**Language Learning Aid**

**Table of Contents**

Intro: Introduction

Chapter 1: Development

Chapter 2: Outcome

Chapter 3 : Reflection

By Justin Williams

**Introduction**

I want to create a program that aids in language acquisition. There are three primary issues with learning/acquiring a new language. Firstly, is the consistency of actually engaging with learning. Secondly, even after learning a language, the most common way of learning is ineffective for the vast majority of people; this is because most courses focus on acquiring new vocabulary, then grammar, and while grammar tends to stay throughout the years, vocabulary is lost fairly quickly after it is not reinforced constantly. Lastly, and most prominently, language learning is boring. Traditional textbook methods make learning a chore, so applications such as Duolingo, which “gamify” the learning process are more sought after. As such, one way I wanted to solve this problem is by engaging people in entertainment while they learn a language. Furthermore, there is also the benefit of allowing the user to more easily understand tone, sentence structure, and flow more so than the generic ‘“My backpack is red.”’ structure of traditional language learning outlets.

With the globalization of the world, the internet, and constant access to all the world’s information, the importance of understanding a second language has only increased. Within the social and economic spheres knowing a second language has been shown to increase income. However, many tribulations come along with learning a language, and even with the wealth of information, many still struggle and work inefficiently toward acquiring a second language. By providing learners with authentic language input and opportunities for interaction, media can help learners bolster their language skills and apply what they learned in the real world. As such, the focus of this senior project would be aiding in second language acquisition by creating a learning aid for foreign media. The design and functionality of the said application will be informed by studies on language acquisition and the most effective means of doing so.

Krashen's theory of second language acquisition emphasizes the importance of comprehensible input in language learning. Krashen namely focuses on the theory and methodology of acquiring a second language, and how important complex input, and context-oriented learning are to acquisition. “The final part of the input hypothesis states that speaking fluency cannot be taught directly…The best way, and perhaps the only way, to teach speaking, according to this view, is simply to provide comprehensible input. Early speech will come when the acquirer feels "ready"; this state of readiness arrives at somewhat different times for different people, however…Accuracy develops over time as the acquirer hears and understands more input.”(Krashen 22) Media, such as television shows and movies, can provide learners with this exposure to authentic language input, this comprehensible input. With the increase of streaming services and online videos implementing an application to integrate itself with videos and pull out the most important language information to ease the watchers' learning experience. By listening to and watching media in the target language, learners can improve their listening and comprehension skills and ability to produce language.

Negotiation of meaning, as discussed by Pica, can be facilitated by media that provide opportunities for interaction. For example, learners can engage in discussions with others about a movie or television show in a language, using that language to clarify misunderstandings and negotiate meaning. Constant interaction within a language bolsters one’s understanding of meaning, conotation, and language “ideaology”. This type of interaction can help learners develop their language skills and will directly apply to their use of said language.

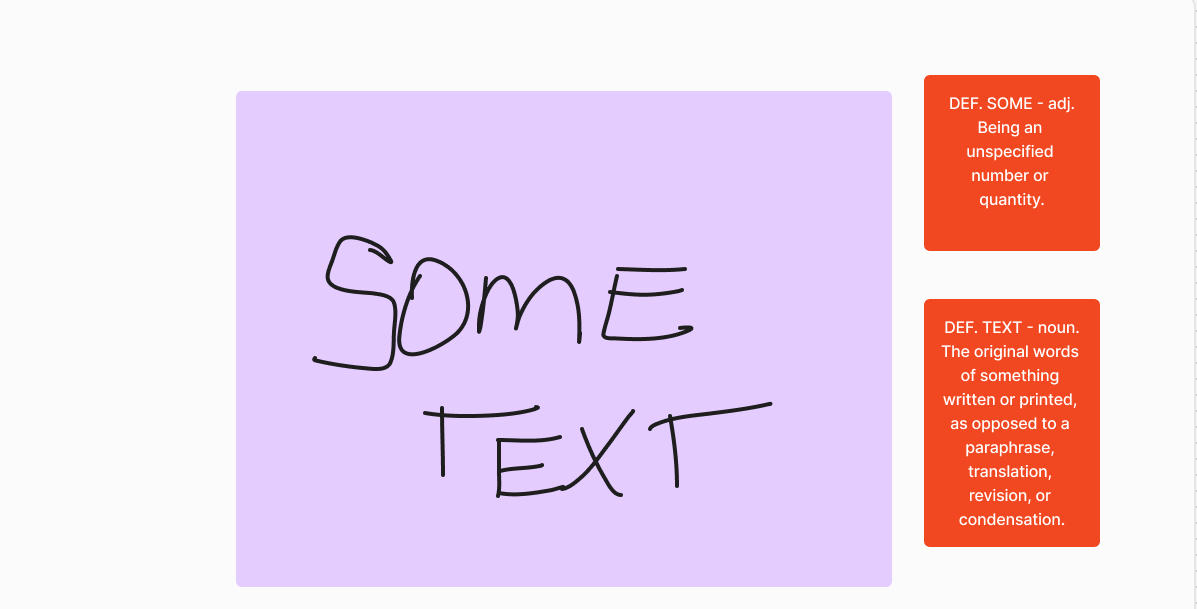
Doughty and Long's Handbook of Second Language Acquisition is a comprehensive and influential collection of articles that provides an overview of the major theories and research findings in the field of second language acquisition. The handbook covers a wide range of topics, including language input, language output, individual differences, social factors, and the role of instruction. It does not directly address the use of media in language learning, however it provides valuable insights to inform the design of a language learning aid. The handbook discusses the importance of both interaction with other learners and interaction with native speakers of the target language. “ Carroll (1967) reported that foreign language majors in American universities who were able to spend their junior years abroad in the country where the language was spoken performed better on the FSI rest of listening comprehension than those who had only spent a summer abroad. The summer travelers, in turn, outperformed those who had never been abroad. These clear results probably reflect the fact that such students, most likely, sought out interaction, and thus comprehensible input …”(Long 2005) A language learning aid for media can facilitate interaction by providing learners with opportunities to discuss media texts with other learners and native speakers of the target language. This can help learners develop their communication skills and increase their confidence in using the language in real-world situations.

Pica's article provides valuable insights into the role of negotiation in second language learning. Negotiation refers to the interactive process of clarifying meaning between speakers of different languages, and Pica argues that it plays an important role in language acquisition. One key insight from Pica's article is that negotiation can lead to a more accurate and comprehensive comprehension of the target language. When learners engage in negotiation, they are forced to ask for clarification, confirm understanding, and negotiate meaning with their interlocutors. This can help learners develop their listening and speaking skills, as well as their ability to comprehend authentic language input. A language aid application can incorporate negotiation activities by providing learners with opportunities to engage in discussions with other learners or native speakers about the media content. For example, learners could be prompted to discuss their interpretation of a scene, clarify the meaning of a word or phrase, or negotiate the meaning of a particular sentence.

Creating an aid for language acquisition will be beneficial for countless reasons. To make sure the application is as effective and efficient as possible these articles will be what informs the design and functionality of said application. By incorporating negotiating activities, and comprehensible input, I believe an application for Second language acquisition using foreign media has the potential to succeed greatly. Taking in the lessons and data taught from these articles would be paramount to creating something effective.

**Chapter 1:**

First before beginning this project, the implementation had to be considered, that is to say, what mediums will it support, what programming language will be used, and basic fundamental functionalities that are core to the intent of the application. The initial design was based upon Javascript, this is because many similar programs have already been tested for singular language support as Chrome extensions. JavaScript also boasts many features that link web frameworks to the backend code. With this in mind, learning JavaScript was the first logical step in this process. To learn JavaScript adequately to create a Chrome extension, I worked on several preliminary projects and worked with my professor to learn the basics. I created a rudimentary Chrome extension, and then I encountered a problem. One of the primary goals of creating this application was functionality offline, and functionality outside of browser space(DOM); the application must be able to function with client-side (on the user’s computer) applications. A Chrome extension would not fulfill this requirement, although the idea could still be supplementary it could never fulfill my primary goals. That encouraged the switch to the Python programming language. Python while slower, and more clunky when integrating with web frameworks, is a much more familiar language.



This rudimentary mockup illustrates the idea in mind. Functionally it would work the same as the dictionary lookup implemented on all modern devices, however the caveat is that it could work with image files also.

After the base language had been decided, design implementation needed to be considered, that is to say how things would exactly function. Major considerations were taken regarding how words would be taken from images and text, and how the relevant definitions would be distributed to the user. This functionality is fundamental to how the application would function, so it was the first logical step in the process. Thankfully, there are many optical character recognition modules and programs for every programming language, from free options to more scalable options like Google’s OCR. For this project, I decided upon Tesseract, because of prior familiarity with its functionality and its free-use policy. The most pressing concern then was the dictionary portion of the project. Design-wise, the basis of this idea came from another similar program rikaikun, which is a browser-based, textual-based character to dictionary software. The application allows for real-time lookups of Japanese characters, giving the English definition as well as the pronunciation, it also supports flashcard creation. The intent was to create this general framework but support multiple languages and the capture of textual data not present in the dom, that is text that is not selectable(i.e. Text in an image). Furthermore, I wanted to support offline functionality and the ability to capture things that are not just present on a browser, for example, those in a dedicated reading executable/application.

When handling the OCR functionality of the application there were many things to account for. Firstly, we must consider what Tesseract’s models were trained upon, so that accurate extrations can be retrieved, the “state” of those models should be replicated. Secondly, we must think about scale. Lastly, is accuracy, how accurate should we be, and how to confirm this accuracy. Research dictated that the best way to retrieve accuracy for OCR programs, Tesseract included, was by cleaning an image, taking out all color, inverting, blurring and then cleaning the blur of an image. The accuracy of just this was promising. For larger, more clear images the cleanup produced minimal errors. The predominant issue was with smaller, more pixelated or fuzzy images, the prior methods would clean the words as if they were part of the background, or make things entirely illegible. That is where scaling comes into play. The logical is thus: If an image is below a certain amount of pixels the algorithm will scale the image up to a certain pixel scale. The reason you must specify that the image is small enough is because scaling a larger image down, would result in a more pixelated and illegible image.

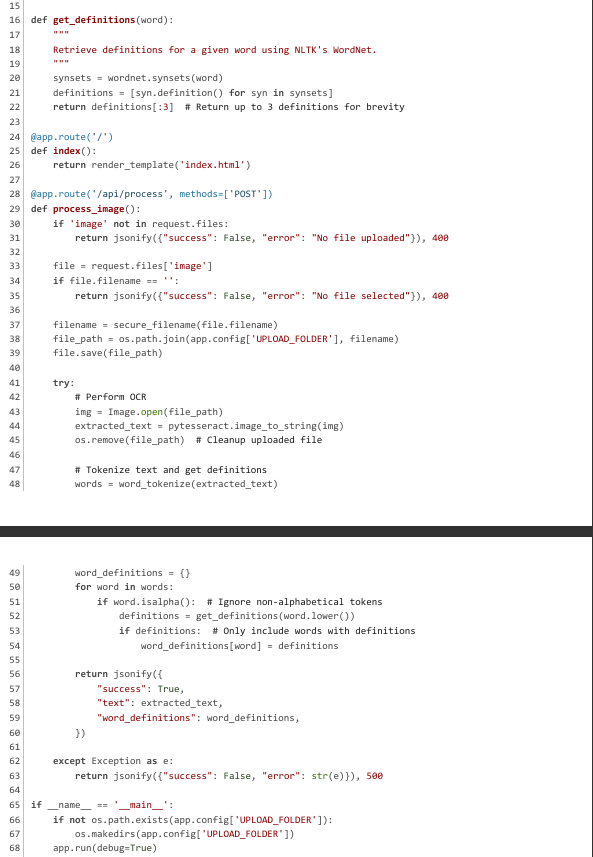
For the dictionary implementation, there are several avenues of progression here: translation of the extracted words and dictionary lookup of a downloaded English dictionary, download of select language to English dictionaries, or download of select language dictionaries. For the intent of this project, the second option would be the most apt, as it allows for the easiest and most direct lookup of terms, without accessing the internet. The issue is however, these dictionaries are not readily downloadable, or they are not in readable formatting. Thus, another part of the process is creating these dictionaries. To do so, a separate application is created that extracts this data from existing online libraries and transfers them into CSV format. After this process, simply form an application with the parameters of a language, that creates a hashmap object, more specifically a data type that equates words to their definitions allowing for timely lookup.

**Chapter 2:**

The scope of the project is vast, so for the purpose of deliverance, I decided to limit the project to present a functional demo of my intent. The demo is using simple python and flask code to run a local web client that allows you to upload and image and the backend code will run on that image. The text is taken from that image using TesseractOCR.

[Demo](https://youtu.be/1XVXVZQ501o)

There are many versions of this code that I have worked with during the process of this project. The keystone functionalities were thus: OCR functionality so that the application could function with any text, not only that which could be selected over (ie. image files), offline functionality, and potential for growth with multiple languages. As it stands the demo only fufills one of said goals and that is the scope that was settled upon.



The code snippet shows the functions for getting a definition, processing the downloaded image, and the basic flask wrapper/functionality. The frontend for the client was mostly HTML and CSS for styling, some javascript was used when handling requests and posts, but not much beyond that. The get\_definition function simply uses NTLKs wordnet package to retrieve definitions given a word, the first 3 definitions are then output. The index function is a wrapper for flask that renders the HTML code. Proccess\_image is a function that gets a json request from the frontend, in the case an image file, saves said file, uses pytesseract to extract text from that file, and then tokenizes the words into a format NTLK works better with. The file is deleted on resolution.

This demo while clunky does showcase what functionality might look like on the application on a micro-scale. There is another version of this application entirely written in tkinkter that uses more complex algorithms for image cleaning and retrieving text. The reasoning behind moving away from that is that it was too clunky and it could not be expanded upon easily. Tkinkter, python’s package for purely python GUIs is limited in function and form and does not have much support beyond itself. For these reasons, a lot of the work that went into that version of the application was scrapped in favor of a more web-oriented application, that used more supported packages.

There were a lot of issues encountered when attempting to create this application, primary among them was the inaccessibility of open-source x-y language dictionaries. Multitrans is one of the better options in this field, however, there was no clear structure to make api calls from their website, and so to use it was a far greater challenge than the scope of this project. The idea persists however to use Multitrans, but first I wish to see if I could create an API to make requests from that website more efficiently. There exists other supported dictionaries however none boast the functionality of one to one “word” to “target language definition” like Multitrans does, so an inefficient workaround would be necessary; such as translating the word first then getting the definition. This however goes against the philosophy I had when creating the application which is to simply retrieve English definitions of foreign words efficiently. Translation falls outside of this scope because it lessens learning outcomes by distributing the visual recognition facet of memory. To preserve the feeling of actually learning rather than simply creating a translation machine was the overall goal of the application.

Ultimately, the application changed in scope many times across development and will likely change many more to come as expansions continue. Looking at the bigger idea behind the Language Learning Aid, a format such as Django(a python web-application framework) would be a better media for the kind of content I want in this application. Namely, user-specific information, flashcard creation, and personalized dictionaries, all of this works more efficiently under the Django framework and as such would need to be ported to it. Furthermore, more sophisticated tokenization and splitting would likely be necessary as the project expands beyond a basic level as NTLK is not infallible in its processing of words. In the future there would likely be entire portions of applications purely dedicated towards specific languages, that is specific lemmatization methods, tokenization methods, and dictionaries for each language. This further favors Django because it has integration with SQL databases and allows you to access them seamlessly. As it stands the application has much more development before it becomes practical, but the initial ideas and research have proven invaluable in understanding what is necessary for further development.

Work Cited

*The Handbook of Second Language Acquisition*. Blackwell, 2003.

Higgs, Theodore V., and Stephen D. Krashen. “Principles and Practice in Second Language Acquisition.” *The Modern Language Journal*, vol. 67, no. 2, 1983, p. 168, https://doi.org/10.2307/328293.

Pica, Teresa. “Research on Negotiation: What Does It Reveal about Second-Language Learning Conditions, Processes, and Outcomes?” *Language Learning*, vol. 44, no. 3, 1994, pp. 493–527, https://doi.org/10.1111/j.1467-1770.1994.tb01115.x.