The Thesis committee for Juan Daniel Pinto
Certifies that this is the approved version of the following thesis:

Creating a Conversational Hebrew Vocabulary List

APPROVED BY

SUPERVISING COMMITTEE:

Esther Raizen, Supervisor

Elaine Horwitz, Co-Supervisor

Creating a Conversational Hebrew Vocabulary List

by

Juan D. Pinto

Thesis

Presented to the Faculty of the Graduate School
of the University of Texas at Austin
in Partial Fulfillment
of the Requirements
for the Degree of

Master of Arts in Hebrew linguistics

The University of Texas at Austin
May 2018

Dedication

Dedicated to

Acknowledgements

Interdum et malesuada fames ac ante ipsum primis in faucibus. Aliquam congue fermentum ante, semper porta nisl consectetur ut. Duis ornare sit amet dui ac faucibus. Phasellus ullamcorper leo vitae arcu ultricies cursus. Duis tristique lacus eget metus bibendum, at dapibus ante malesuada. In dictum nulla nec porta varius. Fusce et elit eget sapien fringilla maximus in sit amet dui.

Mauris eget blandit nisi, faucibus imperdiet odio. Suspendisse blandit dolor sed tellus venenatis, venenatis fringilla turpis pretium. Donec pharetra arcu vitae euismod tincidunt. Morbi ut turpis volutpat, ultrices felis non, finibus justo. Proin convallis accumsan sem ac vulputate. Sed rhoncus ipsum eu urna placerat, sed rhoncus erat facilisis. Praesent vitae vestibulum dui. Proin interdum tellus ac velit varius, sed finibus turpis placerat.

Creating a Conversational Hebrew Vocabulary List

by

Juan Daniel Pinto
The University of Texas at Austin, 2018
SUPERVISORS: Esther Raizen, Elaine Horwitz

Indent and begin abstract here. It should be a concise statement of the nature and content of the ETD. The text must be either double-spaced or 1.5spaced. Abstracts should be limited to 350 words.

Table of Contents

List of tables

List of figures

1 Introduction

This thesis explains the theory behind the creation of the Conversational Hebrew Vocabulary List (hereafter CHVL)—a Modern Hebrew high-frequency word list—along with implications from the project. The list itself is included as an appendix, and can also be downloaded free of charge in a variety of formats. While evaluating past methods for the creation of such lists, it became clear that a large gap in the literature exists when it comes to less commonly taught languages (LCTLs). Because the overwhelming majority of the previous research in this area has been focused on English alone, some important nuances are yet to be addressed. More often than not, the few non-English word lists that do exist, along with much of the research in vocabulary acquisition, have taken at face value some of the findings of this Anglo-centric research, often without questioning whether the same methodologies and conclusions apply to different languages.

The present paper is, therefore, an effort to help educators interested in creating and/or using word lists for their own classrooms, for wider dissemination, or simply for general research purposes. In doing so, it will simultaneously provide an overview of some of the key decisions that must be taken into account for such a project, along with key studies on the topic.

The various uses of word frequency lists can be roughly classified into research applications and practical applications. Examples of research applications include traditional linguistic studies that look for common morphological patterns, corpus-linguistic studies seeking to understand language through "real world" texts, and psycholinguistic studies that explore connections between a speaker's mental lexicon and word frequency. Practical applications of word lists include curriculum and textbook planning for language teachers, vocabulary selection for graded readers and dictionaries, and even independent language study. Of course, the line between research and practical applications can be rather fuzzy. Some of the most important studies lie between these two groups, and attempt to answer questions such as: How can vocabulary knowledge be appropriately tested and measured? What is the role of extensive reading (as opposed to intensive reading) in incidental vocabulary acquisition? What level of vocabulary do learners need in order to read extensively

for pleasure? What level of vocabulary do learners need in order to succeed in an academic setting? What role does specialized vocabulary play in reaching understanding? These questions and their answers rely heavily on the creation and use of trustworthy word frequency lists. Yet due to the resources and effort required to create these lists, they are rarely found in languages other than English.

The theoretical foundation for the creation and use of word frequency lists rests on the observation, made popular by the linguist George Kingsley Zipf in the 1930s and 40s, that if one were to create a frequency list of words in a large enough text, the first word would occur roughly twice as often as the second word, three times as often as the third word, and so on (Zipf 1935, 1949). This exponential distribution is significant because it means that a small number of words make up the bulk of a text, whereas the majority of the words occur very few times. Paul Nation, one of the most influential scholars in the field of vocabulary acquisition, has pointed out that Zipf's Law—as it is has come to be known—can serve as motivation to language learners and teachers, since learning the most common vocabulary in a language covers so much of the communication that naturally occurs (2001).

The primary research question guiding this project is this: *What are the most-frequently used words in conversational Modern Hebrew?* The resulting study also addresses the following secondary research questions, which were necessary to address in order to answer the aforementioned question: *What effect does a corpus of unvocalized texts have on the identification of word families in the computerized creation of a vocabulary frequency list? What factors affect the way that boundaries are demarcated for various levels of word families in Modern Hebrew?* And finally: *What implications might these findings have for word list creation and use as it pertains to other less commonly taught languages?*

The literature review will serve as a basis for many of the important decisions taken during the creation of the CHVL. These decisions—surrounding both corpus and list creation—along with their reasoning, will be explained further in an analysis of the literature. For the sake of clarity, these decisions are listed here at the outset. They are as follows:

Corpus design - *Size:* - *Text types:* The corpus consists of a single text type: conversation. This is to best fit with the list's intended audience. In order to accomplish

this, movie and television subtitles compose the core of the corpus. **List design:** - *Use:* The primary intended audience for the CHVL is composed of beginning-to-low-intermediate learners of Hebrew as a foreign language. It is designed for both receptive and productive language use. - *Word family levels:* The word family level that is best suited for the CHVL's intended audience is the lemma. - *Criteria:* The CHVL was created using exclusively objective criteria, meaning that it is the product of calculations, and it was not manually tweaked in any way. The empirical criteria used were frequency and range.

Following the review of literature and explanation of theory, the process of the CHVL's creation will be explained in detail, along with findings from the project. Possible implications for other less commonly taught languages will then be discussed. Finally, the full CHVL will be provided as an appendix.

2 Review of the literature

2.1 Corpus Design

For a word list to accurately reflect the use of a language in its broadest context, the corpus from which it is extracted needs to be representative of that context. Since it is impossible to analyze all of the communications that take place in a particular language (not even taking into account the fact that language itself is an ever-expanding, ever-changing, *open* corpus), researchers must make do with an approximation of the whole: a bounded corpus of language.

Though the focus of this literature review is the creation of word frequency lists, the truth is that relatively few corpora have been created for this specific purpose. Most corpora have aimed at being general collections that cover the language (usually English) as a whole in an attempt to serve different theoretical and applied uses. Yet despite this broad range of purposes, the creation of corpora has historically revolved around two big questions: (1) how large should the corpus be, and (2) what kinds of texts should it include? I will here address these two points separately, with the recurring emphasis remaining on corpus use for word list creation.

2.1.1 Corpus Size

Conventional wisdom in corpus creation states that more is better. If a word list is to accurately reflect the frequencies of words in the language as a whole, then a corpus must contain enough text to approximate the overall use of discourse. This line of thinking is equivalent to the maxim in quantitative research that a sample should be as representative of the target population as possible. And in order to maximize the statistical probability of this representativeness, the sample must be of an appropriate size for the study. True, larger sample sizes often increase this probability, but they also tend to be more resource-intensive for the researcher. The same is true of corpus size. When creating a vocabulary list, then, what is an "appropriate" corpus size?

Corpora composed of millions of tokens are easy to access today. This is especially true of corpora of written material—corpora of spoken language are still compara-

tively small. Advances in computing power have made it possible to analyze these mega-corpora, something that would have been far too labor-intensive in the not-so-distant past. It is finally becoming plausible for more researchers without access to extensive resources to use these mega-corpora for the purpose of word list creation.

The first project to create a one-million-token corpus was Kučera and Francis' effort at Brown University to compile a corpus of American English texts printed in 1961. They strived to create a corpus with equal amounts of texts from different sources by randomly selecting 500 passages of 2,000 words each from different published materials found at the Brown University Library and the Providence Athenaeum. This mixed design would be used as a model by many of the corpora created during the next few decades: . These began to be compiled at increasingly faster rates. Many of these corpora were created—in part—to serve as parallel corpora of different varieties of English.

What began in 1980 as a collaboration between Collins Publishing and a group of researchers—the *Collins Birmingham University International Language Database* (COBUILD)—became a 7-million-token corpus by 1982. It continued expanding until it was joined to *The Bank of English* corpus in the 1990s, which reached 320 million words in 1997. In 2004, as part of the Collins World Web, it reached 2.5 billion words (HarperCollins Publishers, 2004a, 2004b). Now, with the use of web-crawling applications that scour the internet and collect text at unprecedented speed, there exist English corpora 11 billion tokens (*enTenTen12*) and even 19 billion tokens (*enTenTen13*).

Clearly, then, the sky's the limit when it comes to ever-growing corpora of language. But when it comes to word list creation, is there a corpus size that can be considered sufficient?

Studies have approached this specific problem of corpus size for word list creation by creating multiple frequency lists—each from a different size of corpus—and then comparing the efficacy of these lists themselves. But what makes a word list effective? Different researchers have tackled this question differently.

One way to judge the effectiveness of a word list is to find how closely it correlates with reaction times in a lexical decision task—a widely-used procedure in psycho-

logical and psycholinguistic research. In a lexical decision task, participants are presented with a series of words and non-words, one after the other, and they are asked to judge which is which as quickly as possible. The reaction times are then analyzed for each word. The basic idea behind this experiment is that the average time it takes participants to react to a word reflects something about the way the word is accessed in participants' mental lexicons. Given enough data, it is possible to make certain inferences about the arrangement of this internal lexicon, which has led to various psycholinguistic theories over the years. But what does this have to do with words on a frequency list? There is well-attested evidence to suggest that there exists an inverse correlation between word frequency and reaction time on a lexical decision task (Whitney, 1998; Balota and Chumbley, 1984). In other words, more common words are accessed and recognized more quickly than less common words. Therefore—the thinking goes—an effective word frequency list should correspond to and reflect this reality.

This was precisely the approach taken by Brysbaert & New (2009), who compared respond times collected as part of the massive Elexicon Project (Balota, et al., 2007) to words on a series of frequency lists made from increasingly larger corpora. The corpora used were all subcorpora extracted from the British National Corpus (BNC). With each subsequent increase in token count, the word list correlated more and more closely with the response times from lexical decision tasks. This observation validates the line of thinking described at the beginning of this section regarding the need for large corpora. Brysbaert and New hoped to find an "ideal" corpus size after which the increase in effectiveness would no longer be significant enough to justify the additional cost of resources. After conducting several regression analyses on the two sets of data, they found that the variance in the response times that could be accounted for by corpus size reached a plateau at about 16 million words. In other words, for corpora with less than 16 million words, the size of the corpus had a significant effect on the correlation between word frequencies and average response times for those words on lexical decision tasks. For corpora with more than 16 million words, the effect of increasing corpus size became considerably more subtle. In the end, they concluded that in order to construct an effective word list for *high-frequency* words, a corpus of about 1 million tokens is needed. However, in order to reach the same effectiveness for *low-frequency* words, a corpus size of at least 16 million words

is preferable.

A different, more straightforward methodology is to directly compare word lists made from corpora of different sizes. Rather than judging the "effectiveness" of a list, this approach measures similarities shared between different lists. Hypothetically, doing this at increasing corpus sizes should allow one to find a size after which the variance between lists only minimally decreases. As with the previous approach, the goal here is to find a point at which the benefits of increasing size no longer outweigh the additional needed resources.

Essentially, then, all corpora of sufficient size should result in nearly the same word frequency list—a theory based on a strict interpretation of Zipf's law applied to all natural language. If the appropriate criteria can be found—Sorell (2013) suggests—then this would, at last, provide a solution to Nation's (2001, 2013) observation that, problematically, word lists tend to disagree rather drastically on both the words included and their respective ranking.

Inspired by the computational linguistic measure of *rank distance* (Popescu and Dinu, 2008)—a method for comparing stylistic differences between texts—Sorell (2013) developed a variant of this methodology. First, he used different corpora of the same size to create multiple word lists, one for each corpus, ranked entirely by frequency. He then identified the percentage of words that are *not* shared between each set of two lists. Finally, he averaged these percentages to find the level of variability created at that specific corpus size. The levels of variability he found were remarkably close to each other—despite using a wide variety of entirely different corpora (with no overlap on texts within each one). He then increased the size of each corpus and repeated the process.

In order to calculate this level of variability, Sorell used a modified version of a complex formula that he borrowed from the natural sciences, and called his resulting calculation the *Dice distance*. Though this Sørensen–Dice coefficient that he altered (also known by other names) is widely used in botany and other fields to measure similarity in areas and samples of different sizes, the frequency lists measured by Sorell were all purposefully of the same size. What this means is that—apparently without realizing it—his *Dice distance* was ultimately just a simple percentage: *number of different words between frequency lists / size of frequency list (total words).*

Regardless of the round-about way he used to calculate it, his resulting measure for each corpus size—the level of variability—can be accurately described as the average proportion of difference for word lists at that particular corpus size.

Sorell found that a stable list (about 2% variation) of the most frequent 1,000 words, or a reasonably stable list (less than 5% variation) of the most frequent 3,000, words can be created using a corpus of 50 million tokens. In other words, 1,000-type word lists created from different 50-million-token corpora will likely only differ by 20 words. At the 3,000-type level using the same sizes of corpora, the lists will likely vary by less than 150 words. This is a remarkable level of similarity. Expanding the list to 9,000 types will still only have about 4–7% variation, or 360–630 words. Even corpora of 20 million tokens can be considered sufficient in many cases, since they will result in 3,000-type word list with roughly 5% variation, and 9,000-type word list with less than 10% variation.

Taking a similar approach, though with significant variations, Brezina and Gablasova (2015) compared four corpora of various sizes: The Lancaster-Oslo-Bergen Corpus (LOB), The BE06 Corpus of British English (BE06), The British National Corpus (BNC), and EnTenTen16. These corpora had respective token sizes of 1 million, 1 million, 100 million, and 12 billion. The word list created from each corpus was, in this case, a combination of frequency and dispersion—a measure that will be discussed in more detail later in this paper. The resulting word lists were then compared, and the percentage of shared vocabulary words calculated. Additionally, the researchers also calculated the correlation between the ranking for each word that was shared between word lists. Contrary to Sorell, Brezina and Gablasova considered this final comparison an important part of understanding the effect of corpus size.

The aim of this study was not to find a corpus size after which the difference was negligible, but rather to find if there was a significant difference between word lists made from corpora of different sizes. The study found a 78%–84% overlap between each of the 3,000–lemma word lists. 71% of the words were shared among all four of the lists. Based on this number, Brezina and Gablasova concluded that regardless of corpus size—at least for anything larger than one million tokens—"similar results" are obtained.

This conclusion differs significantly from Sorell's, who concluded that a corpus of at

least 20 million tokens (though 50 million is preferable) is needed for a stable word list with low variability. These disagreements are primarily the result of a difference in what should be considered "stable." At 71% vocabulary overlap—which is sufficient for Brezina and Gablasova—870 words were only found in one of the four lists. This is drastically higher than Sorell's threshold, which at the 3,000-word level varies in roughly 150 words. Note that Nation and Hwang (1995) found a level of overlap similar to Brezina and Gablasova when comparing the GSL, the LOB, and the Brown corpora—a percentage of overlap that they deemed to be not particularly high. As Nation later put it, "Brezina and Gablasova are a bit too tolerant in accepting that 71% or even 78%-84% overlap is good enough. If roughly one out of every four or five words is different from one list to another, that is a lot of difference" (2016, p. 100).

Another difference to mention between these two studies is the unit of counting used. Sorell made lists based on *types*, whereas Brezina and Gablasova preferred the use of *lemmas*. I will explain this important distinction in a later section of this review ("Identifying Words"). For now, it is sufficient to say that the effect of these different measures in comparing word lists created from corpora of different sizes has (to my knowledge) not been studies. This is one area that could benefit from further research.

Lastly, the corpora used by Brezina and Gablasova were all-inclusive: each built on its own philosophy on the way that different types of texts should be balanced in a corpus, but all seeking to be representative of English as a whole. This is also true of the corpora used by Brysbaert and New in their study using response times from a lexical decision task. Contrast this with Sorell's word lists, which were systematically created from corpora that consisted of only one specific text type. Surely, this is a factor to consider in corpus design.

Therefore, having a sufficiently large corpus is important, as demonstrated in this section. But is it enough? How much do the types of texts included in a corpus factor into its effectiveness for word list creation?

2.1.2 Text Types

There's been a lot of debate about the "best" way to balance a corpus' text types. This is a major aspect of corpus design, and one worth delving into. At the end of the day, much of it comes down to the purpose of the corpus. When used for the creation of word lists, one must also consider the intended purpose of the word list itself. Is it for general use or for one of many possible specialized uses? More on this in the next section.

In order to design a corpus with different amounts of text types (i.e. narrative, conversational, academic), clear definitions for these text types are necessary. But is there a better way than the use of subjective genres to classify texts?

Or is there a better methodology than simply mixing a bunch of different texts together, with the hope that the resulting word list covers the language as a whole? This is the most common way of creating frequency lists, but it tends to result in a mix of words that have little relevance to any one purpose. Esoteric, academic words in a beginners' vocabulary list? Science fiction terms in a vocabulary list for business managers? It's obvious that a list is only as good as the corpus from which it's made, which is why a clear delineation of different text types and their qualities is critical.

When speaking of corpus balance, I refer to the proportion of different text types that make up a corpus. Published corpora have taken different approaches in this regard, and published word lists have made use of a variety of strategies for balancing the corpora from which they are made. Coxhead's *Academic Word List* (2000) was created from a carefully-designed corpus that used equally-sized sub-corpora of texts from different disciplines. This suited the purpose of her word list well, since it was intended to serve students from a variety of disciplines.

The importance of identifying a taxonomy of text types based on objective criteria: are there distinguishable linguistic differences between an informal correspondence and a narrative work of fiction? What about between a romance and a fantasy novel?

Biber's early work (1988) conducted an analysis of a wide variety of texts using large corpora to tag syntactic markers and other linguistic attributes that could

potentially be used to define different types of texts. In this study, he found a series of five categories (each consisting of two opposite ends of a spectrum) in which texts varied: involved vs. informational, narrative, situated vs. elaborated, persuasive, and abstract. He then conducted a very large study, which he published as a book, (1995) that found eight distinct, recurring patterns of different combinations of these categories. These groupings serve as a linguistically-based taxonomy that divides texts along objective lines, rather than subjective, culturally-defined genres.

Similar but independent studies were conducted for Somali, Korean, Nukulaelae Tuvuluan, Taiwanese, and Spanish (Biber, 1995; Jang, 1998). For each language, a unique set of text types were identified. However, the texts were found to align along similar distinguishing linguistic dimensions as the English texts.

Sorell (2013) sought to simplify Biber's eight text types into categories suitable for corpora study. He did this by noticing the closely similar ways that some of the text types lined up along Biber's five linguistic categories, also incorporating some extra-linguistic features, such as shared contexts (e.g. predominantly spoken types). He also dropped Biber's two smallest text types, deeming them impractical for corpus study and difficult to isolate. In doing this, he came up with four simplified text types: interactive (conversation), general reported exposition (general writing), imaginative narrative (narrative writing), and academic. Regarding this last type, Biber's study found a sosignificant difference between academic writing in the natural sciences ("scientific exposition") and the humanities ("learned exposition")—he found that natural science uses more concrete language, whereas the humanities tend to use more abstract language. However, Sorell sought to unify these for the sake of simplicity, simply leaving their distinction to "a future study" (p. 68). Sorell acknowledged that his wasn't the first attempt at simplification of Biber's text types, a surprisingly similar effort having been made in the *Longman Grammar of Spoken and Written English* (Biber, Johansson, Leech, Conrad, & Finegan, 1999: 16) and the *Longman Student Grammar of Spoken and Written English* (Biber, Conrad, & Leech, 2002: 23).

Sorell found that each of his four simplified text types yielded a vocabulary frequency list that was as unique as the linguistic criteria that Biber had used. He also measured how different they were from each other, and found all four to be equidistant from

the next in this order: conversation, narrative, general writing, and academic writing (See section on corpus size for an explanation of this measurement). Sorell, therefore, claims that his own study of vocabulary frequency using his simplified text types as a base has "validated Biber's studies by adding a vocabulary dimension to the description of each of the key text types" (201).

Despite the importance of spoken language—or the conversation text type—for language learners and linguistic studies, the number of conversation corpora that exist, as well as their size, is very limited. This is clearly because of the difficulty of gathering large amounts of spoken data that then needs to be transcribed by hand in order to be analyzed. It is true that speech recognition software has come a long way in recent years, but its rate of error remains too high for research purposes. It has been estimated that it takes 40 hours to professionally transcribe one hour of audio recording, making the task too costly. For this reason, some researchers have begun looking at alternative sources for a conversation corpus, including the internet and movie subtitles.

New, et al. (2007) created a 50-million-token corpus of French subtitles. They divided this into four subcorpora, one for each of the type of media from which the subtitles were extracted: French films, English movies, English television series, and non-English-language European films. The reason for using French subtitles from English media is the sheer dominance of English in the film industry. In order to counter-balance the much larger sizes of the two subcorpora extracted from English media, the researchers measured word frequencies for each subcorpora separately, then averaged them to arrive at the final frequency used for their ranked word list.

In order to test the validity of their new approach, New, et al. used two different methods. First, they compared their subtitle word list with word lists created from more traditional corpora. Second, they used lexical decision times—similar to Brysbaert and New (2009) above—to test the rankings of words on their list.

The first test found a .73 correlation with a classical French spoken corpus, the "Corpus de Référence du Français Parlé" (CRFP; Equipe DELIC, 2004). However, when looking at the specific words and semantic categories that differ the most, it's clear that most major differences are caused by the monologue-nature of the CRFP. This corpus was created from a large number of interviews (each asking the

same questions to the interviewee), whereas movie subtitles tend to be composed primarily of people interacting in conversations. This results in more colloquial expressions having higher frequencies in the subtitle corpus. The nature of movies themselves also played a role, resulting in an overrepresentation of words related to action movies and police matters—words like *tuer* [to kill], *prison* [jail], and *armes* [weapons] (p. 665).

For the second test of the subtitle word list, the researchers used the lexical decision times from two previous experiments. They found that the subtitle list's ability to predict lexical decision times was at least equally as accurate as the CRFP frequencies or those from a traditional corpus of written French. In many cases, it actually fared much better, surprising even the researchers themselves. However, this latter test was based on the rather small sample sizes of the two previous experiments (234 and 240 words), limiting the reliability of this test.

Picking up on these findings, and expanding the lexical decision task to a much larger sample size, Brysbaert and New (2009) compiled a corpus of English subtitles ($\text{SUBTLEX}_{\text{US}}$) and evaluated it as part of their study. This corpus is composed of subtitles from a wide variety of American films since 1900, though a majority are from 1990, as well as a large number of American television series. They found that the subtitle frequencies were especially good at predicting the lexical decision times of short words, often surpassing the accuracy of rankings based on the many written corpora they tested. It had more difficulty explaining the response times of longer words, which are more rarely found in film than in literature. Overall, their own conclusion confirmed that of the New, et al. (2007) study, that word frequencies derived from subtitle corpora seem to have a clear advantage over other types of corpora.

Though these two studies arrive at the same conclusion regarding the use of subtitles, more research is needed in this area. If, indeed, subtitles can be considered as appropriate sources for corpora of the conversation text type, their availability will open many possibilities previously made nearly impossible by the difficulty of the collection medium.

2.2 List Design

Perhaps even more complex than appropriately designing the corpus from which to extract vocabulary for a word list, researchers have found a wide range of variables that play a role in the design of the list itself. Questions addressed in the literature deal with the difference between a general service list and a specialized list, differences in the way that a "word" is defined and measured, different ranking criteria used, and the influence of subjective criteria on list creation, among other issues.

2.2.1 General Use vs. Specialized Use

Nation (2016) emphasized the importance of identifying the purpose of a word list before beginning the creation process. He believes that the main purpose of most general-use lists is to select vocabulary that language learners should learn during their first years of study. Though this may be the stated goal of some general-use lists, it is clear that they in fact serve a wide variety of purposes. He rightfully suggests, however, that the goal of serving language learners is far too broad to be very helpful. Language learners come to the task at different ages, with different language needs, and with different reasons for learning the language. A word list that is useful for adult learners intent on attending university will likely not be helpful for young leaners whose language focuses on animals, colors, and other age-appropriate material. And yet general-use lists are far more common than specialized-use lists. This is largely due to attempt at finding the language's core vocabulary.

The majority of word lists in use attempt to describe the vocabulary of the language as a whole. They are designed to be broad and all-encompassing so that they can serve any number of uses and scenarios. Essentially, they are lists that are created for general use. This broad nature of general use lists is reflected in the name of the most widely-used word list, West's *General Service List* (1953). Others include Nation's BNC/COCA lists, Browne's *New General Service List* (2014), Brezina and Gablasova's *New General Service List* (2015), and Dang and Webb's *Essential Word List* (Nation, 2016).

Another way of understanding general-use lists is that their objective is to find what

is often termed the *core* vocabulary. Though not always explicitly stated, the philosophy behind this approach is that the language being used—usually English—has at its center a self-contained lexicon of essential, primary, basic, fundamental vocabulary that then runs through the entire language. There are layers of frequency and increasing complexity beyond this, with regions of specialized language demarcated for specific purposes such as fields of study or external dialects. Still, this core vocabulary is at the center of it all, and the purpose of a word list is to identify what words fall within its boundaries. Sorell (2013) evaluated a number of definitions of core vocabulary found in the literature. He suggests that general use lists, such as West's GSL, serve as intuitively-selected lists of core written communication, whereas survival vocabulary lists—often found in travel guides or similar materials—are core vocabulary lists of oral communication.

Relatively fewer researchers have created word lists aimed at a more specific purpose or target audience. Specialized-use lists can be designed to only include words that belong to a specific domain, such as a discipline or trade. They can also encompass vocabulary found in a broad range of disciplines, but which are common in a specific context, such as academic texts. In this case, they usually serve as supplements to aid language learners who are already familiar with the core vocabulary of the language.

Perhaps the most well-known example of a specialized-use list is Coxhead's Academic Word List (2000), which replaced the University Word List (Xue & Nation, 1984) as the go-to vocabulary list for aspiring students intent on attending an English-speaking university or those entering the academic world. This is considered a *general* academic word list, since it is for academic use in general, and not for a specific discipline.

More specialized lists include those designed for business English courses, or medical English courses. This is sometimes designated *technical vocabulary*. Nation (2016) explains that technical vocabulary is most often taught after students have mastered general-use vocabulary, and after they have some familiarity with academic vocabulary. Chung and Nation (2003) looked into the nature of a technical vocabulary. By studying specialized words in the fields of anatomy and applied linguistics, they found that a large number of technical words are also found in the language's core vo-

cabulary, or have a general academic use as well. However, when used in a technical text, these words take on a specialized definition that is particular to that domain. This means that much vocabulary is shared across layers of vocabulary, though they may vary semantically, based on context.

2.2.2 Identifying Words (Word Family Levels)

One of the most essential questions that needs to be answered when designing a word list is how one is defining a *word*. Though this may seem like a straight-forward decision, it requires thorough planning and a solid understanding of the theory behind the decision. Should *jump* and *jumped* be counted as two different words or just one? What about irregular inflections such as *go* and *went*? In an article aimed at raising awareness of what he calls the "*Word* dilemma," Gardner (2007) points out that the validity of much vocabulary research hinges "on the various ways that researchers have operationalized the construct of *Word* for counting and analysis purposes" (2007, p. 242).

The literature has generally come to accept some key terms that are helpful when speaking of the way words are counted. Beginning with the most basic measurement and progressing to the most complex, we can choose to count tokens, types, lemmas, or word families.

Measuring *tokens* means simply measuring the total number of words. The sentence "I like small dogs, big dogs, and every other kind of dog" contains twelve tokens—twelve words in total. Counting *types* refers to the number of separate and distinct words. That is, *dog* and *dog* are the same type, but *dogs* is a different type—even a single difference makes them different types. The sentence above is composed of eleven types. A level above this, the *lemma* includes the stem of the word and its inflected forms, but not any derived forms of the word (derived forms are usually considered a different part of speech). So *do*, *does*, and *did* are all the same lemma, but *doable* is not. This is because *doable* has the derivational affix *-able*, which turns it into an adjective. Francis and Kučera define lemma as "a set of lexical forms having the same stem and belonging to the same major word class, differing only in inflection and/or spelling" (1982, p. 1).

Finally, the term *word family* is used to describe an even more inclusive level than the lemma. However, its precise definition has often varied among researchers. Bauer and Nation (1993) sought to rectify this problem through an in-depth classification of English affixes. Borrowing from Thorndike's (1941) study of English suffixes, their grouping was based on a series of eight criteria: frequency, productivity, predictability, regularity of the written form of the base, regularity of the spoken form of the base, regularity of the spelling of the affix, regularity of the spoken form of the affix, and regularity of function. (pp. 255–56) They identified seven "levels" of word families, with each successive one including a larger number of affixes, and therefore a larger number of types per word family. One very useful aspect of their particular system is that it places all the previous levels (type, lemma, etc.) within the same framework. Under their schema, a level 1 word family is the same as a type, a level 2 word family is a lemma (including all regular inflected affixes), and level 7 (the highest level) consists of classical roots and affixes beyond what most speakers any longer consider separate affixes.

Nation himself suggests that for the purposes of language learning, these specific family word levels can be used simply "as a starting point as an initial framework of reference" (2016, p. 36). That is, they are one interpretation of how to systematically count words for a frequency list. These levels are based on criteria that reflect the needs of language learners, rather than on any psycholinguistic theory of how speakers' mental lexicon is arranged. Still, the idea of word families aligns closely with theoretical models that dictate morphological decomposition as a constant. These theories propose that words are often deconstructed into independent morphemes in receptive tasks and recognized that way, for example by deconstructing *jumping* into *jump* and *-ing*. At the other end of the spectrum stand theories that would place *jump* and *jumping* as separate lexical entries (Brysbaert and New, 2009, 982–83).

Either way, there is strong evidence to suggest that inflected/derived forms and their base forms do affect each other in some way, suggesting that word families are a measure of a real representation in speakers' mental lexicon. In one such study, Nagy et al. (1989) explored the effect of both inflectional and derivational family frequency during a lexical decision task. They found that both types of morphological relationships lowered word recognition times, leading to the conclusion that inflections and derivational relationships are both represented in the mental lexicon,

either through the grouping of related words under the same entry, or through linked entries. However, all the participants were native English speakers, so to what extent do L2 learners' lexicons reflect the same level of linking?

More recent studies have found that L2 learners' morphological knowledge and word-building ability are not nearly as developed. Ward and Chuenjundaeng (2009) conducted a study that tested the receptive ability of Thai engineering and doctoral students learning English. They were tested for their knowledge of a series of base words, together with various derived forms of the same words. They found a surprising lack of familiarity with the derived words, even when participants knew the base forms from which they were derived. Similarly, but from a productive and not receptive standpoint, Schmitt and Zimmerman (2002) found that learners could produce only a limited number of derived forms when presented with a word family headword. These results challenge the common assumption that "once the base word or even a derived word is known, the recognition of other members of the family requires little or no extra effort" (Bauer and Nation, 1993, p. 253).

There is evidence (Mochizuki and Aizawa, 2000; Schmitt and Meara, 1997) to suggest a positive correlation between vocabulary size and morphological knowledge. If this is the case, then using higher-level word families in Bauer and Nation's framework for word list creation (as is the case in), may not be appropriate for learners with limited knowledge of vocabulary—the very learners that many of these lists target.

Similarly, a study by Jeon (2011) found that L2 learners' morphological knowledge leads to greater reading comprehension. Since many word lists are designed to increase reading comprehension in learners, it follows that they will likely be used by students without strong word-building abilities.

Clearly, then, when it comes to creating a word list, the unit of counting needs to fit the purpose and target audience of that list. Brezina and Gablasova (2015) contend that Bauer and Nation's (1993) higher word family levels ignore the lack of transparency that exists between many of the entries that would be placed under the same word family. Especially when creating a word list for beginners, Brezina and Gablasova point out that the morphological knowledge of language learners is often not developed enough. Because their New General Service List was created for beginners, and since it is intended to aid vocabulary acquisition for both receptive

and productive purposes, Brezina and Gablasova chose the lemma as their unit of measure.

Seeking to quantify the effect of choosing to measure word families as opposed to word types, Sorell (2013) compared the text coverage of frequency lists made from the same four corpora. Each corpus corresponded to one of Sorell's text types (see above). Sorell's definition of "word families" was a slightly modified version of Bauer and Nation's (1993) sixth level of affix inclusion. He found, as would be expected, that the most frequent word families have a much larger text coverage than the most frequent types. This is especially true when measuring type coverage—the most frequent word families accounted for roughly 4–6 times as many types in each corpus. However, when measuring overall token coverage, the top word families only covered about 3–10% more than the same number of most frequent types. Sorell also found that the most frequent 1,000 word families consisted of 6,557 word types in the general writing corpus. The number was similar in the other text types, though somewhat lower.

2.2.3 Objective vs. Subjective Design

(Nation 2016:133) >There are two major approaches to making corpus-based word lists. One is to stick strictly to criteria based on range, frequency and dispersion (Brezina & Gablasova, 2015; Dang & Webb, Chapter 15 this volume; Leech, Rayson & Wilson, 2001). The other is to use a similar statistical approach but to adjust the results using other criteria such as ensuring that lexical sets such as numbers, days of the week, months.

Brezina and Gablasova (2015), p. 3: > Seen from the perspective of current corpus linguistic research (cf. McEnery and Hardie 2011), one of the main problems of West's GSL lies in the fact that its compilation involved a number of competing principles that brought a large element of subjectivity into the final product. When reviewing the compilation principles of the GSL, we can see that in addition to the quantitative measure of word frequency, West also used a number of 'qualitative' criteria for the selection of individual lexical items. These include (i) the ease of learning, (ii) necessity, (iii) cover, and (iv) stylistic and emotional neutrality (West

1953: ix–x). Let us now briefly discuss these principles.

2.2.4 Objective Criteria (Frequency, Range, Dispersion)

Nation (2016), p. 103: > Dividing a corpus into sub-corpora allows the creation of range and dispersion figures. In some ways range figures are more important than frequency figures, because a range figure shows how widely used a word is, and this indicates its "general service". Brysbaert and New (2009) found that a range measure was a good predictor of lexical decision times. Carroll, Davies and Richman (1971) found in their study that frequency and their measure of dispersion correlated at .8538 (page xxix), showing that the more widely used a word is, the more likely it is to be frequent. Some words however are frequent in just one or two texts or sub-corpora and may not even occur in others. The use of a range or dispersion figure or both can indicate such words.

Brysbert and New (2009), pp. 984–5: > Another variable that has been proposed as an alternative to WF frequency is the contextual diversity (CD) of a word (Adelman, Brown, & Quesada, 2006). This variable refers to the number of passages (documents) in a corpus containing the word. So, rather than calculating how often a word appeared in the BNC, Adelman et al. measured how many of the 3,144 text samples in the corpus contained the word. They found that the CD measure explained 1%–3% more of the variance in the Elexicon data.

Brezina and Gablasova (105), p. 8: > ARF is a measure that takes into account both the absolute frequency of a lexical item and its distribution in the corpus (Savicky´and Hlava´c ˇ ova´2002; Hlava´c ˇ ova´2006). Thus if a word occurs with a relatively high absolute frequency only in a small number of texts, the ARF will be small (cf. Cerma´k and Kr ˇ en 2005; Kilgarriff 2009). All four wordlists were then sorted according to the ARF that ensured that only words that are frequent in a large variety of texts appeared in the top positions in the wordlists.

Sorell (2013), p. 89: Dispersion.

2.3 Modern Non-English Word Lists

Gardner, D. (2007), p. 242: > Hazenberg and Hulstijn 1996—Dutch language;

3 Methods: Creating the Conversational Hebrew Vocabulary List (CHVL)

3.1 Overview

As we have seen, the brunt of the effort in high-quality vocabulary frequency list creation has been in the creation of *English* frequency lists. Outside of the English-speaking world, and especially when dealing with less commonly taught languages, the quality of lists is difficult to assess, if they exist at all. Why have not more educators—those who may benefit from these lists the most—decided to undertake such a task?

This need not be a project that one starts from scratch every time. Many tools already exist to make the process smoother. Still, with the rapid pace at which technology changes, these tools tend to quickly become obsolete. They are also usually restrictive to the specific preferences of their creators.

Rather than using these tools, I chose to create a series of (almost embarrassingly-) simple scripts to create the Conversational Hebrew Vocabulary List.

I wrote all of these scripts using the programming language Python. Python was created specifically to be

Beyond explaining the theory behind the decisions that have gone into creating the Conversational Hebrew Vocabulary List, this thesis aims to make the process as reproducible as possible.

3.2 The corpus

Before coding or analyzing anything, it's important to find an appropriate corpus to use and to become familiar with its structure. A useful place to begin is OPUS, which is part of the Nordic Language Processing Laboratory (NLPL), and hosted by the CSC IT center in Finland. OPUS is a database of many open, parallel corpora. These include corpora of movie and television subtitles, TED talks, web-crawled data,

newspapers, and of course, books. The corpora are all free and open to the public.

The CHVL was created using one of OPUS's corpora, the OpenSubtitles2018 corpus. The corpus can be downloaded in a variety of formats, and can be downloaded either as *parallel* corpora, or as a monolingual corpus. A parallel corpus consists of two languages interwoven together. For example, a line from the English subtitles of a movie will be paired with the same line from the French subtitles of the same movie. In theory, this means that each line of the corpus should have the same meaning in two different languages. The creation of parallel corpora has made possible many interesting and useful tools for linguistics, translators, and language learners. These include the open-source CASMACAT project and the ReversoContext tool.

For the purpose of creating a word list, a monolingual corpus is best. Note that parallel corpora will often be composed of less tokens than monolingual ones. This is because parallel corpora will only include movies for which the subtitles exist in both selected languages.

Though it's possible to download plain text files, the most useful format available for download is XML. Indeed, the most common file format used for large corpora is XML. The XML structure allows for nested key-value pairs, which are especially useful for parsed corpora that contain extensive metadata. XML is comparable to JSON, which we will use later to extract specific movie metadata directly from a database.

Another factor to consider is whether to download an untokenized, tokenized, or parsed corpus. An untokenized corpus contains simply the raw lines of text as found in the original subtitle files (divided into lines as they would appear while watching the movie, and labeled with the appropriate time for them to be shown):

```
<s id="49">
  <time id="T39S" value="00:03:22,280" />
 ?      ,
  <time id="T39E" value="00:03:24,120" />
</s>
```

A tokenized corpus has further been split into individual words and punctuation,

such that each word is tagged on its own:

```
<s id="49">
  <time id="T39S" value="00:03:22,280" />
  <w id="49.1"> </w>
  <w id="49.2">  </w>
  <w id="49.3">   </w>
  <w id="49.4">,</w>
  <w id="49.5">   </w>
  <w id="49.6">?</w>
  <time id="T39E" value="00:03:24,120" />
</s>
```

A parsed corpus contains much more information for each token. The data included depends on the features of the language and on the parsing script used, but it can include things such as part of speech, syntactic role, lemma, and even specific features like gender, person, and number. Here is an example:

```
<s id="49">
  <time value="00:03:22,280" id="T39S" />
  <w xpos="ADV" head="49.3" feats="PronType=Int" upos="ADV" lemma=" "
      id="49.1" deprel="obj"> </w>
  <w xpos="PRON" head="49.3" feats="Gender=Masc|Number=Sing|Person=2|
      PronType=Prs" upos="PRON" lemma="  " id="49.2" deprel="nsubj">  </w>
  <w xpos="VERB" head="0" feats="Gender=Masc|HebBinyan=PAAL|Number=Sing|
      Person=1,2,3|VerbForm=Part|Voice=Act" upos="VERB" misc="SpaceAfter=No"
      lemma="  " id="49.3" deprel="root">   </w>
  <w xpos="PUNCT" head="49.3" upos="PUNCT" lemma="," id="49.4"
      deprel="punct">,</w>
  <w xpos="NOUN" head="49.3" feats="Gender=Masc|Number=Sing" upos="NOUN"
      misc="SpaceAfter=No" lemma="   " id="49.5" deprel="obj">   </w>
  <w xpos="PUNCT" head="49.3" upos="PUNCT" misc="SpaceAfter=No" lemma="?"
      id="49.6" deprel="punct">?</w>
```

```
  <time value="00:03:24,120" id="T39E" />
</s>
```

All of the data used to create the CHVL came from a monolingual parsed corpus of Hebrew. The parsing was all done automatically using .

3.3 Cleaning the corpus

Once downloaded, the files inside the zipped folder are organized as follows:

```
Zipped folder in GZ format
    Folder for year X
        Folder for movie A
            Zipped XML in GZ format
            Zipped XML in GZ format
            Zipped XML in GZ format
        Folder for movie B
            Zipped XML in GZ format
            Zipped XML in GZ format
    Folder for year Y
        Folder for movie C
            Zipped XML in GZ format
        Folder for movie D
            Zipped XML in GZ format
            Zipped XML in GZ format
            Zipped XML in GZ format
        Folder for movie E
            Zipped XML in GZ format
            Zipped XML in GZ format
    Folder for year Z
          Folder for movie F
              Zipped XML in GZ format
              Zipped XML in GZ format
```

This organization is straight-forward, except for the fact that there are multiple XML files for each movie. The subtitle files that OPUS has collected, parsed, organized, and made available for mass download were all obtained from the Open Subtitles project (hence the name of the corpus). Because this is a database where users can upload the subtitle files they extract from their own movie collection, there are often multiple uploads for the same movie. For our purposes, this results in movies that can have anywhere from a single subtitle file to dozens of them. Unfortunately, though the tokens in the files themselves are usually the same (with only minor variations in the XML metadata), this is not always true. Some few variations seem to be different and independent translations.

Part of cleaning the corpus, then, entails getting rid of these duplicates. As a means of simplifying the entire process, I chose simply to use the first file in each movie folder. I've included the short Python script for this in its entirety in Appendix 3.3. However, I will here explain what it does in detail so that it can be easily modified to fit different circumstances.

The script first makes a copy of the entire folder structure in the original downloaded (and unzipped!) corpus into a new directory. It then finds the first XML file in each movie folder and copies it into the appropriate place in the new folder structure. This means that it doesn't delete or otherwise change the files in the original corpus in any way.

The first block of code imports necessary modules that are used later in the script (`shutil` and `os`). Lines 7 and 8 define where the original corpus is (`source`), and where the new one will be placed (`destination`).

```
4 import shutil
5 import os
6
7 source = '../OpenSubtitles2018_parsed'
8 destination = './OpenSubtitles2018_parsed_single'
```

Next, a single line of code copies all directories and subdirectories into their new location.

```
11 shutil.copytree(source, destination, ignore=shutil.ignore_patterns('*.*'))
```

Lastly, we create a variable that holds all the XML files located in each movie folder, trim the list to just one, and copy that one into its new location. This process is carried out for one movie folder at a time. The originals are left untouched.

```
14 for dirName, subdirList, fileList in os.walk(source):
15     for fname in fileList:
16         if fname == '.DS_Store':
17             fileList.remove(fname)
18     if len(fileList) > 0:
19         del fileList[1:]
20         src = dirName + '/' + fileList[0]
21         dst = destination + dirName[27:] + '/'
22         shutil.copy2(src, dst)
```

With a newly organized version of the corpus, it's now possible to begin the process of reading and processing data. At this stage, I took some time to gather metadata for all the movies in the corpus in order to identify movies that were originally filmed with Hebrew as their primary language (as opposed to translated subtitles). Because I ultimately decided against this approach for the creation of the CHVL, I will skip that step here. However, a description of that entire process can be found in section 4.4.1 - using original-language movies exclusively.

3.4 Reading data

Before calculating any measures such as frequency, individual lemmas must be extracted from the XML files in the downloaded corpus. There are two ways to go about this. Because XML consists of nested tags and key-value pairs, a dedicated XML parsing tool can be used to extract specific information. In this case we would be creating a list of all *values* in the `'lemma'` *key* within each `<w>` *tag*. The value that corresponds to the `'lemma'` tag below for the word is .

```
<w xpos="VERB" head="0" feats="Gender=Masc|HebBinyan=PAAL|Number=Sing|
    Person=1,2,3|VerbForm=Part|Voice=Act" upos="VERB" misc="SpaceAfter=No"
    lemma="  " id="49.3" deprel="root">  </w>
```

A different approach is to use *regular expressions* to search for a specific string of characters and extract every instance of that string. This is a more brute-force approach, since it ignores the structure of the XML file and treats it all simply as raw text. To find a lemma, a very simple regular expression is sufficient: `lemma="[-]+"`. This will search for any instance of the characters `lemma="`, followed by a combination of any number of Hebrew letters (at least one), followed by the character `"`.

Despite the existence of various Python modules for parsing XML files, I found a simple search using regular expressions to be more efficient for various reasons. First, not all elements in the parsed corpus contain *lemma* attributes. Second, punctuation and non-Hebrew words are often lemmaticized. This means that even after extracting all the *lemma* values in a file, I would still need to use regular expressions to search through the results and delete any that contain non-Hebrew characters. I chose instead to skip the XML parsing step altogether.

I will now explain the code in the script used to create the CHVL. As with the other code, the entire script in its entirety can be found in the appendix (2.1).

After importing necessary packages and initializing variables, two functions near the beginning of the script serve to open a file and extract a list of lemmas from it.

```
37 # Open XML file and read it.
38 def open_and_read(file_loc):
39     with gzip.open(file_loc, 'rt', encoding='utf-8') as f:
40         read_data = f.read()
41     return read_data


44 # Search for lemmas and add counts to "lemma_by_file_dict{}".
45 def find_and_count(doc):
46     file = str(f)[40:-3]
```

```
47      match_pattern = re.findall(r'lemma="[ '"+[ -, doc)
48      for word in match_pattern:
49          if word[7:-1] in lemma_by_file_dict:
50              count = lemma_by_file_dict[word[7:-1]].get(file, 0)
51              lemma_by_file_dict[word[7:-1]][file] = count + 1
52          else:
53              lemma_by_file_dict[word[7:-1]] = {}
54              lemma_by_file_dict[word[7:-1]][file] = 1
```

We then run both of these functions for each XML file in the corpus directory (defined earlier in `corpus_path`).

```
64  for dirName, subdirList, fileList in os.walk(corpus_path):
65      if len(fileList) > 0:
66          f = dirName + '/' + fileList[0]
67          find_and_count(open_and_read(f))
```

The `find_and_count()` function finds each instance of the string described above using a regular expression, then adds the Hebrew part of the string—the lemma itself—to a dictionary. The dictionary is named `lemma_by_file_dict`, and its structure looks like this:

```
'lemma': {'path of file': 'frequency of lemma in file'}
```

A dictionary is at its core a list of key:value pairs. Much like an actual dictionary consists of words and their definitions, this dictionary's keys are made up of all the individual lemmas found by our search. For each lemma, the value is another dictionary—making it a nested dictionary, or a dictionary within a dictionary. The keys for each inner dictionary are the paths of all the XML files (movies) that the lemma appears in, and the value of each is an integer that represents how many times that lemma appears in that file (frequency).

After the script reads each file, it returns a complete dictionary. Here is a sample:

```
:' ' }
        '/he/0/5753574/6853341.xml': 168,
        '/he/0/3607000/5764778.xml': 94},
:'  ' }
        '/he/0/5753574/6853341.xml': 3},
:'   ' }
        '/he/0/5753574/6853341.xml': 6,
        '/he/0/3607000/5764778.xml': 2,
        '/he/0/1278351/3777598.xml': 1}
```

Throughout the rest of the script, this nested dictionary serves as the basis for all of the calculations needed.

3.5 Calculations

For each lemma, the CHVL includes three measures: frequency, range, and U_{DP} (dispersion). It uses dispersion as its sorting value. Let's look at how each of these is calculated. Range will be addressed in the export section, since the script calculates it on the spot as the list is created.

3.5.1 Frequency

Since we've already calculated the frequency of each lemma for each individual file, calculating total frequency per lemma is straight forward. The script simply creates a new dictionary, `lemma_totals_dict`, and adds to it every lemma in the corpus as its keys, with the corresponding value being a sum of the frequencies in all files for that lemma. In other words, {'lemma1':'frequency1', 'lemma2':'frequency2', . . . }

```
116 for lemma in lemma_by_file_dict:
117     lemma_totals_dict[lemma] = sum(lemma_by_file_dict[lemma].values())
```

This returns Using the short example given above, this would result in the following dictionary:

```
' ':262,
'  ':3,
'   ':9
```

3.5.2 U_{DP} (dispersion)

Dispersion is more complicated. In theory, it should provide a single quantifiable measure that incorporates both frequency and range, and which can then be used to sort the word list. There is no agreed-upon, single way to calculate dispersion, and different researchers will use the words in slightly different contexts. The model of dispersion I have chosen to follow for this project is Gries' dispersion coefficient, or U_{DP}, () calculated from Gries' DP. ()

In order to calculate Gries' DP for $lemma_x$, we must first make two calculations for each file in the corpus ($file_i$): the lemma's *expected frequency* if it were perfectly distributed, and its *observed frequency*—or its actual frequency.

$$\textbf{expected frequency} = \frac{tokens\ in\ file_i}{tokens\ in\ corpus}$$

$$\textbf{observed frequency} = \frac{frequency\ of\ lemma_x\ in\ file_i}{frequency\ of\ lemma_x\ in\ corpus}$$

We must then subtract the lemma's observed frequency from its expected frequency, which will return a value between -1 and 1. We can normalize this result by finding the absolute value. Now the closer the result is to 0, the closer that lemma's frequency is in that particular file to what we would expect if it were perfectly distributed throughout the corpus. A higher number (closer to 1), would indicate a heavier load in that file that we would expect.

By performing this calculation for every file in the corpus, adding them all together, and dividing the result by two (since we're using the absolute value and are therefore adding values originally in both directions), we now have Gries' DP. Where `n` is the number of files:

$$\textbf{DP} \;=\; 0.5 \sum_{i=1}^{n} |\text{ expected frequency } - \text{ observed frequency }|$$

A DP of 0 represents a perfectly even dispersion, and a DP close to 1 means a more uneven distribution, where fewer files contain a larger load of the lemma's overall frequency. A DP of 1 is not actually possible.

Gries' usage coefficient, or U_{DP}, is an attempt to make this number more useful. DP is first subtracted from 1 and the result is multiplied by the lemma's total frequency. The full equation for U_{DP} is as follows:

$$\left(1 - 0.5 \sum_{i=1}^{n} \left| \frac{file_i \; tokens}{total \; tokens} \; - \; \frac{frequency_x \; in \; file_i}{total \; frequency_x} \right| \right) \times total \; frequency_x$$

In order to calculate this, the script must first find the number of tokens in each file. Like before, this is done by creating a dictionary, `token_count_dict`, which contains the key:value pairs of file:tokens. Since we already have a dictionary with the number of times that each lemma appears in each file, `lemma_by_file_dict`, we don't need to open and read the files again. Instead, we can add the values in this dictionary and rearrange them into what we want.

```
120 for lemma in lemma_by_file_dict:
121     for file in lemma_by_file_dict[lemma]:
122         token_count_dict[file] = token_count_dict.get(
123             file, 0) + lemma_by_file_dict[lemma][file]
```

We also need to know the total number of tokens in the entire corpus. This is a simple matter of adding all the values in the `token_count_dict` dictionary. The final count is saved into an integer variable, `total_tokens_int`.

```
126 for file in token_count_dict:
127     total_tokens_int = total_tokens_int + token_count_dict.get(file, 0)
```

Finally, the script uses all these measures to calculate DP and then U_{DP} for each lemma, and places them into their respective dictionaries, `lemma_DPs_dict` and `lemma_UDPs_dict`.

```
# Calculate DPs
for lemma in lemma_by_file_dict.keys():
    for file in lemma_by_file_dict[lemma].keys():
        lemma_DPs_dict[lemma] = lemma_DPs_dict[lemma] + abs(
            (token_count_dict[file] /
             total_tokens_int) -
            (lemma_by_file_dict[lemma][file] /
             lemma_totals_dict[lemma]))
lemma_DPs_dict = {lemma: DP/2 for (lemma, DP) in lemma_DPs_dict.items()}

# Calculate UDPs
lemma_UDPs_dict = {lemma: 1-DP for (lemma, DP) in lemma_DPs_dict.items()}
```

With these values all calculated for each lemma, the only thing left is to sort and create the final list.

3.6 Sort and export

In order to ensure that the words on the list do not have an abnormally high frequency in some subcorpora (movies) and are nearly absent in others, some have suggested setting a minimum range or dispersion. All words that fall below this threshold are discarded, and the remaining words can then be sorted by frequency.

Though this is a more systematic approach than that used to create many early frequency lists, it still depends on a subjective decision and the whim of the researcher.

Rather than setting an arbitrary bar, the CHVL is sorted entirely by Gries' usage coefficient of dispersion (U_{DP}). This *modus operandi* ensures that the order of words itself—not just which words make it onto the list and which don't—is decided by

a combination of both relevant measures: frequency and dispersion. This approach also has the added benefit of being entirely objective.

Since we've already calculated the U_{DP} for each lemma, sorting the list is simple.

```
148 UDP_sorted_list = [(k, lemma_UDPs_dict[k]) for k in sorted(
149     lemma_UDPs_dict, key=lemma_UDPs_dict.__getitem__,
150     reverse=True)]
```

A final table is then created (using a list of tuples, `table_list`), with each line consisting of a lemma, its overall frequency, its range, and its U_{DP}. This table is already sorted by U_{DP} as it's being created.

Because the script has not calculated range by this point, it must do so on the spot as it's entering each lemma into the table. It does this with a simple dictionary comprehension that quickly counts the number of files included in the `lemma_by_file_dict`. Here is the resulting code:

```
153 for k, v in UDP_sorted_list[:list_size_int]:
154     table_list.append((k, lemma_totals_dict[k], sum(
155         1 for count in lemma_by_file_dict[k].values() if count > 0),
156         v))
```

Lastly, now that everything is organized into a table, the script opens (or creates, if it doesn't yet exist) a CSV file, writes a header line into it (`LEMMA, FREQUENCY, RANGE, UDP`), and exports the entire table into the file. It then closes it to clear the computer's memory cache.

```
199 result = open('./export/WordList.csv', 'w')
200 result.write('LEMMA, FREQUENCY, RANGE, UDP\n')
201 for i in range(list_size_int):
202     result.write(str(table_list[i][0]) + ', ' +
203                  str(table_list[i][1]) + ', ' +
204                  str(table_list[i][2]) + ', ' +
```

```
205                     str(table_list[i][3]) + '\n')
206 result.close()
```

The list is now complete. The next section will explore the list itself more in-depth.

4 The CHVL: A vocabulary list of conversational Modern Hebrew

4.1 Organization

4.2 Use

4.3 Expansion

4.4 Challenges and future direction

4.4.1 Using original-language movies exclusively

One of the potential downsides of using the OpenSubtitles2018 corpus is that it includes all subtitles of a specific language, even *translated* subtitles from movies filmed in other languages. The question is, does a translated script represent true conversational language as well as an original script?

This is a question that requires more research in order to answer satisfactorily. Though translated subtitles don't need to try to approximate the length and mouth shapes that a dubbed script does, its quality still largely depends on the skills of a translator. Most importantly, it's possible that a translation will not accurately reflect the register of the original. Again, these are important points to consider.

One solution is to simply use movies that were originally filmed in the target language of the corpus. In theory, each XML file in a monolingual OpenSubtitles2018 file should contain a tag that identifies the original language of the movie. In practice, I found that the overwhelming majority of the files contained an empty `<lang>` tag instead. Luckily, there is a way to obtain the desired metadata for each movie in the corpus.

This can be done with a script that uses an application programming interface (API) to fetch specific information from an online movie database. The name of each movie folder in the corpus, which is simply a series of numbers, corresponds to that movies

IMDb ID, which is a unique ID registered with the Internet Movie Database. This makes the process relatively easy, as we simply need to query the database using this ID to receive all of the movie's metadata.

Though IMDb does provide their own API, I decided instead to use an API created for the Open Movie Database (OMDb). This API can be used free-of-charge, but it has a 1,000 movie limit per day. Since the OpenSubtitles2018 Hebrew corpus contains nearly 50,000 movies, I decided instead to pay for a daily limit of 100,000 movies. This only requires a $1.00 donation for each month that one is registered to use the OMDb API.

Once an API key is obtained, a script can be written to obtain the information desired for every movie all at once. In this case, we want to know the original language(s) for each movie.

This script in its entirety is found in Appendix 2.2. It uses an imported Python wrapper for the API, written by Derrick Gilland, which can be found at https://github.com/dgilland/omdb.py. This package can be installed through PIP by entering `pip install omdb` into the command line.

For practical purposes, the script requires one to enter a specific year (or, more accurately, corpus folder name). If desired, an asterisk can act as wildcard: `python OMDb-fetch.py 1988` will fetch data for movies from 1988, while `python OMDb-fetch.py 198*` will do it for all movies in the 1980s. In order to fetch data for all movies in the database at once, use `python OMDb-fetch.py *`. I don't recommend this, however, since it may overload the server and cause the script to time out.

The script begins by creating a list of all movie directory paths for the desired year.

```
15 for name in glob.glob(
16         './OpenSubtitles2018_parsed_single/parsed/he/' + year + '/*/'):
17     IDs.append(name)
```

Each item in the list is then trimmed to include only the name of the movie folder, which is *almost* equivalent to the IMDb ID.

```
20 IDs = [os.path.basename(os.path.dirname(str(i))) for i in IDs]
```

In order to make the IDs match those in the database, additional zeros must be added to the beginning until they are seven digits long.

```
23 for i in IDs:
24     while len(i) < 7:
25         IDs[IDs.index(i)] = '0' + i
26         i = '0' + i
```

The list is then sorted numerically in order to more easily interpret the results: `IDs.sort()`.

The API key is set in line 32, but be sure to replace `906517b3` with your own key, which can be obtained at http://www.omdbapi.com/.

```
32 omdb.set_default('apikey', '906517b3')
```

The script then prints a table header, fetches the title, year, and language(s) for each movie, and prints the results directly into the computer terminal.

```
35 print('# ' + year + '\n' +
36       'IMDb ID\tTitle\tYear\tLanguage(s)')


39       for i in IDs:
40           doc = omdb.imdbid('tt' + i)
41           print('tt' + i + '\t' +
42                 doc['title'] + '\t' +
43                 doc['year'] + '\t' +
44                 doc['language'])
```

5 Implications for other less commonly taught languages

5.1 Easy reproducibility and growth

Appendix 1: Conversational Hebrew Vocabulary List (CHVL)

Appendix 2: Scripts

Appendix 2.1: HebrewLemmaCount.py

```
#! /usr/bin/env python3
# -*- coding: utf-8 -*-

import re
import os
import gzip
from collections import defaultdict


############################################################
# ---------------- INITIALIZE VARIABLES ---------------- #
############################################################

# Define path for topmost directory to search. Make sure this points to
# the correct location of your corpus.
corpus_path = './OpenSubtitles2018_parsed_single'

# Initialize dictionaries
lemma_by_corpus_dict = {}
lemma_totals_dict = {}
token_count_dict = {}
lemma_DPs_dict = defaultdict(float)
lemma_UDPs_dict = defaultdict(float)

total_tokens_int = 0
table_list = []

# Set size of final list
list_size_int = 5000
```

```
30
31
32  ##############################################################
33  # ------------------- DEFINE FUNCTIONS ------------------- #
34  ##############################################################
35
36
37  # Open XML file and read it.
38  def open_and_read(file_loc):
39      with gzip.open(file_loc, 'rt', encoding='utf-8') as f:
40          read_data = f.read()
41      return read_data
42
43
44  # Search for lemma and add counts to "frequency{}".
45  def find_and_count(doc):
46      corpus = str(f)[38:-4]
47      match_pattern = re.findall(r'lemma="[ '"+[ -, doc)
48      for word in match_pattern:
49          if word[7:-1] in lemma_by_corpus_dict:
50              count = lemma_by_corpus_dict[word[7:-1]].get(corpus, 0)
51              lemma_by_corpus_dict[word[7:-1]][corpus] = count + 1
52          else:
53              lemma_by_corpus_dict[word[7:-1]] = {}
54              lemma_by_corpus_dict[word[7:-1]][corpus] = 1
55
56
57  ##############################################################
58  # ------------------- OPEN AND READ ---------------------- #
59  ##############################################################
60
61  # Open and read all files. If calculating only for a specific language,
62  # comment out this code and uncomment the large block that follows.
63  #
```

```
for dirName, subdirList, fileList in os.walk(corpus_path):
    if len(fileList) > 0:
        f = dirName + '/' + fileList[0]
        find_and_count(open_and_read(f))

############################################################
# ---------------- LANGUAGE-SPECIFIC BLOCK -----------------
#
# This large block of code is for creating a list using only movies #
# with a specific primary language (in this case, Hebrew). Be sure to #
# uncomment the relevant lines of code, and to comment out the block #
# above. #
#
#
# Create list of IDs for movies with Hebrew as primary language. #
# This makes use of a text file that must already exist with this list. #
#
# Hebrew_IDs_list = []
# with open('./Hebrew_originals.txt', 'r', encoding='utf-8') as f:
#     read_data = f.read()
#     Hebrew_IDs_list = re.findall(r'\s\stt[0-9]+\t', read_data)
# Hebrew_IDs_list = [line[4:-1] for line in Hebrew_IDs_list]
#
#
# Delete extra 0s at the beginning of Hebrew movie IDs. #
#
# for item in Hebrew_IDs_list:
#     if item[0] == '0':
#         Hebrew_IDs_list[Hebrew_IDs_list.index(item)] = item[1:]
# for item in Hebrew_IDs_list:
#     if item[0] == '0':
#         Hebrew_IDs_list[Hebrew_IDs_list.index(item)] = item[1:]
#
#
```

```
98  # Open and read files for movies with Hebrew as the primary language. #
99  #
100 # for dirName, subdirList, fileList in os.walk(corpus_path):
101 #     if len(fileList) > 0:
102 #         f = dirName + '/' + fileList[0]
103 #         folders = re.split('/', dirName)
104 #         if folders[len(folders)-1] in Hebrew_IDs_list:
105 #             find_and_count(open_and_read(f))
106 #
107 # ------------- END OF LANGUAGE-SPECIFIC BLOCK -------------
108 ##################################################################
109
110
111 ##################################################################
112 # --------------------- CALCULATIONS --------------------- #
113 ##################################################################
114
115 # Calculate token count per corpus
116 for lemma in lemma_by_corpus_dict:
117     for corpus in lemma_by_corpus_dict[lemma]:
118         token_count_dict[corpus] = token_count_dict.get(
119             corpus, 0) + lemma_by_corpus_dict[lemma][corpus]
120
121 # Calculate total frequencies per lemma
122 for lemma in lemma_by_corpus_dict:
123     lemma_totals_dict[lemma] = sum(lemma_by_corpus_dict[lemma].values())
124
125 # Calculate total token count
126 for corpus in token_count_dict:
127     total_tokens_int = total_tokens_int + token_count_dict.get(corpus, 0)
128
129 # Calculate DPs
130 for lemma in lemma_by_corpus_dict.keys():
131     for corpus in lemma_by_corpus_dict[lemma].keys():
```

```
        lemma_DPs_dict[lemma] = lemma_DPs_dict[lemma] + abs(
            (token_count_dict[corpus] /
             total_tokens_int) -
            (lemma_by_corpus_dict[lemma][corpus] /
             lemma_totals_dict[lemma]))
lemma_DPs_dict = {lemma: DP/2 for (lemma, DP) in lemma_DPs_dict.items()}

# Calculate UDPs
lemma_UDPs_dict = {lemma: 1-DP for (lemma, DP) in lemma_DPs_dict.items()}


#############################################################
# -------------- SORT LIST AND CREATE TABLE -------------- #
#############################################################

# Sort entries by UDP
UDP_sorted_list = [(k, lemma_UDPs_dict[k]) for k in sorted(
    lemma_UDPs_dict, key=lemma_UDPs_dict.__getitem__,
    reverse=True)]

# Create list of tuples with all values (Lemma, Frequency, Range, UDP)
for k, v in UDP_sorted_list[:list_size_int]:
    table_list.append((k, lemma_totals_dict[k], sum(
        1 for count in lemma_by_corpus_dict[k].values() if count > 0),
        v))

#############################################################
# ---------------- SORT-BY-FREQUENCY BLOCK -----------------
#
# Sort entries by raw frequency (total lemma count). To sort the final #
# list by frequency instead of UDP, comment out the above code within the #
# "SORT LIST AND CREATE TABLE" section, and also uncomment the relevant #
# lines of code in this block. #
#
```

```
#
# Sort entries by raw frequency #
#
# frequency_sorted_list = [(k, lemma_totals_dict[k]) for k in sorted(
#     lemma_totals_dict, key=lemma_totals_dict.__getitem__,
#     reverse=True)]
#
#
# Create list of tuples with all values (Lemma, Frequency, Range, UDP) #
#
# for k, v in frequency_sorted_list[:list_size_int]:
#     table_list.append((k, v, sum(
#         1 for count in lemma_by_corpus_dict[k].values() if count > 0),
#         lemma_UDPs_dict[k]))
#
# ------------- END OF SORT-BY-FREQUENCY BLOCK -------------
#################################################################

# Calculate list size for 80% coverage and set that as the list size. Note
# that if the initial list_size_int (set near the beginning of the script)
# provides less than the desired coverage, it will default to that instead.
#
# added_freq_int = 0
# count = 0
# for k, v in UDP_sorted_list:
#     if added_freq_int / total_tokens_int < 0.8:
#         added_freq_int = added_freq_int + lemma_totals_dict[k]
#         count = count + 1
#     else:
#         break
# list_size_int = count

# Write final tallies to CSV file
result = open('./export/HebrewWordList2.csv', 'w')
```

```
200 result.write('LEMMA, FREQUENCY, RANGE, UDP\n')
201 for i in range(list_size_int):
202     result.write(str(table_list[i][0]) + ', ' +
203                  str(table_list[i][1]) + ', ' +
204                  str(table_list[i][2]) + ', ' +
205                  str(table_list[i][3]) + '\n')
206 result.close()
207
208 # Print final tallies. Uncomment this code to see the results
209 # printed instead of writing them to a file.
210 #
211 # for i in range(list_size_int):
212 #     print('Lemma: ' + table_list[i][0] +
213 #           '\tFrequency: ' + str(table_list[i][1]) +
214 #           '\tRange: ' + str(table_list[i][2]) +
215 #           '\tUDP: ' + str(table_list[i][3]))
```

Appendix 2.2: OMDb-fetch.py

```
#! /usr/bin/env python3
# -*- coding: utf-8 -*-

# import re
from sys import argv
import os
import glob
import omdb

# year = '1996'
script, year, id_start = argv

dirs = []
p = []


for name in glob.glob(
        '../OpenSubtitles2018_parsed/parsed/he/' + year + '/*/'):
    p.append(name)
# p = Path('../OpenSubtitles2018_parsed/parsed/he')
# p = list(p.glob('[198-199]*/*/*.xml'))

p = [os.path.basename(os.path.dirname(str(i))) for i in p]

for i in p:
    if i not in dirs:
        dirs.append(i)

for i in dirs:
    while len(i) < 7:
        dirs[dirs.index(i)] = '0' + i
        i = '0' + i
```

```
33
34 dirs.sort()
35
36 # for i in dirs:
37 #     print('tt' + i)
38
39 print('# ' + year + '\n' +
40       'IMDb ID\tTitle\tYear\tLanguage(s)')
41
42
43 omdb.set_default('apikey', '906517b3')
44
45 for i in dirs:
46     if id_start != '':
47         if i > id_start:
48             print('tt' + i + '\t', end="", flush=True)
49             doc = omdb.imdbid('tt' + i)
50             # if doc['language'] == 'Hebrew':
51             print(doc['title'] + '\t' +
52                   doc['year'] + '\t' +
53                   doc['language'])
54     else:
55         print('tt' + i + '\t', end="", flush=True)
56         doc = omdb.imdbid('tt' + i)
57         # if doc['language'] == 'Hebrew':
58         print(doc['title'] + '\t' +
59               doc['year'] + '\t' +
60               doc['language'])
```

Appendix 2.3: single_file_extract.py

```
#! /usr/bin/env python3
# -*- coding: utf-8 -*-

import shutil
import os

source = '../OpenSubtitles2018_parsed'
destination = './OpenSubtitles2018_parsed_single'

# Copy the directory tree into a new location
shutil.copytree(source, destination, ignore=shutil.ignore_patterns('*.*'))

# Copy the first file in each folder into the new tree
for dirName, subdirList, fileList in os.walk(source):
    for fname in fileList:
        if fname == '.DS_Store':
            fileList.remove(fname)
    if len(fileList) > 0:
        del fileList[1:]
        src = dirName + '/' + fileList[0]
        dst = destination + dirName[27:] + '/'
        shutil.copy2(src, dst)
```

Appendix 3: List of movies used

6 References

Vita

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam et turpis gravida, lacinia ante sit amet, sollicitudin erat. Aliquam efficitur vehicula leo sed condimentum. Phasellus lobortis eros vitae rutrum egestas. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Donec at urna imperdiet, vulputate orci eu, sollicitudin leo. Donec nec dui sagittis, malesuada erat eget, vulputate tellus. Nam ullamcorper efficitur iaculis. Mauris eu vehicula nibh. In lectus turpis, tempor at felis a, egestas fermentum massa.

Brezina, V., & Gablasova, D. (2015). Is there a core general vocabulary? Introducing the new general service list. *Applied Linguistics*, *36*(1), 1–22. https://doi.org/10.1093/applin/amt018

Brysbaert, M., & New, B. (2009). Moving beyond Kučera and Francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for American English. *Behavior Research Methods*, *41*(4), 977–990. https://doi.org/10.3758/BRM.41.4.977

Coxhead, A. (2000). A new academic word list. *TESOL Quarterly*, *34*(2), 213–238. https://doi.org/10.2307/3587951

Nation, I. (2016). *Making and using word lists for language learning and testing.* Amsterdam: John Benjamins Publishing Company. https://doi.org/10.1075/z.208

Sorell, C. J. (2013). *A study of issues and techniques for creating core vocabulary lists for English as an international language* (Unpublished Dissertation). Victoria University of Wellington, Wellington, New Zealand.