



Louisiana French

An Analysis of the Verb Frequency within 28 interviews in Louisiana French

CS 4624: Multimedia, Hypertext, and Information Access
Final Report

Virginia Polytechnic Institute and State University
Blacksburg, Virginia, 24061
10 May 2019

Instructor: Dr. Edward Fox

Clients: Dr. Katie Carmichael, Dr. Aarnes Gudmestad

Group: Rahul Ramakrishnan, Saharsh Shrivastava, Abhi Oddula, Will Duong, Joe Chu

Table of Contents

Table of Figures.....	2
Table of Tables.....	3
Executive Summary.....	4
1. Introduction.....	5
1.1 Objective.....	5
1.2 Clients.....	6
1.3 Constraints.....	6
1.4 Clarifications.....	7
1.5 Roles.....	7
1.5.1 User Roles.....	8
1.5.2 Project Roles.....	8
1.6 Scope.....	9
1.7 Privacy.....	10
2. Requirements.....	11
3. Design.....	12
4. Implementation.....	13
4.1 Introductory Phase.....	13
4.2 Experimental Phase.....	13
4.3 Phase 1 - TreeTagger.....	14
4.4 Phase 2 - Corrections.....	14
5. Testing/Evaluation/Assessment.....	16
6. Future Plans.....	17
7. Acknowledgements.....	17
8. Manuals.....	18
9. Lessons Learned.....	21

Table of Figures

1. Louisiana French.....	5
2. Dr. Katie Carmichael.....	6
3. Dr. Aarnes Gudmestad.....	6
4. Experimental Phase Scraping Avoir.....	14
5. TreeTagger Results.....	15
6. Homebrew.....	18
7. Terminal 1.....	19
8. Terminal 2.....	20
9. Terminal 3.....	20
10. Script 1.....	20
11. Tagger Results.....	21

Table of Tables

1. Project Roles.....7
2. Milestones for project.....8

Executive Summary

The goal of our project is to assist in the sociolinguistic research of Dr. Carmichael and Dr. Gudmestad. Their project involves the research of Louisiana French, a dialect of the French language, spoken in Louisiana. When we first contacted our clients expressing our interest in this project, they sent us a corpus of interviews conducted by Dr. Carmichael. These transcribed dialogues analyze the dialogue of the interviewees speaking in Louisiana French.

Our group's goal is to help our clients analyze verb frequencies in the interviews given to us. We are initially given a verb bank, and our responsibilities are to find and count each conjugation of each verb in the bank. To find each conjugation, we had to account for both regular and irregular verbs in the bank. Irregular verbs are conjugated differently, and for that reason, it involves a different conjugation process. Our first phase of the project involved getting familiarized with the French language in general and how conjugations of a verb, in French, take place. During this time, we have also decided to use Python to build scripts that would extract data from our corpus.

Though generating Python scripts was original implementation, we have soon realized that using TreeTagger software would be the best and most efficient way to go about the next steps of our project. TreeTagger is a software that can annotate text with part of speech information. Using this software, we are able to calculate the frequency of specific verbs and verb conjugations by parsing the data returned from TreeTagger. TreeTagger uses parameter files for specific languages in order to determine the parts of speech for sentences. Our group is currently using the French parameter file.

In some instances; however, French and Louisiana French can differ. For this reason, we have decided that we need to implement a parameter file specifically for that dialect. Our clients, in this way, can make use of the TreeTagger specifically for their dialect of interest. As an additional requirement, we will create a training corpus for the TreeTagger and turn it into a parameter file so it can assist our client for their future endeavors in researching Louisiana French.

1. Introduction

Our project aims to analyze pronoun choice from the speakers of our interviews. Specifically, we are looking for the frequency of the verbs in our corpus of interviews in order to better analyze the data presented and the Louisiana French language as a whole. Our group is working to identify and count the verb frequencies throughout 28 interview transcripts for our client, Dr. Katie Carmichael and Dr. Aarnes Gudmestad.

1.1 Objective

Our objective is to determine which verbs are used to what frequency in our corpus of interviews. In the aftermath of Hurricane Katrina, our client's research focus shifted to the deeper linguistic changes that occurred in the area due to the storm. Louisiana French, a dying dialect of the French language, originated in Louisiana after French colonials settled in the area. It is important to keep in mind that this dialect is only spoken in Louisiana. We are examining records recorded dialogues from the Pont-Au-Chien Indians of Pointe-Aux-Chenes, Louisiana who speak in this dialect.



Figure 1. Louisiana French

The main goal of this project is to assist Dr. Carmichael and Dr. Gudmestad with their research by automating tasks that can be automated, specifically generating verb counts. Going through the 28 interviews manually and finding verbs would be a painstaking process. Instead our approach focuses on data parsing and parts of speech tagging. Both approaches center around programming and software. With our solution,

we will not only be able to save our clients valuable time, we will also contribute to their overall research in analyzing the dialogue of Louisiana French speakers.

1.2 Clients

Our clients are Dr. Katie Carmichael, a professor from the Virginia Tech Department of English, and Dr. Aarnes Gudmestad, a professor from the Virginia Tech Department of Modern and Classical Languages and Literatures. With their numerous publications in the field of sociolinguistic variations and language, they are our point persons for our Louisiana French project.



Figure 2. Dr. Katie Carmichael



Figure 3. Dr. Aarnes Gudmestad

Dr. Carmichael, a professor in the Virginia Tech English Department, primarily focuses her research on the study of how society and culture influence words and speech patterns. Carmichael received the NSF grant in 2018 to research the linguistic effects of Hurricane Katrina. Though a small component of this research involves the work we have been tasked to complete, there is information that can be analyzed from the data we provide for this project.

1.3 Constraints

With any project, there are limitations involved with both our scope and the time we have to complete the project. As our time is constrained in our semester long project, it is important for us to organize our project so that we have adequate time to rollout our final deliverable.

The final deliverable for our project involves determining specific frequency counts within the overall corpus then entering that information, for each verb coded in a pre-existing spreadsheet, as a separate column. In other words, our main deliverable is

a additional column in the original spreadsheet indicating verb frequency for the verbs included in the dataset.

The data we have to parse from is given to us in transcribed .docx files. The number of files we have to work with can potentially increase if our clients decide to add more data to the corpus. With this in mind, we must consider efficiency when implementing our solution. If our clients decide to add new interviews to the overall corpus, a new verb may be introduced requiring a verb count for. In this case, we would have to consider both current implementations and features that would be useful for the future.

1.4 Clarifications

When coupled with a subject, French verbs require a conjugation, which means you have to tweak the base form of the verb a little bit to make it grammatically correct. The base form of a verb is called the infinitive - this is the verb being conjugated.

The two types of verbs that we are handling for our project are regular verbs and irregular verbs.

Regular verbs are verbs where the conjugation follows a pattern. That is, the french verbs *parler* and *manger* will both be conjugated by dropping the -er at the end of the verb. Since this follows a specific pattern, we can consider these verbs regular.

Irregular verbs are a little more complicated. For irregular verbs, the conjugations do not follow a pattern. The conjugations for these words can look very different than what you would expect; as such, our group has to make sure that we make sure the conjugated verbs we are searching for are correct, especially since some of the most common verbs in spoken French are irregular verbs.

1.5 Roles

To assist us in completing this project, we have decided to take on roles in the making of our frequency counts. We found that each role has a significant impact on the overall project and that some roles are spread out amongst the team so that the entire team can have a strong understanding of the overall project. We have also identified and analyzed our user roles for this project.

1.5.1 User Roles

The roles related to our corpus of interviews involve the interviewers and the interviewees. Both roles benefit from our results. The interviewers consist of Dr. Carmichael and other researchers who are studying pronoun choices from our speakers. Their research involving linguistic change will benefit from the verb counts that we generate from the interviews. The interviewees consist of Pont-Au-Chien Indians, who are well represented in this study. The results of our project can help Dr. Carmichael and Dr. Gudmestad further analyze Louisiana French and better understand the group that they are interviewing. The group that is being interviewed in turn has their language studied and partly preserved through research.

1.5.2 Project Roles

Group Members	Roles
Abhi	Aiding work with scraping tool Scribes meetings
Joe	Aiding work with TreeTagger Tool In charge of python script for TreeTagger
Rahul	In charge of Presentations 3 and 4 Member who will submit to VTechWorks
Saharsh	In charge of Scraping Script
Will	Team Leader In charge of TreeTagger software In charge of presentations 1 and 2

Table 1: Group Roles

1.6 Scope

The Louisiana French project is a semester-long project that follows the Agile software development process. It consists of requirements, design, prototyping, implementation, and testing. We follow the Agile methodology with iterations throughout our project in case requirements or setbacks occur. Our milestones to date are listed as follows.

Date	Milestone
February 8	Client meeting: understanding project requirements
February 26	Presentation 1
March 4	Client and professor meeting: demo of progress and clarification of project
March 6	Completed avoir script Deliverable: Data on how many instances of the verb 'avoir' appear in given interviews. Instances counted based on conjugation, not the infinitive (base) verb.
March 10	Started implementation with TreeTagger
March 26	Presentation 2
April 4	Finished implementation with TreeTagger
April 8	Demo of progress with Dr. Gudmestad
April 11	Presentation 3
April 12	Putting together final deliverable
April 30	Final presentation and VTURCS
May 4	Final deliverable sent to client
May 7	Approval of final deliverable by client

Table 2: Milestones for project

1.7 Privacy

One aspect we would like to discuss is that the data we have received (our corpus of interviews) is IRB-protected. IRB-protected data means that the data is collected from human subject and thus the storage/storing of it needs to be done in methods that protect against it becoming publically available. For this reason, the data or information about the interviewees may not be released. Though we would be happy to share the results of our project, Images of our data may not be placed on this document due to privacy concerns. Our clients are trusting that we keep this data secure, and we have to ensure that this trust is kept.

2. Requirements

Our requirements were created after many talks with our clients and our professor. Through careful consideration, we have found a list of requirements that we can use to get our project started.

Our base requirement is to find the total frequency of each verb, which will be further explained in the implementation section. These verbs have been conjugated to their appropriate parts of speech within the interviews, therefore requiring us to understand the stem of the verb in order to further examine where the verbs have been used. That being said, we would also need to figure out how to take in account for irregular verbs. Whereas a regular verb like 'parler' with the je pronoun will have the -e ending and be conjugated into 'je parle', an irregular verb like 'aller' will be conjugated into 'je vais'. This would lead us into developing additional algorithms to deal with these verbs. Finally, the client has asked us to record any metadata that we encounter in the processes of meeting our base requirement. This will help them have more information about their dataset, allowing them to analyze it in different ways. As a result, we will be keeping track and presenting any information we find of use to our clients.

Another requirement that we have recently taking up is fixing the corpus for people who decide to pursue this project in the future. The corpus had some problems, such as accent placements and some stuttering done on the part of the speaker. Because of this, we decided that another requirement for ourselves was to create a copy of the corpus with changes, and another spreadsheet with word in question, where it is located, and if we changed it or not. Even if we didn't manually change the word, we wanted to make sure that the client could go in and (potentially) fix the word.

3. Design

Our client provided us with the power to design and implement our solution as long as they received their deliverable. Thus, the design choices were made by the team and presented to the client for approval. Given a list of requirements, we created milestones where we would complete each part of the list by a certain deadline.

We decided that our primary design utilizes TreeTagger and Python. We wanted to use software and design a script that is easy to use, accessible, and powerful as well as convenient for our client to use. Similarly, we found a product that had a lot of information online, and the developer of TreeTagger was more than happy to help us as we tried to work with the software.

On top of that, we knew that we would have to use Python scripts to extract data. When we get the results from the TreeTagger, we can use a Python script to parse that data into a dictionary of verb conjugations matched with their frequency.

Lastly, all of the data we acquire will be inserted into a spreadsheet that will be easily presentable to the clients.

4. Implementation

4.1 Introductory Phase

Because none of our group speaks French, it took some time to figure out how conjugations work in French. After our first meeting with our client, we realized that, first and foremost, we had to figure out the general structure of Louisiana French sentences. Afterwards, we figured out how verbs and verb conjugations fit with Louisiana French sentences.

This introductory phase led us towards the Experimental Phase, as we had a general idea about how French verb conjugations worked.

4.2 Experimental Phase

When we first came up with our requirements, we knew we had to create a python script to parse through each of the files. However, we didn't anticipate how many verbs we truly needed to search for, on top of the six conjugations of each that could be found in the corpus. Similarly, we didn't anticipate how long each of the interviews actually was.

Initially, we created a small python script that would take manual entry of a conjugated verb as a parameter. With this parameter, the script would search through the entire corpus and both the total number of instances and the locations of each instance of that specific verb conjugation. This was a great starting point for us. However, we noticed that this method was not efficient. Overall, there were over 80 verbs in the corpus, each with multiple verb conjugations. As such, it would have taken a lot of time and computing power to scrape the entire corpus.

```

[('(j)'ai', 1)]
[('j'ai', 1)]
[('j'ai', 1)]
[('j'ai', 1)]
[('jh'ai', 3)]
[('jh'ai', 1)]
[('jh'ai', 1)]
[('jh'ai', 1)]
[('j'ai', 4)]
[('j'ai', 1), ('(j'ai)', 1)]
[('j'ai', 1)]
[('l'ai', 1)]
[('j'ai', 1)]
[('j'ai', 1)]
[('j'ai', 1)]

```

Figure 4. Experimental Phase Scraping Avoir

Above is an example of what one of our Experimental Phase runs looked like. This example uses the first person form of the verb *avoir*. Despite this being a correct result, it would have taken too long to find and may not have been as reliable in the long run.

Through a combination of recognizing our struggle and meeting with our instructor and client, we realized that we had to drastically change the approach of our implementation. Dr. Fox provided us with a suggestion to use a parts of speech tagging software called TreeTagger. This leads us into Phase 1 - TreeTagger.

4.3 Phase 1 - TreeTagger

This was our largest phase, and what we spent the bulk of the time implementing. TreeTagger is a software that could return a frequency of specific verbs and verb conjugations, based on the French language structure. It 'tags' items based on the likelihood of whether it is a verb or not. This can also include english words that are used in French speech - this was the case pretty often, as Louisiana French speakers still used some english within their speech. Either way, TreeTagger is able to tag these words as verbs as well, because it still fits into where a French verb would occur.

TreeTagger utilizes parameter files for specific languages. We are currently using the French parameter file. This parameter file has data that allows it to build a tree based on the parts of speech. Even though we are only looking for verb data, we can find a lot

more data, such as which words were tagged as nouns, adverbs, and more. Because of this, TreeTagger is extremely valuable for our clients, as it can be used on further research surrounding the Louisiana French Dialect.

Once TreeTagger tags each word in each sentence in a manner as explained above, we have to parse the given data. Using a small python script, we are able to parse this data. Below, we have figures for the TreeTagger results. A dictionary is created for each unique verb found; as more instances are found, the frequency is incremented.

```
{'être': 197, 'avoir': 314, 'parler': 37, 'préférer': 1, 'apprendre': 5, 'j'h'étais': 2, 'faire': 40, 'punissé': 1, 'punir': 1, 'venir': 16, 'travailler': 2, 'aller': 23, 'j'h'ai': 16, 'ouvrir': 2, 'finir': 1, 'pouvoir': 20, 'rester': 16, 'grouiller': 3, 'ic': 1, 'icitte': 3, 'sortir': 7, 'tremper': 1, 'lever': 4, 'protéger': 2, 'arriérer': 2, 'protectait': 1, 'protecte': 1, 'élever': 1, 'perdre': 1, 'j'h'avais': 2, 'mixer': 1, 'comprendre': 6, 'naître': 2, 'montrer': 5, 'célébrer': 1, 'vous-autres': 4, 'manger': 5, 'cuire': 11, 'Christissime': 1, 'ouvrer|ouvrir': 2, 'amener': 7, 'aimer': 13, 'mener': 1, 'connaître': 23, 'bouillir': 2, 'chauffer': 1, 'user': 3, 'Geneva': 1, 'f': 1, 'soigner': 5, 'prendre': 3, 'arrêter': 4, 'j'h'essaye': 1, 'mettre': 9, 'signer': 1, 'voir': 8, 'falloir': 3, 'rencontrer': 4, 'croire': 10, 'gagner': 1, 'pêcher': 1, 'commencer': 2, 'oublier': 2, 'rappeler': 4, 'lire': 5, 'écrire': 4, 'étudier': 1, 'devoir': 4, 'penser': 1, 'dire': 13, 'garder': 2, 'baigner': 1, 'e': 2, 't': 2, 'changer': 3, 'gâter': 1, 'voyager': 1, 'j'h'allais': 1, 'écouter': 1, 'agacer': 1, 'j'h'aime': 1, 'j'h'aimais': 1, 'and': 1, 'chanter': 1, 'the': 4, 'danser': 7, 'haler': 1, 'plancher': 1, 'marier': 1, 'brûler': 2, 'lire|liser': 1, 'asseoir': 2, 'dormir': 1, 'boucaner': 1, 'boire': 1, 'marcher': 2, 'guetter': 3, 'jouer': 2, 'dépendre': 1, 'menir': 1, 'commenter': 1, 'her': 1, 'savoir': 1, 'vouloir': 5, 'j'h'a': 1, 'regarder': 1, 'cez': 1, 'suivre|être': 1, 'entendre': 1, 'appeler': 6, 'bourrer': 2, 'plumer': 1, 'ramasser': 1, 'laver': 2, 'passer': 2, 'sécher': 1, 'chésér': 1, 'coucher': 1, 'couvrir': 1, 'piquer': 1, 'amuser': 1, 'bâtir': 1, 'é': 1, 'raconter': 3, 'souvenir': 1, 'x': 1, 'exister': 1, 'jongler': 4, 'n-avait': 2, 'nous-autres': 1, 'enterrer': 1, 'suivre': 1, 'arriver': 2, 'disparaître': 1, 'quitter': 2, 'tait': 2, 'érailler': 2, 'railler': 1, 'demander': 2, 'donner': 1}
```

Figure 5. TreeTagger Results

Some of the problems we encountered revolved around the corpus itself. All of the apostrophes (') were actually encoded as the grave symbol (`). Naturally, this was throwing off the results, as the some of the French words which utilized apostrophes were not properly being translated. However, this problem opened a new phase for the project, called the corrections phase.

4.4 Phase 2 - Corrections

Because of the size of the corpus, this phase took up the rest of the semester. As we were using the Tree Tagger, we noticed that certain words were not tagged correctly. We also noticed that this was occurring with many types of words, such as nouns, pronouns, adjectives, and adverbs. As such, we started investigating why there were bugs within the result data.

As explained above, one of the major bugs we found was that the apostrophe was being replaced with the grave within the corpus; this was fairly easy to fix. However, the rest of the bugs required some more thought, and would have required a better knowledge of French to fix. As such, we manually went through each of our results, and made sure that the words we found within the corpus were actually verbs or derived

from verbs. Otherwise, we noted the exact word that is wrong and where it is located in the corpus. If we knew that the word was wrong due to an accent, we noted down the word and location, and changed the word in a copy of the corpus, which we are sending back to our clients.

This process, although monotonous, was very important to confirm that validation of our results. Since these results had no possible way of being automated, we had to pull the trigger on making this a manual process.

This phase has been completed, and our client has accepted the data that we have provided.

5. Testing/Evaluation/Assessment

As implementation of TreeTagger continued, our group decided that we needed a way to verify that the results are actually correct. Though TreeTagger has been used by others before, we must still recognize that errors can occur, especially since Louisiana French can differ from regular French. As such, we decided to scrape WordReference.com for French conjugations of the verbs that exist within the corpus.

However, we realized that this plan wouldn't work as we expected. Because of how French conjugations work, there are certain words within the corpus that are repeated, but they don't necessarily mean the same thing. Some conjugations may be similar to other words in French. As such, this didn't provide us the accuracy that we initially wanted.

Our next approach, however, was to manually check each of the instances that we found. Though this was arduous, we realized that we didn't necessarily have an enormous corpus; we could evenly distribute the load amongst the 5 members and make the work much more manageable for the group. This would also ensure the accuracy and validity, since there are very few times that we could really cause an error. Naturally, we are just trying to see if the data's verbs do occur that many times; as such, finding the verbs within the corpus would not be a problem at all.

As of now, the testing process has been complete and the data has been approved by our clients.

6. Future Plans

Though we had initially hoped to make a parameter file, we had to change our specifications for the semester to ensure that we stayed focus and the validity of our actual data was perfect. As such, we decided to remove the requirement we made for ourselves to create a parameter file. However, this doesn't mean that a group in the future couldn't do it.

Our initial idea was to create a parameter file that could be used by TreeTagger. This parameter file would be made *specifically* for Louisiana French. Though we don't know the process to do this yet, the contact we made with the creator of TreeTagger made it seem as though as long as we had a corpus of sentences that we could provide, it wouldn't be a fairly difficult process. Nonetheless, we feel that creating a TreeTagger parameter file for Louisiana French would be extremely helpful to the clients in the future as the project pans out to more types of words or sentence structures.

On top of this, we believe it is important that someone go in and fix the corpus with all of the proper words. As we were going through the TreeTagger data, it became apparent that certain words were not accented properly, spelled properly, or were shown to be inaudible. As such, it is important that someone edit the files with fixes. Though we have started this process, our knowledge in French limited how far we could go for this requirement. I suggest that someone do this in the future, and that at least one member of the group should be somewhat fluent in French.

7. Acknowledgements

We would like to thank our clients, Dr. Katie Carmichael and Dr. Aarnes Gudmestad of the Department of English and the Department of Modern and Classical Languages and Literatures at Virginia Tech. They helped our team gain a better understanding of the French language for this project and offered us resources to better understand how verb conjugation works. We also appreciate the time they took to meet with us in order to clarify our doubts and help this project run more smoothly. They can be contacted at katcarm@vt.edu and agudmest@vt.edu respectively.

We would also like to thank Dr. Edward Fox, the professor of our course for his input in streamlining our project and offering us very helpful suggestions. Additionally, we also thank our classmates for their assistance in giving us constructive criticism on our presentations.

8. Manuals

Our clients are macOS users, therefore this user manual will be for macOS only. Following this user manual will lead to a proper installation of Python 3 as well as the TreeTagger software with the correct dependencies. There is no implemented user-interface or application that the TreeTagger runs on. It will be run directly from the terminal.

8.1 - Python 3

Explanation: Python is a high-level programming language that allows for object-oriented design as well as scripting. Python is often used to automate tasks and is a common programming language because of the amount of libraries available to assist with projects. To make usage of TreeTagger easier, there is a Python wrapper for the TreeTagger function calls. Following these instructions will lead to a full installation of the Python 3 dependencies on your device.

Note:

If you are a developer with dependency installation experience: Install Python 3 and skip to section 2.

1.1 Installing Homebrew

Explanation: Homebrew is an open-source software package management system that makes installation of software as simple as one-line homebrew calls. Installing Homebrew will make the Python 3 installation almost effortless.

1. Go to <https://brew.sh/> and you will see a website that looks like this.

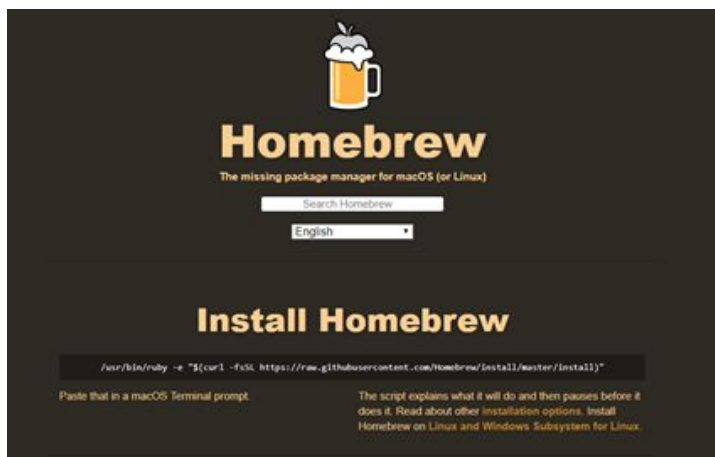


Figure 6. Homebrew

2. There will be a line where you can copy a command into your terminal. If you copy “/usr/bin/ruby -e “\$(curl -fsSL <https://raw.githubusercontent.com/Homebrew/install/master/install>)” into your terminal, it will install Homebrew.

1.2 Installing Python with Homebrew

Once Homebrew finishes installing. Type “brew install python3” and Homebrew will install python3 for you.

8.2 – TreeTagger

Explanation: TreeTagger is a part-of-speech tagging software that is available for multiple languages. Installing TreeTagger and the dependencies will allow you to tag your text and provide part-of-speech information for text.

Note: TreeTagger is a tool developed for research and teaching purposes. If using commercially, the user must contact the developer with a commercial license. Any distribution of this software without permission or commercially is unethical.

2.1 Installing TreeTagger

1. First go to the website where the TreeTagger files are <http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/>
2. Scroll down to installation and install the tagger package for the system. (Pick macOS)
3. Save this into a directory where you will be operating.
4. Install the tagging scripts, bashfile, and parameter files for the desired languages into the same directory.
5. Open your terminal and access the where you have downloaded the files by typing: ls

```
Williams-MacBook-Pro-3:~ Will$ ls
?
?.pub
Applications
Creative Cloud Files
Desktop
altar.txt
clone
eclipse
eclipse-workspace
git
```

Figure 7. Terminal 1

A list of your directories should appear.

6. Type cd “directory name” and enter to navigate to the directory.

```
[Williams-MacBook-Pro-3:~ Will$ cd Desktop/  
[Williams-MacBook-Pro-3:Desktop Will$ cd TreeTagger/  
Williams-MacBook-Pro-3:TreeTagger Will$ █
```

Figure 8. Terminal 2

Here William has navigated to his desktop by typing cd Desktop and then to his TreeTagger directory by typing cd TreeTagger

7. Then William runs Python by typing python3 in the terminal and then types the following...

```
[Williams-MacBook-Pro-3:TreeTagger Will$ python3  
Python 3.6.2 (v3.6.2:5fd33b5926, Jul 16 2017, 20:11:06)  
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin  
Type "help", "copyright", "credits" or "license" for more information.  
>>> import treetaggerwrapper  
>>> tagger = treetaggerwrapper.TreeTagger(TAGLANG='fr')  
>>> tags = tagger.tag_text("ENTER YOUR TEXT HERE")  
>>> print(tags)  
['ENTER\tVER:infi\tenter', 'YOUR\tNAM\tYOUR', 'TEXT\tNAM\tTEXT', 'HERE\tNAM\tHER  
E']  
>>> █
```

Figure 9. Terminal 3

```
Import treetaggerwrapper  
tagger = treetaggerwrapper.TreeTagger(TAGLANG='fr')  
tagger.tag_text("ENTER YOUR TEXT HERE")  
print(tags)
```

The user will replace the “ENTER YOUR TEXT HERE” with what text they want to tag with the quotation marks. Then it will print out the tagged text.

8.3 – Running the Script

1.To run the script provided. The user will download the python script into the same directory as the TreeTagger folder and then the user will have to modify the location of the text file.

```
file_location = "Interviews/interview0SF10.txt"
```

Figure 10. Script 1

If the current text file is in the same directory then that will be renamed to
“./textfile_name.txt”

2. In the terminal as the same location as the TreeTagger directory. The user will
type “python3 python_file_name.py”

3. Results will be displayed like below.

```
{'être': 197, 'avoir': 314, 'parler': 37, 'préférer': 1, 'apprendre': 5, 'j'h'étais': 2, 'faire': 40, 'punissé': 1, 'punir': 1, 'venir': 16, 'travailler': 2, 'aller': 23, 'j'h'ai': 16, 'ouvrir': 2, 'finir': 1, 'pouvoir': 20, 'rester': 16, 'grouiller': 3, 'ic': 1, 'icitte': 3, 'sortir': 7, 'tremper': 1, 'lever': 4, 'protéger': 2, 'arriérer': 2, 'protectait': 1, 'protecte': 1, 'élever': 1, 'perdre': 1, 'j'h'avais': 2, 'mixer': 1, 'comprendre': 6, 'naître': 2, 'montrer': 5, 'célébrer': 1, 'vous-autres': 4, 'manger': 5, 'cuire': 11, 'Chrismisse': 1, 'ouvrer|ouvrir': 2, 'amener': 7, 'aimer': 13, 'mener': 1, 'connaître': 23, 'bouillir': 2, 'chauffer': 1, 'user': 3, 'Geneva': 1, 'f': 1, 'soigner': 5, 'prendre': 3, 'arrêter': 4, 'j'h'essaye': 1, 'mettre': 9, 'signer': 1, 'voir': 8, 'falloir': 3, 'rencontrer': 4, 'croire': 10, 'gagner': 1, 'pêcher': 1, 'commencer': 2, 'oublier': 2, 'rappeler': 4, 'lire': 5, 'écrire': 4, 'étudier': 1, 'devoir': 4, 'penser': 1, 'dire': 13, 'garder': 2, 'baigner': 1, 'e': 2, 't': 2, 'changer': 3, 'gâter': 1, 'voyager': 1, 'j'h'allais': 1, 'écouter': 1, 'agacer': 1, 'j'h'aimais': 1, 'j'h'aimais': 1, 'and': 1, 'chanter': 1, 'the': 4, 'danser': 7, 'haler': 1, 'plancher': 1, 'marier': 1, 'brûler': 2, 'lire|liser': 1, 'asseoir': 2, 'dormir': 1, 'boucaner': 1, 'boire': 1, 'marcher': 2, 'guetter': 3, 'jouer': 2, 'dépendre': 1, 'menir': 1, 'commenter': 1, 'her': 1, 'savoir': 1, 'vouloir': 5, 'j'h'a': 1, 'regarder': 1, 'cez': 1, 'suivre|être': 1, 'entendre': 1, 'appeler': 6, 'bourrer': 2, 'plumer': 1, 'ramasser': 1, 'laver': 2, 'passer': 2, 'sécher': 1, 'chésier': 1, 'coucher': 1, 'couvrir': 1, 'piquer': 1, 'amuser': 1, 'bâtir': 1, 'é': 1, 'raconter': 3, 'souvenir': 1, 'x': 1, 'exister': 1, 'jongler': 4, 'n-avait': 2, 'nous-autres': 1, 'enterrer': 1, 'suivre': 1, 'arriver': 2, 'disparaître': 1, 'quitter': 2, 'tait': 2, 'érailler': 2, 'railler': 1, 'demander': 2, 'donner': 1}
```

Figure 11. Tagger Results

Because of the the scripting we used was very basic, the developer manual was deemed unnecessary. The python script will parse through each line of code and find the words that are tagged as verbs, and take the **infinitive** form of the word.

9. Lessons Learned

There are many lessons to be learned from a project of this magnitude. As we progress through the various phases, we will find more takeaways.

One of our biggest problems was when we underestimated the amount of time the scripting could take. Making an algorithm is fine; however, it is also important to ensure that it is as efficient as possible. When the client uses the application, it is imperative that we give a program that works well, and gives them timely results.

Luckily, we realized we were stuck around that time Dr. Fox wanted to meet us and our client; as such, we were able to hash out a potential solution. This is our Phase 1 solution - we used TreeTagger to find verb frequencies, a solution we investigated after a recommendation from Dr. Fox.

Another major problem was that the project revolves around a French dialect. Since none of us had much experience with French, we had to figure out how conjugations worked within a French sentence, and how language structure how subject-verb combinations would cause different conjugations.

We decided to meet as a team with the client. They were able to explain a lot of the basics of French verb conjugations and provided us with websites that we could use to find conjugations to most french verbs. From there, we took it upon ourselves to figure out how conjugations worked, and were gradually able to develop a deeper understanding of French verb conjugation.

Another major problem was after TreeTagger was run. As aforementioned, there were some problems with the results of our project. Though we were able to extract all of the verbs we needed, we still had some other parts of speech showing up in our verb frequency list.

We decided to investigate this, and noticed that the corpus needed to be cleaned. There were certain words, such as the letter 'x' which we found out was just a placeholder for an inaudible word. Similarly, we were able to see that certain symbols and accents were not applied properly. This posed a big problem for us because none of us had much experience in French.

We learned to be very adaptable throughout this project. We all had never done anything like this, as this was in a completely different language completely. Because of this, we were made to be flexible throughout the project. With helpful tips from our clients, peer reviews, and Dr. Fox, we were able to provide solutions despite some of the problems that we had faced.

References

1. "Aarnes Gudmestad." *College of Liberal Arts and Human Sciences | Virginia Tech*, liberalarts.vt.edu/departments-and-schools/department-of-modern-and-classical-languages-and-literatures/faculty/aarnes-gudmestad.html. Last referenced: May 2019.
2. Carmichael, Katie, and Aarnes Gudmestad. *Language Death and Subject Expression: First-Person-Singular Subjects in a Declining Dialect of Louisiana French*. May 2018. Last referenced: May 2019.
3. "Katie Carmichael." *College of Liberal Arts and Human Sciences | Virginia Tech*, liberalarts.vt.edu/departments-and-schools/department-of-english/faculty/katie-carmichael.html. Last referenced: May 2019.
4. Schmid, Helmut. "TreeTagger - a Part-of-Speech Tagger for Many Languages." *TreeTagger*, www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/. Last referenced: May 2019.