pieces of information from the data.

Semantic meaning in natural language is deeply embedded within the structure, just as each individual command in assembly language tweaks the final result. It is how a sentence written in passive voice can have the exact same meaning as a sentence written in active voice; take, for example, the active sentence "the cat loves its owner." The textual representation has the exact meaning as the passive text, "the owner is loved by

the cat." This persistent meaning despite differing syntax, aptly named "deep structure," is one of the greatest difficulties in textual processing for machine translation. It is here that the similarities between linguistic syntax and computer syntax diverge; while sentences can have the exact meaning with different syntax for human interpreters whose experience with the language is contextualized by such things as the person's experience, emotion, setting, and culture, computers may not produce the same output when the order of the commands differ—the computer cannot bridge the differences with outside elements as a human does. This is compounded when the computer is asked to translate from one language to another, and this experience can be akin to humans learning a second language in which they first decode word for word and lack the semantic or cultural experiences to fully translate. Thus, if there is a sentence in, say, English, which has one meaning, and a sentence in English that has the exact meaning but a different structure, how would a computer translate that sentence into another language? The most brute-force "solution" would be to hard-code the different possible structures. In other words, the programmer would explicitly program each and every possible grammar structure into the code. This, however, is not feasible; computers have finite memory, while natural language allows for potentially infinitely mutable structures. Language evolves naturally and infinitely, growing with each new human experience, with every slip of the tongue, with every nuanced and different use. So, what is the solution? How might we program a computer translator to be better at language-to-language translation that captures the syntax and the semantics? This paper is an exploration of some of areas of exploration for written or textual translation. It cannot yet solve the problems but rather highlights areas for further development.

Thus, the questions are difficult to answer, and many in the field of artificial intelligence and natural language processing think they have stumbled upon one promising area: language models and bots. Each, however, struggles when confronted with colloquial language. A proposed solution would be to consider computer systems as language learners, using language acquisition research to build language models that prioritize grammatical phenomena in the order that humans learn them; another would be to use already created grammar models to build language models. This thesis focuses on using one such grammar model to translate written language between English, a high-resource language, and Ethanski, an artificially constructed language the author created specifically for this thesis (used to simulate a low-resource language).

Background

Grammar Models

Grammar models, in psycholinguistics, are generally used to best capture how the human brain uses and parses grammar; many formal models have been created – falling into the categories of either dependency grammars or phrase structure grammars – and each has a different, nuanced view on how grammar works. Some work well for teaching second languages, while others work well for syntactic research, and still others – namely head-driven phrase structure grammar (HPSG) and lexical-functional grammar (LFG) – work best when combined with computer science for natural language processing.

To define head-driven phrase structure grammar and lexical-functional grammar, dependency grammars and phrase structure grammars must first be defined. Phrase structure grammars focus more on the constituency of the words within a sentence, breaking the sentence into multiple, simplistic categories, with an emphasis on the locations of these constituents (Chomsky, 1956, p. 117). For our purposes, a constituent refers to an easily identifiable subpart of a sentence.

In contrast, dependency grammars focus primarily on the links between words, namely how strongly—or weakly—each word *depends* on the structural center of clauses, which is normally assumed to be the verb (Müller, 2020, p. 367). This differs from phrase structure grammars which focus primarily on the *constituency* of words within clauses from a linear perspective (Müller, 2020, p. 53). Constituency within syntactic theory,

particularly within Chomskyan Universal Grammar Theory, refers to the relationship between individual units that allows such units to function as a unit together (Carnie, 2007, p. 72). For example, a noun phrase (NP) can be composed of a noun as well as several other optional units, defined as follows: NP → (D)(AdjP+)N(PP), where *D* stands for determiner, *AdjP+* refers to adjective phrases repeated as necessary, *N* stands for noun, and *PP* stands for prepositional phrase. To more clearly present the difference between the two, it could be said that dependency grammars focus on the dependency between every word in a clause and its structural center while phrase structure grammars focus on dependencies between each word within a clause, regardless of any perceived structural center.

With that being said, both head-driven phrase structure grammar and lexical-functional grammar are soundly within the realm of phrase structure grammars, but not without references to dependency grammar. Furthermore, both contain further nuance between the generative-enumerative and model-theoretic approaches. Generative-enumerative approaches—reserved primarily for phrase structure grammars and other similar models—are built upon the concept of being able to generate strings of symbols (or words) in order to derive any well-formed string (Müller, 2020, p. 509). This is similar to the computer science theory of CFGs, where a simple language can be described using a series of production rules where derivation involves taking a starting symbol and generating the target string step-by-step. To illustrate, one could imagine having a string with different colored marks up and down the length, adding beads to the string accordingly; when you reach a certain color, you know to change the bead to place on the string. This is similar to how a computer processes a context-free grammar.

When originally created, lexical-functional grammars within computer software were intended to be generative-enumerative (focused on generating all possible strings within a language), but further adaptation through research changed its form to model-theoretic. Model-theoretic approaches are the direct opposite of generative-enumerative approaches; rather than specifying rules that must be followed in the symbol/word selection process, model-theoretic models constrain lexical items and allow everything that is not outright banned by the constraints (Müller, 2020, p. 510). Referring back to the string example, this string would allow beads not explicitly denoted by the colors on the string, as long as the beads were not explicitly disallowed. Both head-driven phrase structure grammar and lexical-functional grammar are classified as model-theoretic languages, but both fulfill this in different ways.

Lexical-functional grammar structures are, first and foremost, strict and explicit. They focus primarily on *describing* the relationship between words and their functions within a phrase or clause within the theoretical domain of mathematical modeling (Kaplan, 1995). Each aspect of a sentence is broken down into its components according to subject, object, and predicate, with additional modules that can be added in order to classify tense, perspective, and number. For example, the sentence

The silly dog **gave the contentious cat a mimosa after dinner**.

NP			V		NP			NP		Prep Phrase	
DET	ADJ	N	V	DET	ADJ		N	DET	N	P	N
S							IO	DO			OP

[prepositional phrase = adverbial]

The complete predicate is emboldened. Here the functions not the grammatical categories (e.g., noun, adjective, verb) are of import.

This calls to attention the lexical aspect of the model with the emphasis on the importance of individual lexical items; it is a lexicalist idea, that each word adds a significant amount of information to the semantics of a sentence. An important summarization of the point of lexical-functional grammar is that the importance is based on the relationship between all lexical items within a sentence, not their position in the sentence (Müller, 2020, p. 228). So, in the sentence above, the relationship among the determiner, adjective, and noun to form a noun phrase functioning as the subject versus the determiner, adjective and noun that forms the noun phrase that is functioning as the indirect object and so on is essential.

Similar to lexical-functional grammar, head-driven phrase structure grammar structures are strict, explicit, and highly lexical. However, rather than focusing on the subject, object, and predicate relationship, head-driven phrase structure grammar structures focus primarily on compiling phonological, syntactic, and semantic information for each lexical item (Müller, 2020, p. 268). The selection criteria are generally programmed to focus on extracting words according to their place within a sentence, and, since the grammars are informed by written corpora, the extracted words are then checked against the stored data, determining whether or not the word is capable of fulfilling the sentence placement inferred during extraction.

Head-driven phrase structure grammar structures were created almost specifically for computational linguistics and natural language processing, and this is evident in the structure of the model. As a general rule, head-driven phrase structure grammar does not

allow for overcomplicated linguistic ideologies in the attribute-value matrices; this grammar rules out structures like movement and the idea that all branches of a structure are binary, and ultimately leads to simpler, flatter structures that are easier to parse (Müller et al., 2021, p. 5). Emphasis is placed on the form and meaning of words, compiling them into a larger structure that shows the form and meaning of entire sentences or clauses (Müller, 2020, p. 268). Thus, the noun cat might be broken down into subcategories such as noun, singular, animate, animal, feline.

It is inefficient to create an entire diagram every time a sentence needs to be translated, but these models are useful in creating the underlying system of machine translation software. For the heavy-lifting of machine translation, various methods are used, mainly neural networks, statistical models, and CFGs. Time constraints for this project removed the possibility of using neural networks and statistical models, leaving CFGs as the only feasible method. To be clear, CFGs are most similar to the phrase-structure grammar model, as noted above, that were developed by Noam Chomsky, where sentences are modeled using each phrase and each component (Chomsky, 1956). These grammars can easily be written in syntax tree form (can be seen in *Fig. 1 and Fig. 2*), and this is often how they are seen when not written for computer science purposes. In computer science, CFGs are written primarily for programming languages and their syntax and also hold great theoretical value within the more theory-motivated areas of the discipline (Dassow & Masopust, 2012). In computational linguistics, however, they are used often as a form of parsing. When used for the purposes of machine translation, the grammars have to be made for each language, with each word in each language being

separated according to part of speech. In order to fully analyze the part of speech of each word, lexical resources are usually needed.

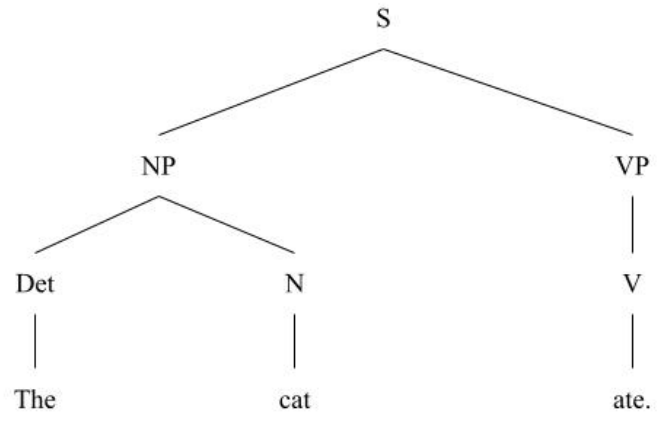

Fig. 1

In *Fig 1*, “The cat” is the subject, while “ate” is an intransitive verb, or a verb that does not take an argument, or object.

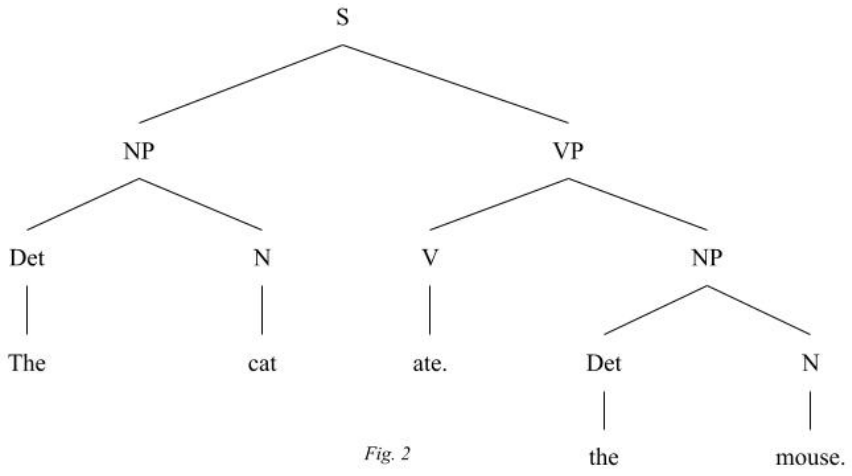

Fig. 2

In *Fig. 2*, “The cat” is still the subject, but the verb “ate” becomes monotransitive, or, a verb that takes one argument, which is “the mouse,” the direct object.

The Computer as Language Learner

Traditionally, many of us would not think of computers as language learners, or, rather, not in the way that is often meant. When used to describe human speakers, it usually means a person that is actively learning a language, whether they be a child learning their first language, or an adult learning their second, third, or even fourth. When used to describe computers, it generally means a computer that is being actively *taught* language, albeit in a different manner than that of human learners. In modern discourse, most language-learning "computers" are chatbots, or other forms of artificial intelligence (AI).

Chatbots can be defined as computer programs that use various methods to communicate through text with humans. To go more in-depth, most chatbots are one of three kinds: rule-based, intellectually independent, and AI powered (Lishchynska, 2022). Rule-based bots are, by definition, the most constrained of the three models. These bots are normally constructed with a choice-based graphical user interface (GUI) and usually constrain the user–or customer–to the choices that are available (Lishchynska, 2022). These bots are seen most often on business websites and are used as the default bot for such sites. Intellectually independent bots, although not as advanced as AI-powered bots, are still quite a step up from rule-based bots. They are still constrained, but they allow the user to type their inquiries, and then the program parses the customer's input for certain keywords (Lishchynska, 2022). These keywords are then connected to pre-programmed answers, which are then given to the user. Finally, AI-powered bots are the most advanced model and combine the best aspects of the rule-based bots and the intellectually independent bots. They use machine learning, AI, and natural language processing (NLP)

in order to parse user input and output relevant responses (Lishchynska, 2022). However, before these three models were fully developed, most bots were just scientific curiosities and their rudimentary forms could only reproduce pre-programmed language—they were not producers at any level of the output.

The history of chatbots is quite extensive and begins with the influential works of Alan Turing. He argued that although it seems quite impossible to give a computer the full capacity of thought, it may be possible to give them the ability to imitate human communication (Zemcik, 2019). With his work, he brought into light one of the most compelling goals in computer science whose test became known as the Turing test. The Turing test required that a program be indistinguishable from a human in conversation. Following this, the first chatbot, ELIZA, was made at MIT with the purpose of providing psychiatric help to those who needed it and was even regarded by many to be an actual person (Shum et al., 2018; Zemcik, 2019). Then, the Parry bot was created and passed the Turing test for the first time, by providing human-like responses to the proctor during the test, creating a milestone that marked a turning point in the research (Shum et al., 2018).

What Problems Remain in Natural Language Processing (NLP)

Human linguistic abilities are unique, as is evident in the many complexities of language and meaning that humans most often find no difficulty understanding (but, of course, they do still often have difficulties understanding each other's written and spoken language). In contrast, current NLP experiments fail to replicate understanding of these complexities; although this could be partially solved by programming multiple connotations and structures, that is, as previously stated, only a partial solution (Wallis,

2011). Furthermore, the sheer amount of data needed in a program to produce fewer errors would most likely render it inefficient and difficult to repair. Of course, the best solution would be to completely replicate human linguistic abilities, but this would require a complete understanding of human cognitive abilities and the neural networks that enable these abilities (Heinrich & Wermter, 2018) for which we still lack. Doing so would be similar to receiving a piece of unknown technology and having to reverse engineer it in order to replicate it. Many computational linguists try to work around this type of engineering by programing computers with extensive "dictionaries" in a misinformed attempt to give software the ability to understand and comprehend the many facets of words. The deficit of this approach is that the computer program may still select the inappropriate word from a list of choices since it still lacks all the experiences—emotional, setting, culture of semantics (meaning) and pragmatics (context)—that a human uses.

As the technology develops, accuracy does improve, although this is mainly due to the incorporation of deep learning and machine learning techniques, and not due to the efficacy of the dictionaries themselves (Kalra et al., 2022). The dictionaries do little to actually improve the comprehension skills of their programs. Unfortunately, these computational linguists are mistakenly conflating knowledge of denotations with full comprehension (i.e., semantics and pragmatics), though this is obviously not their intention, and recent forays into machine learning and deep learning are providing new avenues that improve the efficacy and efficiency of such dictionary systems.

In basic information extraction systems, these memorized definitions could be useful in extracting relevant data from a set, though it is not very useful outside of these

basic situations. Even in these basic situations the usefulness is not absolute; in the case of metaphorical phrases, denotations are not of much use at all (Russell, 2016, p. 37). Metaphors are simply too unpredictable for even a machine-learning algorithm to predict possible meanings from all denotations of any word (Russel, 2016, p. 53). Slang is even harder to incorporate than metaphors, due to the fact that a lot of slang pays no mind to denotation. Take, for example, the phrase "that's lit"; although it means "that's interesting," it is difficult to figure out the meaning from denotation alone, and current software is not capable of including context during analysis. In NLP, this inability to analyze figurative language is a debilitating limitation. This would require computational linguists to be able to program the linguistic models in such a way that the model's grammar usage is not only functionally motivated, but semantically motivated as well (Periñán Pascual & Mestre-Mestre, 2016, p. 240). Semantically-motivated grammar is reminiscent of human language usage; as such, effective NLP programs would naturally replicate the human way of utilizing meaning (semantics) and context (pragmatics) with grammar.

Currently, some of the better computer-powered interpreters are several NLP/artificial intelligence (AI) programs now available to the public like Cleverbot™. Still, these programs, while intended to respond logically and coherently, often respond in illogical ways; these moments make it very easy to remember that they are not, in fact, humans with humanlike linguistic capabilities. In the case of personal assistants, there are several examples of their inability to answer what the average person would consider a "simple" question, like "is the sky blue?" (Grosz, 2018). The flaw is in the lack of available comprehensive rules that can be programmed; obviously, the solution does not

lie in creating generic rules that cover the basic knowledge or basic syntax of a language but rather the crux lies in those slightly complex parts of language, as the programmers have not worked out for their machines at this point. To answer "Is the sky blue?" the program must be able to understand the subjectivity of color interpretation, and, unless the program possesses a camera with the ability to interpret light wave frequencies, it would be unable to give an answer that it created itself; rather, the answer it would give would be based on what the training data taught it, not on what could actually be perceived in that moment. In other words, if a program is taught by training data that the sky is blue, but the sky has suddenly turned red, it would be unable to accurately tell you what color the sky is.

Functionally, computational linguistic theories on NLP systems have changed since the endeavor first began, and these changes have been significant, but they are seemingly not significant enough to really matter in analyses like the Turing test (Grosz, 2018). At this point, the limitations far outweigh the current uses, although with further research this could very well not be the case. The necessary research, however, would need to be in something other than writing general linguistic guidelines for NLP systems to follow (Heinrich & Wermter, 2018). It could be proposed that the research be done, instead, focus on furthering our understanding of how humans communicate, and the neurological components required in hopes that the knowledge can be applied to NLP systems.

As stated previously, a lot of the problems in current NLP systems lie in the inability to accurately discern meaning from any given input that strays from denotation. This, of course, is not a problem when humans consume linguistic data; it is quite

difficult to find someone who is always literal, so many people have had to learn how to understand figurative language. This is a disadvantage that NLP systems must face. Unfortunately, as noted previously, very little is known about the neurological components of human linguistic communication, so there is quite a bit of work to be done before the information is useful for computational linguists (Heinrich & Wermter, 2018). For example, there is very little understanding of how the brain rationalizes grammar structures that do not generally follow logical patterns, like prepositions. Utilizing knowledge of neurological processes would, as stated previously, allow computational linguists to replicate said processes and create effective NLP systems that are actually capable of human-computer interaction (Heinrich & Wermter, 2018). Human language capabilities can be narrowed down to certain regions of the brain like the superior temporal gyrus and the inferior temporal gyrus; assuming the neural connections can be "reverse engineered" and replicated in a simulated neural network made of a series of programs, the NLP systems that incorporate this technology would be able to utilize language in an almost human way (Heinrich & Wermter, 2018). This would, of course, bar the parts of language that are built upon personal experiences, as NLP systems would be hard pressed to experience such things.

Furthermore, much of language is based upon sensory interactions, something that would be unique to human speakers. Considering NLP systems are pieces of internal software, it would be difficult – if not impossible – to create those kinds of sensory experiences for the systems to draw upon for context (Deemter, 2016). For example, in terms of referential statements, adjectives generally work on a gradient (Deemter, 2016); due to the fact that NLP systems cannot and have not experienced the relevant contexts,

without some kind of failsafe they will be unable to properly process the given information (i.e., as machines, they cannot tell whether the sky is blue). This can be bypassed by tweaking the hypothetical neural networks to give the NLP systems the ability to utilize machine-learning technology to comprehend and utilize figurative language on a gradient. There is still, however, the problem of the NLP systems comprehending referential statements in the first place. The gradient adjectives are functionally vague with or without context, mainly due to the fact that the foremost context is less important than the personal context of the speaker (Deemter, 2016). Current computational models have great difficulty in even identifying these vague descriptors; there are, generally, very few accurate linguistic models to base the necessary grammatical guidelines on (Bobrow & Simmons, 1970; Deemter, 2016). It could be entirely possible to create a lexicographical database of referential adjectives, but language growth is quite a large barrier in that regard, though deep learning provides a possible solution that can surpass computational and semantic limitations (Kalra et al., 2022). A better solution, perhaps, would be to create a set of guidelines that effectively pick out referential adjectives, and to utilize machine-learning to give software the ability – once the adjectives are selected– to store the contextual usage of the adjective for future use.

NLP systems, although mostly used for long strings of written and/or verbal input, could possibly be used to analyze dialogue. Unfortunately, this could be fairly problematic due to the current nature of NLP systems alone. Assuming steps are taken to help the systems understand referential statements, there still exists the inherent difficulty in understanding the multiple layers of semantic and pragmatic components of human

language. People employ language in ways that are contradictory to the standard forms of their language (Grosz, 2018). Furthermore, in written input, it could be easy for an NLP system to mistakenly conflate the subject of one speaker's sentence with the subject of another as a result of the incomplete sentences often spoken by native speakers (Grosz, 2018). The human mind is capable, however, of making these distinctions, and through providing NLP systems with a way to properly simulate human neurological function computational linguists could bypass this entire issue. Unlike basic information extraction, analysis of dialogue requires an understanding of intention (Grosz, 2018; Lehnert, 1997). Assuming it is possible to program the ability to understand intentions, the actual algorithm for analyzing separate speakers is as simple as a basic stack data structure (Grosz, 2018)—in other words, the algorithm could simply stack the speakers in their speaking order, allowing the program to then analyze each speaker individually. Of course, this simplicity hinges on the ability to extract intention from context alone. This is a little more intensive than artificially assembling the "neural connections" through, say, linked nodes; rather, this would require the ability to simulate abstract thought through programming, which is something no one really has an idea on how to accomplish yet. However, that is only due to the fact that no one knows how thought works.

Current machine-learning capabilities have allowed current NLP systems to effectively learn meanings without the need for programmed lexicons. Instead, NLP can rely on actual examples of language usage to learn how to create sentences. This is a necessary strategy, but it also raises the problem of storage. Language grows almost exponentially, so expecting any system to have enough storage for a possibly infinite amount of information is far too much to ask. This can be bypassed through having

multiple layers of modality—or human inputs, such as typing, mouse, eye tracking, etc.—in a system and storing such information in linked nodes (Heinrich & Wermter, 2018). Modality would provide the opportunity to replicate the multiple layers of human sensory interaction, which would allow NLP systems to further any comprehension of human language (Heinrich & Wermter, 2018). Furthermore, considering the difficulty in referential statements, neural modality would allow NLP systems to understand the gradient, thus providing another level of usable comprehension in things like complex information extraction (Heinrich & Wermter, 2018; Deemter, 2016; Lehnert, 1997). A base lexicon, however, would be necessary in order to give any NLP system the ability to understand at least the simplest of sentences, which would enable basic comprehension.

Considering this need for a base lexicon, it would be conducive to provide the base lexicon that young children assemble rapidly during the first few years of life (National Institute on Deafness and Other Communication Disorders, 2018). Of course, every child experiences a different life, and thus receives a different lexicon, but there is a base lexicon (as well as structure) for shared language speakers; this would be the lexicon that would be programmed (Guthrie et al., 1996). The goal would eventually be to have multiple linked nodes for every word in the base lexicon; considering linked systems utilize far less storage than a stack, queue, or binary tree data structure, this would not be entirely inefficient, nor would it be cluttered and difficult to repair or edit. Utilizing the base lexicon in this way, it could be possible to have entire linked lists that specialize in specific areas of language. It would be similar to if the human brain was compartmentalized. Unfortunately, even modularity will not fix the seemingly inherent inability to comprehend emotional language (Bobrow & Simmons, 1970). Through

utilizing a base lexicon, computational linguists exclude a wide variety of emotionally charged language, which by extension also excludes a variety of emotionally charged documents (Grosz, 2018). Due to the potential widespread usage of NLP systems, excluding any possible uses is dangerous to the overall functionality of the technology. Machine-learning would not be enough to provide all possible meanings for emotional vocabulary, as emotional vocabulary is usually on a gradient, like the vocabulary in referential statements. Unlike referential statements, however, these emotional statements are far more reliant on personal context than linguistic context (Bobrow & Simmons, 1970; Deemter, 2016). It is far beyond current technology to simulate emotional context in any way, so this provides quite the unique challenge for not only computational linguists, but neuroscientists as well.

Rather than simulating emotional contexts, it could be possible to create baselines for each possible emotion and quantify them; providing such information would provide basic information for NLP systems to build onto. This could work similarly to the basic lexicon. Furthermore, this could utilize the system hypothetically put in place to deal with figurative language. In a testing scenario, NLP systems could even be briefly programmed to work in a way similar to a chatbot to take in as much input as possible (Grosz, 2018). Utilizing this testing environment could give NLP systems the ability to learn as much information as possible to store in the linked nodes (Lehnert, 1997; Heinrich & Wermter, 2018). It could be seen in comparison to how children subconsciously strive to encounter as much language as possible. Utilizing neuroscience to learn how children do this and eventually applying this to NLP systems would allow computational linguists to streamline the process and allow their software to learn not

only vocabulary but grammatical structures in an even more logical and efficient way (Grosz, 2018; Heinrich & Wermter, 2018). If this were the case, the comprehension skills of NLP systems would be far more efficient and useful than current systems.

Semantically, NLP systems fall short, and they will continue to fall short as long as they use only functional grammar rules to parse input. If these systems are given semantically-focused grammar alongside the base lexicon and linked node system, they would be far more prepared to properly analyze any input (Lehnert, 1997). Linguistically, humans communicate through semantically-focused grammars rather than functional grammars; following this, it would only make sense for NLP systems to work similarly, if not in the same way, considering these systems would have to process natural human language (Lehnert, 1997). Unfortunately, most people do not consciously think about the grammar that they are using; rather, they are so used to speaking grammatically (whatever that means to them) that they can correctly follow their language's grammar rules without any difficulty. As such, computational linguists will have to consciously think about the grammars that their NLP systems will need, and they will need to learn how to replicate the effortless usage in their programs (Lehnert, 1997; Heinrich & Wermter, 2018). Grammars, however, have to be fairly general in order for them to be utilized to the extent of the NLP system's ability (Lehnert, 1997). If written generically enough, a proper NLP system can analyze input far more efficiently than a human (Lehnert, 1997). Written language is the simpler of the two between written and spoken; this would be more effective with written language, although this is more to do with the difficulty surrounding accents than differences in grammatical structure.

For all future uses of NLP systems, it is integral that they be as efficient and effective as possible. Any error in analysis could be dangerous depending on the situation (Grosz, 2018). Take, for example, someone asking for emergency services to be called; if the NLP system is unable to properly analyze the request, it will ultimately be unable to oblige (Grosz, 2018). There are several instances where any request is required by a system to be worded in a specific way. Due to differences in lexicon or dialect, many users would be denied usage of a system. This is the importance of creating generic semantics-focused grammars. An ineffective system might simply extract information and do what it is programmed to think is the correct thing to do. This, of course, is not the solution, as it would ultimately cause more problems than it would solve (Lehnert, 1997). For example, if someone were to ask a personal assistant system a random, nonurgent question about a local hospital, a system that functions in this way might be programmed to call emergency services whenever the word "hospital" is mentioned. This would ultimately cause far more harm than good.

Unfortunately, much of the issue in NLP is that computers are treated as being fundamentally separate from humans in terms of language learning, leaving us with very few efficient and accurate ways to deal with computational understanding of natural language. These issues deeply permeate the field of NLP, but the issues are most glaring in machine translation due to the necessity of reconciling differences between differing languages.

Issues in Machine Translation

Ultimately, all of these issues cause very real problems for machine translation, affecting the accuracy and efficiency of translations. In machine translation, several

issues require addressing, like culturally insensitive and generally inaccurate translations. Even now, computational linguists are only somewhat closer to creating software that can accurately translate any sentence in any language to any other language; note that in this scope, "accuracy" refers to semantic accuracy. Many who have taken foreign language courses have experienced this inability to take in proper semantics, namely in translation homework; it can be difficult to translate some sentences through software like *Google Translate* without the full context, which is not often given in language homework. These issues and errors are due to the inherent complexity of human language; there are layers of complexity and context hidden in every clause, each dealing with different aspects of personal cultures, experiences, and emotions. Unfortunately, these things remain inimitable due to the general lack of knowledge surrounding how the human brain works when dealing with language. However, that is not to say that developing machine translation software capable of semantically accurate translations is impossible; instead, this just provides an interesting puzzle that will likely require the combination of multiple structures. As such, although there are several issues regarding these lexical resources, it is ultimately the most informed way to accomplish accurate machine translation systems that respect semantic and cultural meaning.

In translating from one language to another, a certain level of affectation (emotional meaning) is lost; this is ultimately due to the lack of understanding regarding affective, or emotional, language in machine translation software (Valitutti et al., 2004). Much of the issue is that communication of any kind requires massive amounts of context, and it is difficult to give these contexts to machines; a lot of context is acquired through years of experience communicating with others, but researchers are expecting

programs to acquire this context within significantly shorter amounts of time. Previous attempts to facilitate accurate machine translations included the incorporation of a machine-readable dictionary or a lexical knowledge database; these attempts – specifically *WordNet* by George Miller – tried to create a sort of referential database that focused on synonyms, antonyms, hyperonyms, hyponyms, meronyms, and the like (Valitutti et al., 2004). The database was stored in a sort of "lexical matrix," or a matrix that specifically recorded the relationships between word forms and their connotations or denotations (Valitutti et al., 2004). However, lexical knowledge databases are best used when utilized as a reference for human-computer interactions (HCI) rather than an overarching solution to the HCI issue as a whole.

Beginning with these sorts of lexical knowledge databases – and, specifically, their role as lexical resources – the issue of taxonomies must be discussed. Language exists in categories that are simultaneously rigid and flexible, such as in the case of verbs being nouns (e.g., gerunds); these lexical resources are required to be both specific and intentional when classifying words in the target languages (Guarino, 1998). However, when these lexical knowledge databases were first being built, they ran into the issue of ISA overloading, or the overwhelming of the database in terms of Is-A relationships (Guarino, 1998). There is the issue of oversaturating the database in reference to words and any possible polysemy; one word may have multiple meanings, but to separate those meanings may remove access to a possible meaning of a word. Such removal would ultimately result in incorrect translations when presented with a particular context.

Another possible issue – in terms of *WordNet* – would be the almost required overgeneralization to place entire lexicons into categories. Weaker definitions of rigid

classifications would be necessary based on the words being used due to every word not fitting into strictly one category (Guarino, 1998). Although *WordNet* starts from heavily general categories and narrows them when necessary, there would also be the need for categories so specific they include maybe one or two words; this, however, would be inefficient and take up an inordinate amount of memory. Furthermore, the necessity for affective language mastery in a system built for HCI would require lexical sorting on an "emotional" level (Valitutti et al., 2004). Although using machine-readable dictionaries to create these lexical knowledge databases is the general course of action, it ultimately leaves out the emotionally charged aspects of language and thus gives translations the short end of the stick (Neff & Mccord, 1990; Valitutti et al., 2004). As such, the ideal solution seems to be the combination of these techniques combined with another that would allow for the acquisition of figurative connotations as well as literal denotations.

In acquiring figurative connotations and literal denotations, two strategies have followed from this problem's realization: statistical approaches and neural network approaches. Beginning with statistical methods, many programs exist that analyze corpora and do something with the data, such as joke programs that analyze entire seasons of shows and subsequently generate their own episode or scene based on the data observed. At the outset of these programs, most relied on a statistical approach, analyzing the words and phrases used and creating a sort of statistical model that would allow for recreating something according to how it is already been created (Brown et al., 1990). However, this is an inexact science; although the ability to calculate the probability of a word existing in a translation given the words that preceded it is there, it is only a probability and is thus not guaranteed to be correct (Brown et al., 1990). Assuming the

translation software would be searching through the lexical resources at its disposal, difficulty could arise when there is no direct translation for a word or phrase. This scenario is something even human translators have difficulty with; the lack of metalanguage for some concepts is a difficult hurdle to get over and one that statistical analysis is not well-equipped to jump due to its inability to process language outside of what is statistically significant. This leaves the translation software vulnerable against sentences outside of its training data. Here lies the question of human language recursion: Can software translate structures for which it has not been programmed? Or, can we now program enough of the structure of a given low-frequency language for the computer program to fill in the gaps and translate a written sentence into or out of that low-frequency language?

Furthermore, since machine translation is a highly literal, or word-to-word-process, testing would almost certainly result in the translation of "nonsense sentences." These nonsense sentences – or sentences that have no real meaning but are grammatically appropriate, such as "colorless green ideas sleep furiously" – while ultimately meaningless, these sentences help test the accuracy of translation software. This proves to be another issue regarding the statistical model for machine translation; its abilities prove to be valid *only* in pragmatic senses due to its need for aligned multilingual corpora (Brown et al., 1990; Koehn, 2005). Aligning corpora in the sense of a database is remarkably simple when considering languages such as English and French – other than a few particle outliers in French, the word order is similar enough to have a 1:1 alignment between the two languages in which the order of the sentence remains the same in the translation (Brown et al., 1990). However, statistical machine translation heavily relies on

parallel corpora; without such corpora, the translation accuracy suffers (Koehn, 2005). It is similar to the Rosetta Stone; the corpora provided by such a stone would be parallel across Greek and Egyptian and thus would be able to translate fairly accurately using the data found on the stone.

Neural networks, however, have – in recent years – become valuable structures in many different subfields in machine learning. Neural machine translation has been adopted by many companies that have produced translation software like Google; it is crucial, however, to remember that neural machine translation is not a "holy grail" and possesses its own set of drawbacks (Wu et al., 2016). Rather than using the corpora from the target languages to construct statistical models, the neural networks use the corpora to learn how the words are used and attempt to use this information to create accurate translations (Bahdanau et al., 2015). Neural networks, generally, are trained through HCI instead of through compiled corpora, and this results in less training time, the training is more intensive, and increases the overall cost of the model (Wu et al., 2016). However, these neural networks ultimately cut down on the number of separate models needed to complete one translation; statistical machine translation, as a general rule, requires multiple models to accurately translate from one language to another. The phrase to be translated goes through numerous models to be translated into the target language, while in neural machine translation, one large network is trained to minimize the number of models needed (Bahdanau et al., 2015). When correctly done, neural machine translation takes only one model.

In terms of the weaknesses of neural machine translations, they fall prey to three significant flaws: slow training speed, inefficacy when dealing with uncommon words,

and, in some instances, failure to translate all of the source sentence(s) (Wu et al., 2016). As a relatively new strategy in machine translation, not much has been accomplished in terms of solving these issues; because of this, many of the current instances are experimental and may or may not be ideal for use outside of a research setting.

The statistical model currently shines when presented with large corpora compared to the neural network model; this is due to its inherent ability to analyze each word in the initial language as it attempts to translate into the target language (Brown et al., 1990). However, the neural network model finds difficulty translating large corpora due to its large number of required parameters (Wu et al., 2016). Despite this, the neural network model – if "perfected" – would end up being the ideal model due to its end-to-end nature, or, simply, due to its ability to completely bypass any inner processes and learn all internal processes concurrently rather than sequentially, similar to acquiring multiple skills at one time instead of learning them one after the other. Interestingly, while the statistical model translates straight from the initial language to the target language based on the statistical likelihood of each word in a phrase, neural machine translation encodes and decodes the sentence in the initial language in an attempt to make the translation more accurate and able to vary between input length and output length (Bahdanau et al., 2015). This encoding/decoding technique also allows the same neural network to include multiple decoding techniques to decode the same sentence from the input language into several different languages (Bahdanau et al., 2015). This makes even just one neural network far more versatile than a statistical model due to the inherent limitations in a statistical model because they have to be specifically developed according to specific corpora and specific database tables with the appropriate probabilities.

Assuming a neural machine translation model that explicitly works between English and Russian, translating the sentence “I love to rest” does not employ resources beyond the statistical mode, which, while accurate in this scenario, can cause issues in others. In this translation, the input “I love to rest” would result in the output “я люблю отдыхать.” Using the *WordNet* strategy of defining classifications of the sentence, it first begins with a functional parsing of the propensity (statistical likelihood) for grammatical classes to fill functional roles. From this analysis, the neural network would then encode each word from the input according to their part of speech (Goldberg, 2017). As such, the word “I” in subject position would then be encoded as a pronoun, “love” would be encoded as a transitive verb, and “to rest” which then follows that sentence’s transitive verb, the two-part infinitive is subsequently encoded as a noun/noun phrase filling the direct object slot. Difficulty arises when there is tension between classification of words’ functions and their grammatical classes.

When decoding the input into the output, another problem area could be matching multiple words into possible singular words in the target language. In the given example, “to rest” is matched with “отдыхать.” The grammar selection is accurate, here, but we can imagine errors such as identifying “to rest” as a noun could yield a selection from the target language array such as the synonymous “спать.” The solution for this issue could be in training the neural network to properly align certain types of phrases and parts of speech with their appropriate counterparts in the output language (Bahdanau et al., 2015).

It is important to be careful, however, to not strictly model the language itself in a prescriptive manner; doing so would significantly reduce not only the efficiency of such programs but the accuracy as well (Goldberg, 2017).

In terms of training neural networks for machine translation, incorporating lexical resources – both in the forms of corpora and databases, although both are used in different ways – is incredibly important. As stated previously, neural networks generally take longer to properly train than statistical models due to neural networks requiring a different type of training than the statistical models (Wu et al., 2016). By providing the corpus-style lexical resource, the neural network is able to learn how words and phrases are used rather than just the probability that they will follow the previous words (Bahdanau et al., 2015). However, the database-table-style lexical resource allows the neural network to classify words according to different tags and classifications (Erjavec et al., 2001). To effectively use such a database in the most efficient manner, the tags that would be used would be those that are most prevalent; of course, to maximize the accuracy of the neural network, another table could be created that would include tags that aren't likely to appear, to avoid an error in classification (Erjavec et al., 2001). Importantly, these tables could be generated by analyzing several corpora on the web; or, as many chatbots do, the tables could be generated by interacting with humans and learning from what they say (Volk et al., 2002). Ultimately, the textual resources for these lexical resources are numerous and have the potential to be incorporated by way of written data stored on the internet. Thus, the internet can serve as an unlimited input source to be harvested and used as teaching models for structure and lexical items. The question remains on how well semantics and pragmatics are encoded from these data.

Another interesting prospect is that of post-editing. However, this is done regularly in professional translations when they are done largely by machine, and it could be done to learn what the neural network needs to improve its accuracy. Taking a

translation produced by a machine and editing it to understand inaccuracy could be integral to increasing these neural machine translators (Allen, 2001). Post-editing could even allow for a solution with polysemy; by analyzing the problems that the neural network may have with multiple connotations or denotations for a single word by correcting the translation, the issues could be corrected and, if able to be replicated, fail-safes could be put in place to avoid such issues from happening again (Boas, 2005).

Generally speaking, the issue of machine translation is not one that can be solved with one simple solution; instead, it would require a combination of methods both new and old, complex and straightforward, and, with these methods together, proper machine translation software may be developed and may be able to bridge the gap between speakers of different languages. There are many to choose from in the realm of lexical resources and many that may be required to efficiently and accurately translate from one language to another. Lexical databases – whether used to classify words according to parts of speech or provide dictionaries and definitions – are important to include as backbones for any machine translation software. The incorporation of corpora to train neural networks on word order and word usage is another crucial aspect of lexical resources, and in order to create neural networks that can continue learning and continue correcting themselves, post-editing must be used so as to learn from the mistakes and correct them until there are no more mistakes. As such, the importance of these lexical resources, post-editing, and the nature of neural networks themselves is their ability to learn from the resources given to them and the world's ability to continue providing these written resources.

Arising from the intersections of linguistics, computer programming, and exploring the remaining problems with NLP programs, the following questions were explored: how can we translate between languages with little-to-no documentation and how effective are CFGs when constructing these translation systems? In the following sections, each question is discussed first through the linguistic lens, and software development, followed by a discussion of next steps and future investigations as informed by this first attempt at a better translation system for low-resource languages.

SECTION TWO: PERTINENT LINGUISTIC FEATURES FOR NLPS

Corpora in NLP and Machine Translation

NLP, in certain contexts, can be viewed as a statistical science; such was the case up until fairly recently, when neural networks became more popularized in professional and research settings. The abstract concept itself is simple: use probability functions to determine the most likely occurrence for the next section of the sentence, translation, and the like (Charniak 1997, p. 33). Using these statistical techniques seems independent of corpora, but this assumption is naive; the nature of statistical analysis relies on corpus analysis to build the actual probability models. Working without these corpora would leave the models uninformed, and similar to how a study cannot be analytically accurate without a large dataset, the same could be said for these statistical models.

In order for computers to understand how to accurately complete processes like part-of-speech tagging, the instructions must be clear and comprehensive. To get the correct answers, the logical steps must be perfect. It's similar to trying to find the probability of a yellow ball being in a set of red balls while being unaware of the exact number of balls of either color, much less the total number of balls; it would be nearly impossible to make an informed decision when trying to calculate the probability, and the same could be said of a computer trying to use a statistical model without the actual data to inform it. Collecting the data is, in a way, the backbone of the statistical model, and this data can be most effectively found in corpora (Charniak 1997, p. 34). Even without

large corpora, the data is still usually being collected from corpora of some kind, whether it be a single sentence or even just a couple of text messages. Written data is, fundamentally, stored in corpora and to do without corpora is to do without an entire branch of statistical data. The importance of such data can be seen even in deep learning contexts as in He, 2019, where the written data informs the sentence alignment algorithm (p. 6).

Furthermore, corpora are an integral part of neural machine translation. Much like more statistical methods of processing natural language, neural networks generally take in certain amounts of parallel data between one language and another in order to learn the patterns in a way that is methodologically similar to that of the human brain's own neural networks (He, 2019, p. 5). However, that is not to say the methods used after extracting the data from the corpora are the same; while statistical methods analyze the likelihood of words in a particular sentence, neural networks use this information to build relational algorithms (He, 2019, p. 7). In theory, neural networks could be created without having access to parallel corpora or even any corpora at all. By analyzing the language(s) by hand, one could calculate the same relational algorithms; this, however, is a much more strenuous task than that of using corpora to train the neural network. In this case, working without corpora would be a major handicap for the technology.

Moving to older, less-accurate technologies, CFGs and part-of-speech parsers also greatly utilize corpora to classify different lexical items into their corresponding "box" in the analysis. The use of corpora in these areas is less straightforward than explained with previous systems: rather than explicitly learning and calculating the statistical probability or relations, the corpora is used to learn how to classify the lexical items into categories

such as determiners, pronouns, verbs, adjectives, and the like. Although these systems rely on more than just training corpora (Denis & Sagot, 2012, p. 723), the fact remains that the corpora themselves are an important part of the systems. To show an example of the importance, consider learning a complex topic in calculus, without having the opportunity to practice it; in theory, it is possible to be able to use the information given to properly solve a differential equation, but it is much less likely than if the opportunity to practice was given. In this scenario, training corpora are synonymous with the opportunity to practice.

The importance of part-of-speech tagging (POS tagging) – and, by extension, corpora – is one that permeates every aspect of NLP (Lv et al., 2016, p. 647). Of course, in this area, the corpora are annotated so that the system can learn the parts of speech for the corresponding lexical items, but they are corpora, nonetheless. In order to emphasize the importance of corpora, the importance of part-of-speech tagging will be explained. As one of the most basic problems of NLP, POS tagging acts as the basis for many analytical practices in many different areas of NLP (Lv et al., 2016, p. 648). The general function is as such: by categorizing the different parts of speech for different words, the higher-level system will be – in theory – more capable of properly completing its task in a more efficient manner. This type of tagging is not restricted to parts of speech; it has also, in recent years, been used to classify types of paraphrasing (Vila et al., 2015, p. 356). Analysis of annotated corpora with the expectation of accurately parsing and categorizing both paraphrases and lexical items is, arguably, the most important aspect in the area for using corpora. Without the corpora – annotated or otherwise – the algorithms will be

forced to work with data given by hand or without any relevant data at all, ultimately jeopardizing the validity of any linguistic analysis it may attempt.

Similarly, even low-resource languages – languages with little-to-no large corpora to reference – rely heavily on either the corpora they have available or the corpora of similar languages. Web scraping, the "scraping" of data from webpages, is often used in these situations (Tahir & Mehmood, 2021, p. 8549). In fact, one of the easiest ways in the modern era to collect and find corpora is to scrape webpages; the data is often easy to access, easy to parse, and tends to offer a much more genuine evaluation of language use than that of traditional texts. Web corpora, even when being used to analyze languages with large numbers of speakers and large numbers of data, can give much insight into the linguistic variations of different dialects (Cook & Brinton, 2017, p. 644). This variant analysis is key to providing accurate translations regardless of dialect; due to the high variance in dialectal lexicons, even omitting one dialect could result in incorrect translations when translating from that dialect to another language, or even another dialect. This could be seen in English. In the Southern dialect, "Coke" is used to refer to many sodas rather than just *Coca Cola*. In other dialects, however, "Coke" means only *Coca Cola*. Without analysis of these individual dialects through their corpora, these distinctions could be lost and could completely mistranslate the semantics, as well as the syntax.

From the view of computational linguistics, corpora are among the most important resources available, and this is evident in the construction of many open-source corpora. *WordNet*, one of these open-source corpora, was the initial resource that prioritized semantics over syntax (Apidianaki & Sagot, 2014, p. 656). Much of the

current research completed on these wordnets revolves around combining models from different languages and allowing the models to expand semantically and lexically, with hopes that the validity of translations grows (Apidianaki & Sagot, 2014, p. 657). Without these corpora, the research in this field would have stalled, and many of these open-source libraries – which current software rely heavily upon – would be nonexistent.

Context-Free Grammars in NLP

In computer science, recursion is a method in which a function calls itself within its definition. For example, if one were to create a recursive program for calculating Fibonacci numbers, the method could call fibonacci(n-1) within itself, thus calling another instance of that method within itself. While recursion can be incredibly useful, it runs the risk of running infinitely, if not written correctly; in order to avoid this, explicit base cases must be written, and each possible text case *must* lead back to at least one of the base cases. Recursion in terms of CFGs is not much different from this definition. In CFGs, recursion involves having one kind of phrase on both sides of the grammar. For example, we could have the grammar in *Fig. 3*.

$$S \rightarrow NP\ VP$$

$$NP \rightarrow DET\ NP \mid PP\ NP$$

$$VP \rightarrow V\ NP$$

Fig. 3

In this grammar, the noun phrase exists on both sides, and the grammar will loop infinitely as it attempts to write each sentence. For each noun phrase it tries to write, it has to write an infinite number more.

For a more formal definition of a CFG, take the following definition $G = \{V_N, V_T, P, S\}$, where V_N is a set of terminals (or lexical items), V_T is a set of nonterminals (abstractions that do not appear in the actual well-formed strings), P is the starting symbol (always one of the nonterminals), and S is a set of rules for the grammar (Hopcroft & Ullman, 1977, pp. 49-50). Any grammar G is used to generate the language L. For example, the following in *Fig. 2* is a context-free language.

$G = \{V_N, V_T, P, S\}$

$V_N = \{slept, saw, walked, the, a, that, this, man, park, dog\}$

$V_T = \{S, NP, VP, Det, N\}$

$P = S$

$S = \{$
$S \longrightarrow NP\ VP$
$NP \longrightarrow Det\ N \mid N$
$VP \longrightarrow$ *'slept'* $\mid$ *'saw'* $NP \mid$ *'walked'*
$Det \longrightarrow$ *'the'* $\mid$ *'that'* $\mid$ *'this'* $\mid$ *'a'*
$N \longrightarrow$ *'man'* $\mid$ *'park'* $\mid$ *'dog'*
$\}$

Fig. 4

Generally, CFGs aren't often used to translate between natural languages, but, for the purposes of this thesis, it is sufficient to investigate the structures necessary to translate to and from a language with little to no written documentation.

SECTION THREE: GRAMMATICAL STRUCTURES OF ENGLISH AND ETHANSKI

For the purposes of defining the languages used in this project, the basic grammar structures will be defined in the following subsections. Due to the limitations of context-free grammars, the structures used and defined are extremely simple, focusing mostly on the importance of clearly and explicitly defining even extraordinarily simple grammars to allow for syntactically accurate translations.

Basic English Structure

English, despite being a Germanic language, diverges from the usual verb-second sentence structure to instead use a subject-verb-object (SVO) sentence structure. The difference can be highlighted in the two sentences "I have a dog" and "neither do I." In the first sentence, the subject "I" is followed by the verb "have," which has "a dog" as the direct object. In the second sentence, the verb occupies the second position, and is preceded by either a single word or a group of words—also referred to as a constituent—while being followed by the rest of the sentence. While English does, in certain contexts (as in the example sentence "neither do I") use verb-second, it does not use it for all sentences. To further show the difference between the two structures, if English used verb-second sentence structure for all sentences, sentences such as "When I get home, I take a nap" would be instead structured as "When I get home, take I a nap."

In terms of questions, English forms interrogative sentences by inverting the subject and the auxiliary or copula; however, in sentences where no auxiliary exists, the auxiliary "do" is inserted while the structure is inverted (Carnie, 2007, p. 264). For

example, take the sentence “you have a dog”; since there is no auxiliary to invert with the subject, the auxiliary “do” must be inserted, creating the question “do you have a dog?”

English—and most (if not all) languages—cannot fully be represented in a CFG due to not being context-free, though many broad HPSG grammars have been written for the language. Despite this limitation, for the purposes of this thesis, the relative simplicity granted by CFGs is useful in analyzing the ability of such systems to translate syntactic structures without much worry for semantics.

Basic Ethanski Structure

Ethanski, the artificially constructed language (conlang) created for this thesis, is a language based on Scandinavian languages, although it does possess question particles much like Mandarin. I designed Ethanski to contain very little lexical or structural data, for the purposes of this thesis, namely because it is being used to imitate a low-resource language; note that the reason a conlang was used instead of an actual low-resource language was due to lack of time and resources.

Like English, Ethanski’s word order is SVO, or subject/verb/object. This is, of course, relative to any conjunctions, numbers, or articles also used in the sentence. The exception is with interrogative sentences; questions are VSO, or verb/subject/object, similar to other Germanic languages.

Due to this basic similarity, translating simple sentences, i.e., sentences with single verb phrases, between the two languages occurs without trouble; for example, to say “I like soccer,” the program would be able to translate each word individually—along with conjugations and the like—and form the sentence. In this case, we know that “I” is “Cjá.” The verb “to like” in Ethanski is “sá séniz,” or “sénizca” after conjugation. The

object, soccer, is “fútvo.” Thus, the translated sentence is “Cjá sénizca fútvo.” Specific word order, however, may change depending on the type of clause.

One difference between the two languages is due to two characteristics of Ethanski: verb conjugation and the usage of a particle to denote interrogatives. The conjugation table can be seen in *Fig. 3*.

I -ca	You (formal) -jyt
You (informal) -rja	They (plural) -in
He/She/It/They (singular) -an	We -sjat

Fig. 5

These conjugations bring inherent difficulty in using a context-free grammar to translate, requiring the program to allocate storage in order to keep track of the respective subject’s pronoun. That is not to say, however, that such a thing would be impossible; in computer science—since recursion is usually computationally expensive—dynamic programming, or the technique in which a problem is broken into subproblems whose answers are saved, is often used to store data for later recursions, reducing the need to calculate results repeatedly. The referent and its relevant pronoun may be stored together and thus used in additional machinations in order to construct the translation.

In terms of interrogatives, the particle *ní* is placed at the sentence end in Ethanski to form an interrogative. While this could make translations from Ethanski to English straightforward when it comes to questions, English poses interesting issues due to its general question structure. Do-support is difficult to parse with only CFGs; for the sentence “I have a cat,” the question form would be “Do I have a cat?,” whose structure is difficult to put into CFGs when combined with other structures.

Additionally, not every question has an interrogative pronoun:

For example,

"**What** is today?"

Where the interrogative pronoun is bolded, and:

"**Do** you have a cat?"

Where the "do" is added instead of an interrogative pronoun.

To avoid these problems in this thesis, a distinction was made between the interrogative and non-interrogative "do" forms as well as differences between "do" and interrogative pronouns."

Designing Context-Free Grammars

To create the translator, the context-free grammars had to be created for each language. The grammars were written in a text file to be accessed during the parsing process; they can be found in Appendix 1.

Each language has one comprehensive CFG, with the starting symbol being defined as multiple sentence structures. Due to the large lexicon, the CFGs do not include all possible words; instead, there is a select set of words are stored in its dictionary. For translation, the keys for English (words) and their corresponding values in Ethanski (words) are listed with additional values for their corresponding parts of speech. For example, the following table shows some of the words available for translation and their parts of speech.

English	Ethanski	Part of Speech and Simple Definition
Bear	**Eca**	n. mammal native to America and Eurasia
Bear	**Sa od**	v. to endure
Sleep	**Sa gunzo**	v. to be in a state of sleep
Sleep	**Jidjix**	n. a state in which the mind and body rests

Table 1

Assembling the Translator

The translator was assembled in a Jupyter notebook, a Python IDE (or development platform) that allows for interactive processing where code can be broken

up into individual parts, allowing for separate processing according to the separate cells the code was written in. An input widget was created to allow dynamic input by users, although it should be noted that due to the computational limitations of this project, full lexicons were not used; the goal was not to create a full translator, but to create a translator that can give us insight into how we might translate between a high-resource language and a low-resource language.

With the given input in the given language, the input was split according to delimiters (usually spaces) then stored in an array. Once in the array, the words are iterated in order to determine the type of sentence of the given input. This focused heavily on word order and the relationship between constituents. The importance of this step was mainly for questions; the differing syntax between the two languages when it comes to questions requires this kind of parsing to allow proper translations. In order to translate questions a boolean variable was defined within the program and labeled as *interrogative*. The interrogative array consisted of the parser identifying such question indicators as *Wh-word* or question marks. The interrogative variable, essentially, flagged whether or not the given sentence was a question or not based on the array that defined interrogatives. This determined which starting terminal definition would be used.

Once that was determined, the next step was to translate each individual word into the target language, storing it in another array. For example, pronouns—since the target language was Ethanski—require morphological suffixes that agree with the verb, i.e., verb agreement. Therefore, the morphological suffix patterns were copied into the array and saved in a storage variable, allowing for proper verb conjugation in the translation from English to Ethanski.

Thus, if a sentence in English were being translated to Ethanski, a sentence or questions would be entered for the CFG to “read” and “translate”. When this process is initiated, the translator starts with analyzing each cue of the language sample (i.e., a sentence or a question). The terminal definition—or decoder—uses the array and subsequently selects the target language words (Ethanski), orders the words according to the proper syntax programmed in, and conjugates the verbs and pronouns, need be. Once the sentence is completely parsed from English to Ethnanski, the translation is outputted. The CFG is, likewise, able to output Ethnanski sample sentences or questions in English form using the same processes in tandem with the language’s terminal definitions and array of features.

SECTION FIVE: DISCUSSION AND FURTHER RESEARCH

When examining the output, as expected, the semantic acuity, i.e., the meaning-making ability, of the translator was significantly lacking. Much like the grammatical validity of Chomsky's sentence, "Curious green ideas sleep furiously" (1957, p. 116), the translator created structurally accurate sentences in both English and Ethnanski; however, these products were also often nonsensical. Seeing as semantically felicitous sentences were not the focus of this project, this was not troublesome. In the future—with more time—the ideal would be to create a knowledge graph database in order to create weighted relationships between words and their translations, giving heavier weights to the more accurate translations, but also allowing for a gradient according to context. However, with CFGs, the context aspect obviously is not as important; hence the name "context-free" grammars. The translator, though, did easily parse sentences like the one in *Fig 6*.

Fig. 6

In terms of syntax, though, the translator functioned much better than expected. As stated in the section on grammars and context-free grammars, it can be hard to definitively write out a grammar in a parseable fashion that does not lead to infinite

recursion. Careful writing and the lack of sentence generation easily avoided this problem; however, the limitation of CFGs required an avoidance of more complex structures (i.e., multiple clauses [more than one verb phrase] in a sentence such as *The dog **gave** the cat a mimosa **to calm** him down after the owners **left** for the Bahamas*. In future, perhaps if complex grammar structure is pursued, it would be best to program in grammar structures such as head-driven phrase structure grammar or lexical-functional grammar structure, which would allow for more specific and complex grammar structures to be accurately translated. This problem can be seen in *Fig. 7*, where the expected translation would be *Dakki kero vy jaxta sa yr i cro ryx*, as opposed to the given translation, which has the verb "run" conjugated for the mouse. The issue can more than likely be attributed to the inability of context-free grammars to accurately parse verbs in multiple contexts.

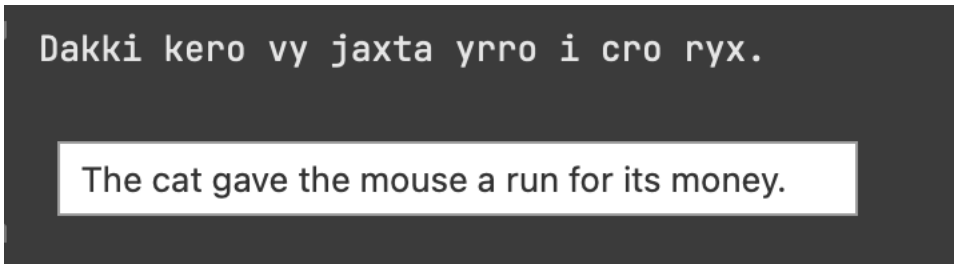

Fig.7

Another issue found in the translator was its inability to properly handle "do"-support, despite my attempts to classify the dummy article (Carnie, 2007, p.264)—the inserted "do" auxiliary in English question formation (You like ice cream. [declarative] > Do you like ice cream? [interrogative/question]as functionally and denotationally different from the verb. A solution, if still using CFGs, would probably be to do a cursory parse of the given sentence, that is having the parser identify the "do" before it moves to

a word-by-word examination. Thus, before splitting up the sentence and storing it in the array, the parser would identify the “do” within the overall structure of the question and determine the placement of the dummy article as key to question formation, thereby giving the translator the opportunity to see it as different from the lexical verb (e.g., “like” in the sentence example). This can be seen in *Fig. 8*, where the expected translation would be *Arja zto dakki ni?*, rather than the given translation. The translator incorrectly sees the word “do” as an unconjugated verb instead of a word that does not need to be translated because it has no lexical meaning in Ethanski’s question syntax.

Fig. 8

Overall, this project granted me valuable insight as to how difficult it is to translate between languages when one has very little data. Using these data and this experience, I can further pursue research in this area of natural language processing and machine translation, working to provide faithful translations, beginning with proper syntax (the purpose of this thesis) and, in future, focusing more heavily on semantically faithful translations.

References

Apidianaki, M., & Sagot, B. (2014). Data-driven synset induction and disambiguation for wordnet development. *Language Resources and Evaluation*, *48*(4), 655–677. https://doi.org/10.1007/s10579-014-9291-2

Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Brown, P. F., Cocke, J., Pietra, S. A. D., Pietra, V. J. D., Jelinek, F., Lafferty, J. D., Mercer, R. L., & Roossin, P. S. (1990). A statistical approach to machine translation. *16*(2), 7. https://aclanthology.org/J90-2002.pdf

Carnie, A. (2007). *Syntax: A generative introduction*. Blackwell.

Charniak, E. (1997). Statistical techniques for natural language parsing. *The AI Magazine*, *18*(4), 33–43.

Chomsky, N. (1956). Three models for the description of language. *IRE Transactions on information theory, 2*(3), 113-124.

Cook, P., & Brinton, L. J. (2017). Building and evaluating web corpora representing national varieties of English. *Language Resources and Evaluation*, *51*(3), 643–662. https://doi.org/10.1007/s10579-016-9378-z

Dassow, & Masopust, T. (2012). On restricted context-free grammars. *Journal of Computer and System Sciences*, *78*(1), 293–304. https://doi.org/10.1016/j.jcss.2011.05.008

Deemter, K. van. (2016). *Computational Models of Referring: A Study in Cognitive Science*. The MIT Press.

Denis, P., & Sagot, B. (2012). Coupling an annotated corpus and a lexicon for state-of-the-art POS tagging. *Language Resources and Evaluation*, *46*(4), 721–736. https://doi.org/10.1007/s10579-012-9193-0

Erjavec, T., Evans, R., Ide, N., & Kilgarriff, A. (2001). The concede model for lexical databases. http://www.lrec-conf.org/proceedings/lrec2000/pdf/335.pdf

Goldberg, Y. (2017). *Neural network methods in natural language processing*. Morgan & Claypool Publishers.

Grosz, B. J. (2018). Smart Enough to Talk with Us? Foundations and Challenges for Dialogue Capable AI Systems. *Computational Linguistics*, *44*(1), 1–15. https://doi-org.libsrv.wku.edu/10.1162/COLI_a_00313

Guarino, N. (1998). Some ontological principles for designing upper level lexical resources. https://arxiv.org/abs/cmp-lg/9809002v1

He, H. (2019). The parallel corpus for information extraction based on natural language processing and machine translation. *Expert Systems*, *36*(5). https://doi.org/10.1111/exsy.12349

Hopcroft, J. E., & Ullman, J. D. (1969). *Formal languages and their relation to automata*. Addison-Wesley Publishing Company.

Kalra, V., Kashyap, I., & Kaur, H. (2022). Generation of domain-specific vocabulary set and classification of documents: Weight-inclusion approach. *International Journal of Information Technology, 14*(1), 275–285. https://doi.org/10.1007/s41870-021-00830-8

Kaplan, R. M. (1995). The formal architecture of lexical-functional grammar. *Formal Issues in Lexical-functional Grammar*, *47*, 7-27.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. https://homepages.inf.ed.ac.uk/pkoehn/publications/europarl-mtsummit05.pdf

Lehnert, W. G. (1997). Information extraction: What have we learned? *Discourse Processes*, *23*(3), 441–470. https://doi-org.libsrv.wku.edu/10.1080/01638539709544999

Lishchynska, D. (2022, April 5). *What are bots? How do chatbots work?* BotsCrew. Retrieved April 14, 2022, from https://botscrew.com/blog/what-are-bots/

Lv, C., Liu, H., Dong, Y., & Chen, Y. (2016). Corpus based part-of-speech tagging. *International Journal of Speech Technology*, *19*(3), 647–654. https://doi.org/10.1007/s10772-016-9356-2

Müller, S. (2020). *Grammatical theory: From transformational grammar to constraint-based approaches*. Language Science Press.

Müller, S., Abeillé, A., Borsley, R. D., & Koenig, J. P. (Eds.). (2021). *Head-driven phrase structure grammar: The handbook*, (Vol. 9). Language Science Press.

National Institute on Deafness and Other Communication Disorders. (2018). *Speech and Language Developmental Milestones*. NIDCD. https://www.nidcd.nih.gov/health/speech-and-language

Neff, M. S., & Mccord, M. C. (1990). Acquiring lexical data from machine-readable dictionary resources for machine translation. *In Third International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages (TMI-90)*, 85–90. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.132.8355&rep=rep1&type=pdf

Russell, S. W. (2016). *Computer interpretation of metaphoric phrases*. Walter de Gruyter GmbH & Co KG.

Tahir, B., & Mehmood, M. A. (2021). Corpulyzer: A novel framework for building low resource language corpora. *IEEE Access*, *9*, 8546-8563. https://doi.org/10.1109/ACCESS.2021.3049793

Valitutti, A., Strapparava, C., & Stock, O. (2004). Developing affective lexical resources. *PsychNology Journal*, *2*, 61–83. https://wndomains.fbk.eu/publications/psychno.pdf

Vila, M., Rodriguez, H., & Martí, M. (2015). Relational paraphrase acquisition from Wikipedia: The WRPA method and corpus. *Natural Language Engineering*, *21*(3), 355-389. doi:10.1017/S1351324913000235

Volk, M., Pajusalu, R., & Hennoste, T. (2002). Using the web as corpus for linguistic research. *Tähendusepüüdja. Catcher of the Meaning. A Festschrift for Professor Haldur Õim*. Publications of the Department of General Linguistics 3. University of Tartu.

Wallis, P. (2011). From data to design. *Applied Artificial Intelligence*, *25*(6), 530–548. https://doi-org.libsrv.wku.edu/10.1080/08839514.2011.587155

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, Ł., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., … Dean, J. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv:1609.08144 [Cs]*. http://arxiv.org/abs/1609.08144

Zemcik, T. (2019). A brief history of chatbots. *International Conference on Artificial Intelligence, Control and Automation Engineering.*

APPENDIX: CONTEXT-FREE GRAMMARS FOR ENGLISH AND ETHANSKI

Basic English Context-Free Grammar (noninterrogative)

S → NP VP

NP→ (D)(AdjP+)N(PP)

VP → V NP

PP→ P NP

Basic English Context-Free Grammar (interrogative)

S → Aux NP VP | Wh-NP VP | Wh-NP Aux NP VP

NP→ (D)(AdjP+)N(PP)

VP → V NP

PP→ P NP

Basic Ethanski Context-Free Grammar (noninterrogative)

S → NP VP

NP→ (D)N(AdjP+)(PP)

VP → V NP

PP→ P NP

Basic Ethanski Context-Free Grammar (interrogative)

S → VP NP *ni* | Interrogative VP NP *ni*

NP→ (D)N(AdjP+)(PP)

VP → V NP

PP→ P NP