A CONTEXT-AWARE VOCABULARY MANAGEMENT AND READING ASSISTANCE SYSTEM USING MACHINE LEARNING AND NATURAL LANGUAGE PROCESSING

Zhanhao Cao[1] and Yu Sun[2]

[1]Troy High School, 2200 Dorothy Ln, Fullerton, CA 92831
[2]California State Polytechnic University, Pomona,
CA, 91768, Irvine, CA 92620

ABSTRACT

Through the increase in the popularity of online reading, many people rely on online dictionaries to further understand the text [1]. However, looking up a word manually is a great inconvenience as well as a form of distraction [2]. This paper develops a chrome extension to automatically detect the difficult words for each user, and provide the words' associated definition with a mouse hover. The chrome extension can be customized by adding and removing personal difficult words and personal easy words [3]. Also, the chrome extension offers a deeper level of analytic, including the system analyzing part of speech of the world, to further understand the definition of a selected word or sentence. The chrome extension is applied to a school/work setting in order to improve the working efficiency by providing a simple model to analyze the word definition; it is also useful for casual reading, especially to those that aren't fluent in English. Following the strict SDLC model, the end of the testing stage reflects that most of the users gave positive feedback to the chrome extension with most of the comments centered around convenience and accuracy [4]. Through alpha testing and a small sample of beta testing, the Chrome extension presents productivity improvement on difficult texts.

KEYWORDS

Chrome Extension, NLP, Cloud Computing.

1. INTRODUCTION

The topic is mostly centered around two issues: time management issues and language difficulty [5]. Since the return from the pandemic, I noticed the increase of reliance on the internet, and thus a large number of time people spend on the internet. I have also noticed the increase of distance between different cultures and groups, not just physically, but also the understanding of each other. Through this observation, I see the opportunity to make a useful chrome extension to address both topics. The chrome extension is extremely beneficial to students and workers who often read material from a webpage, as well as the English learners who wish to understand the material or study the language conveniently while reading an online text [6]. High school and college philosophy classes are greatly benefited from the chrome extension because the reading material in those classes is generally moderately difficult as well as covers a variety of areas. In a context where the understanding of the text is more important than the connotation and literary devices, computer software can greatly improve efficiency and understanding. In this case, this

David C. Wyld et al. (Eds): CONEDU, CSITA, MLCL, ISPR, NATAP, ARIN - 2022
pp. 83-93, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.121007

chrome extension is extremely useful among philosophy students. Moreover, time management is becoming a huge issue for high school students, and searching up definitions manually or manually analyzing the context of a word is extremely time-consuming, distracting, and inconvenient. There is no benefit to manually searching up anything when it can be done efficiently and accurately through a computer program. Also, through the distance created by the pandemic, the inclusiveness of every community has become a great issue. This application improves the accessibility of non-fluent English speakers, making them feel more inclusive [7].

Some of the techniques that were previously used without the reading chrome extension are physical dictionaries, googling definition, or Quizlet made by other users [8]. Although all of those techniques usually provide a thorough and accurate definition, they are usually time-consuming and make the reader lose the flow of thinking while looking for the definition through a different resource. These proposals assume readers' willingness to invest their time to go through the inconvenient process to understand the material, which is rarely the case in practice [9]. These proposals are accurate in the areas that they are supporting, but they are not designed for online reading, yet they are still the primary resources for the readers. Similarly, tools such as POS tagging can be used to provide deeper analysis to the accurate definition in a given context, but the actions needed to go through a POS tagging and then a definition is extremely inconvenient and time-consuming. They are the working methods that are impractical. When referring to the resource outside of the web page, it is difficult to draw a connection to the text: they do not address the context of the word and they do not provide a repeated accessible route to the analytic such as hovering over the highlighted word. Also, it is a struggle for the users to manually find the words that they do not understand, which requires a deep understanding, but it is also a premise to the tools available. Another reading assistance such as dark mode reading is addressing a similar issue—but they do not support the understanding of the text nor assist the non-fluent English speaker to achieve a more inclusive community. There are many useful resources available on the internet to address similar issues, but none of them are useful specifically for reading an English article online.

To solve the problems that other tools do not address very well, a chrome extension that can analyze the difficult words automatically and provide easily accessible definitions, as well as an analytic of the definition, would solve the issues. The tool that I am creating addresses the issue starting from a simple interface, where the entire text will be scanned to find the predicted uncommon word and its definition [10]. The uncommon word is also customized with a simple right-click to select whether to add or remove a word from the common or uncommon list, where no words from the common word list will be highlighted and defined, and the rest of the uncommon words will be highlighted and defined. This feature creates accessibility that the chrome extension is useful to almost everyone. Another feature in the right-click drop down menu is its analytic ability, which is an essential ability when working with the definition. Words often have many different definitions across parts of speech and they would often have different meanings, and it's often difficult to know which definition to use when the word is not understood. Through some natural language processing and machine learning, the computer will calculate the part of speech of a word in a certain context, which provides an accurate definition of an uncommon word in a sentence, making reading easier and more convenient for the users.

First, I tried to implement the chrome extension into my daily life. Over the span of two weeks, I used the chrome extension in my philosophy reading assignments, and I saw a significant increase in productivity, and I was able to focus and understand the text much better. In my own experiment, I purposely applied all of the features that the chrome extension provides such as adding and removing common and uncommon words into my account or using deep analysis of a word in context. I have demonstrated an experiment that is applicable in a real-life scenario and it has proven to be useful. I also used public opinion by presenting the chrome extension in a game

competition, where this powerful program can help the gamers further understand a game's explanation or patch notes. The public's positive feedback on the chrome extension is reflected as the chrome extension ended up being top 3 out of over 500 competitors. Lastly, I gave the chrome extension file to some friends as a small beta testing sample, which provides valuable feedback on users' opinions. Through a survey that I gave, I asked specifically about convenience, increase in productivity, and its contribution to a more inclusive environment. Although minor issues were reported such as the speed of the program on a large website, I have received an average rating of 9.8/10 inconvenience, 8.9 in increasing productivity, and 8.6 to a more inclusive environment (all English learners presented 10/10 in this study. Through my own testing and some general feedback, the chrome extensions addresses all of the topics that it was meant to contribute positively to.

The above concludes the introduction of the chrome extension from its purpose to its application. Section 2 of the paper will detail the challenges I went through during the experimenting and designing stage. Correspondingly, section 3 will detail the process of conquering the challenges along with the solutions. To further understand the experiment, section 4 will discuss the design of my experiment and followed by section 5 to draw a comparison to some related work. Finally, section 6 will conclude the research as well as present some future improvements or expansion of the work in the future.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Observation in a real-life context

I have always wanted to write a computer program to assist human productivity and make the environment a more inclusive place, and the positivity of this topic has become more apparent ever since the pandemic. However, it is a challenge to come up with a solution that is generally applicable that has not been done. I did a lot of research regarding the issues of productivity and inclusiveness, and then it follows with the observation of my surroundings to come up with a program that I want to make. I also had to visualize the application of the program to determine the specific features that I want to include in order to achieve my goal, as well as find out the best form of program for the convenience aspect of the topic. Every aspect of the program design is supported by my observation of my environment's daily challenges, my own challenges, and my understanding of computer software.

2.2. Code/Server/Chrome Extension organization

The clarity of the organization becomes an issue when there are many aspects of the program that need to work together to make the final product work. Although there isn't any complicated class structure, the communication between each aspect of the program causes confusion when many similar features share a similar name. Also, errors would often occur under manual input. Encapsulation is still needed even when the methods are under the same class, but it becomes a challenge to separate them out based on each method's purpose for the final product. It's also necessary to understand the information that needs to be communicated through the chrome extension and the server. It is important for the connection of client-server communication because the chrome extension serves as an input to the server's action, yet it also presents the output that comes from the server. It's crucial to understand and organize the features of each class or method and how it is strictly communicated with the others.

2.3. Chrome boundaries and external library’ s documentation

The program uses many pre-existing sources, but not all of them are generally applied and have clear documentation. The implementation of the popular libraries such as flask and firebase was not as challenging, but libraries such as beautiful soup for web-scraper and spacy for word Analytics have methods that I was previously unfamiliar with. Since some of the functions that I am looking for are niche, the less well-known libraries have to be used, and it was a large effort looking for an external library with the feature that I am looking for. However, the biggest challenge of resources comes from chrome plugin implementation. The limitation of the programming language that can be used for the Chrome plug-in forces me to use javascript as its main language. Also, chrome extension requires a strict format that it presents unclearly in its documentation. Moreover, the program fails if any part of the implementation does not fit the requirement of chrome plugins.

3. SOLUTION

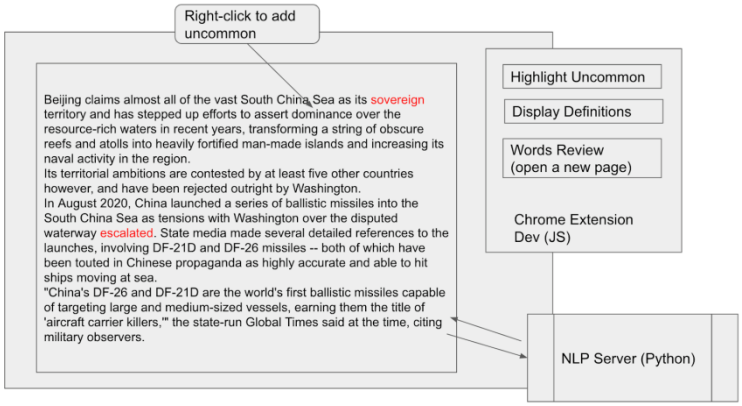

Figure 1. Screenshot of Chrome plugin

When using the Chrome browser, the chrome plugin will highlight the uncommon words for a specific user along with the words’ definition when the mouse hovers over it with one click of a button. The button is located on the top right under the pop-up of the Chrome extension, and it is the interface that the user can interact with. In the webpage, the users can highlight the word and right-click to perform actions such as adding/removing a word to their specific common/uncommon list or they can run a deep analysis of the context they highlighted to understand the texts better. The function is achieved by creating a natural language programming server that includes all of the methods of the function, including web-scraping a website’s HTML given a link, which can extract the raw form of the unfiltered text and structure that can be filtered and analyzed. Other functions such as adding/removing words or defining words are all defined in the server, and all of them can be accessed through a link associated with each method. The information dedicated to each user is stored in Firebase, where each user has a unique account and lists that are unique to them. The chrome extension part, it’s written in javascript, and its function is to read the information in the webpage, mainly accessing and changing the information from the content section. It also defines the appearance of the program, from the way the words are highlighted to the pop-up that the user can interact with. Also, the drop-down from the right-click is also defined under the chrome extension code.

```
# define all unknown words in a website
@app.route('/scrape_website')
def define_all():
  url = request.args['url']
  definitions = {}
  print(url)
  a = scrape_words(url)
  un_common_word = get_uncommon_word("", a)
  count = 0
  print(len(un_common_word))
  #print(un_common_word)
  for word in un_common_word:
    print("############################")
    print(word)
    print("This is " + str(count) + " loops")
    count += 1
    definition = word_definition(word)
    if(definition != None):
      definitions[word] = definition
  return definitions
```

Figure 2. Screenshot of code 1

The main function is completed by running many of the sub-functions after giving it some necessary information. The route /scrape_website is used for communication with the chrome extension. Concise documentation is used to distinguish the difference in each method, and printing lines are used for debugging purposes and to see the progress of the word definition. Similarly, other functions each as adding/removing common/uncommon words are written in methods given its necessary parameter; concise documentation is used everywhere throughout the code to manage the system properly. All of the methods are used either as a part of a bigger method, or it can be accessed by providing a route to it and connecting it to the Chrome extension through a listener.

```
# performing test
@app.route("/")
def homepage_test():
  return("test")


# def build_database():
#     f = open("common10k", "r")
#     common_words = f.readlines()
#     f.close()

#     print(common_words)
#     for i in range (len(common_words)):
#         common_words[i] = common_words[i].rstrip()

#     words = db.collection(u'project_data').document(u'words')

#     words.update({u'commonWords': firestore.ArrayUnion(common_words)})

# build_database()
```

Figure 3. Screenshot of code 2

In order for the program to work, the machine has to understand what words are considered to be basic words, which is done by storing the most basic 10k words in the database as the default common words. However, adding 10k words manually is not realistic, so the commented code above is used to build the database given a file of words, which is a tool that can be used by developers for changes in the database, supporting the communication between the server and the database.

```python
from flask import Flask, request
from flask_cors import CORS
from PyDictionary import PyDictionary
import firebase_admin
from firebase_admin import firestore
from firebase_admin import credentials
from bs4 import BeautifulSoup
import urllib.request as urllib
import spacy
app = Flask(__name__)
cors = CORS(app)

cred = credentials.Certificate("ReadingExtensionDatabaseServiceKey.json")
firebase_app = firebase_admin.initialize_app(cred)
db = firestore.client()
```

Figure 4. Screenshot of code 3

Implementation of the structure of the server and functions such as dictionary and web scraping is properly implemented. These resources are crucial for the functioning of the methods as well as communication between two different components such as the server and the database. Spacy—a word analytic tool that supports the deep analysis function of the program—is implemented and applied properly to meet the function of the application, which is to help the user better understand the word given a context. Py-Dictionary—the default dictionary that provides the detailed raw definition of a word—is used to identify the definition and whether the input is a word. BeautifulSoup is used as a web scraping tool, which can extract most of the texts accurately given HTML, which can then be filtered to clean words in the body section and communicated with the content of the Chrome extension.

```javascript

chrome.runtime.onMessage.addListener(
  function(request, sender, sendResponse) {
    if (request.type === "get_url") {
      //alert(request.data);
      // call
      $.ajax({
        type: 'GET',
        url: 'https://readingextension.laziestcactus.repl.co/scrape_website?url=' + request.data,
        data: JSON.stringify({'url': request.data}),
        encoding:'UTF-8',
        success: function (resp) {
          //console.log(resp);
          // var res = JSON.parse(resp);
          var myMap = resp;
          var words = [];
          for (var m in myMap) {
            //console.log(m);
            //console.log(myMap[m]);
            words.push(m);
          }

          //var all = document.getElementsByTagName("*");
          var all = document.getElementsByTagName("p");
          for (var i=0, max=all.length; i < max; i++) {
              highlight(all[i], words, myMap);
          }
          alert("Uncommon words identified!");
        },
        error: function(er){
        JSON.stringify(er)
          //alert(er.responseText);
        }
      });
    }
  }
);
```

Figure 5. Screenshot of code 4

The Chrome extension aspect of the program is implemented following strict formatting provided by Chrome's format documentation. Alert and console.log are used and commented on throughout the code segment for debugging purposes. The purpose of the code segment is to add a listener from Chrome and use it to control the functions in the server. Similarly, other aspects such as manifest, background, and popup are all implemented by following Chrome's format documentation and focuses on how it can be used to interact with the server and the webpage's content. The style of the popup and interface is also designed as a part of the plug-in.

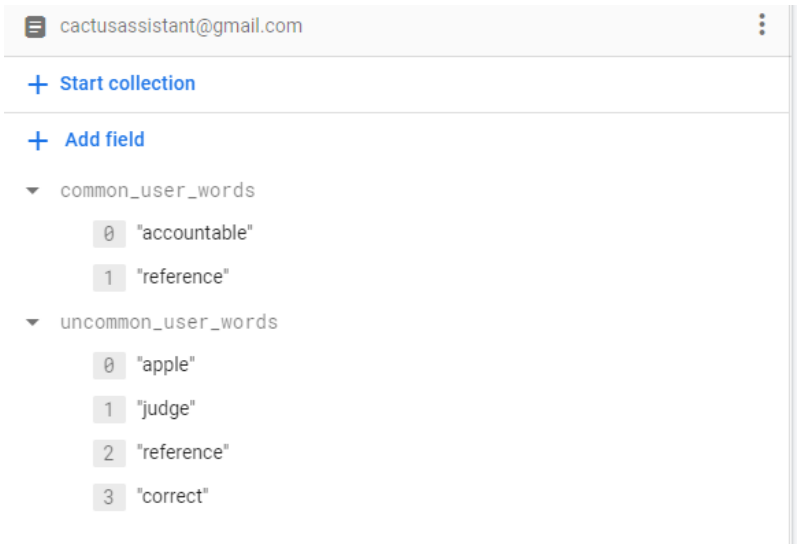

Figure 6. Screenshot of common word

Firebase is used as the database that stores the default information in the database, the users, and the user's information. Above is an example of a user's common word and user's uncommon word, and these can be added through a listener created in Chrome extension, which calls a function in the server to add/remove the word into the database.

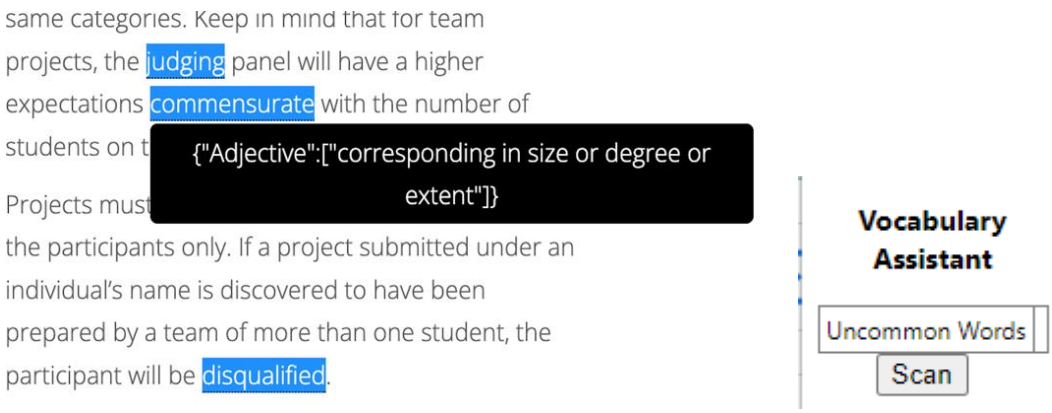

Figure 7. Content style and the interface

Above is the content style and the interface that the user will interact with. There is also another interaction with a right-click drop-down menu and the context menu of the Chrome Extension. The buttons that the users can interact with are linked to listeners through Chrome extension, which will invoke functions in the server, and maybe change the user's personal content in the database.

4. EXPERIMENT

4.1. Experiment 1

An experiment is run on myself, where I use the app to monitor the changes in my productivity while doing philosophy homework. I time myself how long I spend on my philosophy homework for two weeks using the program and two weeks that I do not. Each week, I will be summarizing the main benefits and drawbacks of using or not using the program. The purpose is to find the areas that the app should try to assist by finding the weakness of reading from a webpage and see whether the problems can be addressed in the program. The criteria of success are measured through my feedback and the timetables.

With this chrome extension, I notice significant positive changes in my productivity and understanding as a result of being more focused and having convenient access to definition, which reduces my anxiety in an otherwise tedious assignment. Though the time saved is not a result of the time saved from looking up the word, but from the focus that it brings to me that I complete the work about 21.6% quicker as data collected in a span of two weeks. I have also noticed that I was able to understand the text better especially when the philosophy assignment is talking about an area that I was previously unfamiliar with, and I was able to learn the word much quicker with easy access to its definition. In comparison to reading on a paper, reading on a web page brings a greater distraction and frustration to look up for the definition of the word, which I have noticed that I often get carried over to do other things or lose the flow of thinking that is necessary for a philosophy assignment.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	avg.
35	38	31	34	42	n/a	n/a	36	36	31	32	37	n/a	n/a	35.2
29	28	31	24	25	n/a	n/a	26	23	31	29	30	n/a	n/a	27.6

Figure 8. Result of experiment 1

4.2. Experiment 2

The next test involves the other participants in my philosophy class, including two non-fluent English speakers. Also in a span of two weeks, I ask them to send feedback on what can be improved, what is the most helpful aspect of the Chrome extension, and a rating of the Chrome extension based on 3 categories: convenience, productivity, and inclusive environment. The rating will be out of 10, and the score of 5 is neither improving nor making it worse. The data will be analyzed by taking the average, and the data analysis will have a spotlight for the two non-fluent English speakers.

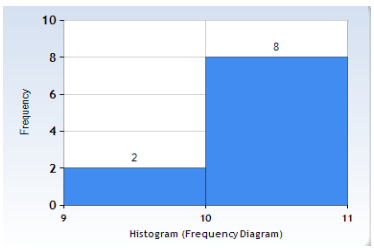

Figure 9. Result of experiment 2 (1)

This is a frequency graph of the rating received from the experimenters, where there are 8 scores of 10 and 2 scores of 9 for the convenience of the program. Based on the experimenter's feedback, there is no negative response regarding the convenience. Conversely, there is feedback complimenting how easy the program is to use for everyone. The two non-English speakers presented a score of 10, which is expected because the time saved for non-English speakers will be significant with this program.

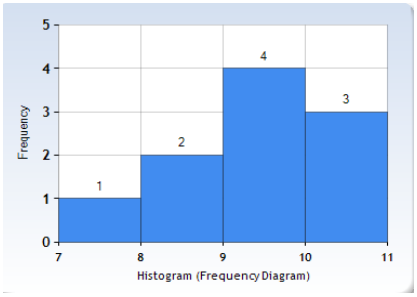

Figure 10. Result of experiment 2 (2)

Corresponding to the convenience aspect, all of the users responded with positive feedback for productivity. However, some users suggested that the program does not increase productivity, with further investigation with people that give a score of 8 or lower, I find that they usually do not look up the unknown words to understand the articles. Once again, the non-English speakers both presented a score of 10 for productivity.

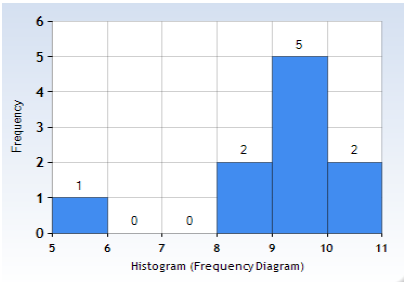

Figure 11. Result of experiment 2 (3)

For the category for an inclusive environment, none of the fluent English speakers presented a score of 10, and even one experimenter gave a score of 5 (no improvement from the program). However, most non-English speakers said that they do not feel the inclusive environment for themselves, but they do believe the program will help those who aren't fluent in English. Their suspicion turns out to be correct, where both of the non-English speakers once again present a score of 10.

Experiments from myself have presented its uses for convenience and productivity, where I can finish my assignments significantly quicker with better focus and less stress. Similar to the other fluent English speakers, I do not see the improvement for an inclusive environment, but I do think the perspective will be different from the non-English speaker. Through the experiment on myself, I can better understand the feedback from the other experimenters. Expectedly, almost every experimenter presented the benefit for convenience and productivity. However, I did not expect people who are extremely good at English to not see an improvement in productivity even though it's logical. As shown in the data, the Chrome extension is useful for everyone, but it is extremely useful for non-English speakers, who gave the extension a 10/10 in every category.

Unrelated to the topics presumed in the experiment, some feedback requested improvement in the interface and expanded the program into more areas. The deep analysis feature is rarely used but it has received positive feedback regardless.

5. RELATED WORK

The work analyzes students' habit of reading the digital text through student's ability to understand grammar visually [11]. The work stresses the difference in each student's reading ability. My work is a complement to this research, where my goal is to reduce the gap in students' reading ability on a webpage. Compared to the current research, they offered solutions through knowledge, where my solution for those that cannot understand texts very well is through technological assistance. Also, the research presents a deeper and more scientific understanding of students' understanding of the digital text, where I focused more on the application of the program and the creation of a program that is easy to understand and use.

This related work focuses on the natural language processing of a sentence that uses computer language to identify sentence type, annotation, and other broad aspects of language processing [12]. It presents a deep analysis of any texts pasted into its application. Compared to my work, this related work focuses more on the analysis of the sentence rather than the application to improve productivity and make a more inclusive community. This related work is more on representing the power of computers to understand the natural language. The strength of the related work is on the variety of ways to analyze the text and the off of website limitation by pasting the text into an application. The strength of my program is in the real-life application to understand the text, where the deep analysis feature in my program is similar to this related work.

The related work focuses on the Chrome extensions that help the students succeed academically in a variety of areas [13]. It focuses on the technological tool that can be translated to real-life applications through the Chrome browser. Compared to my work, this related work is more about discussing the technology's impact in our current world; on the other hand, my work presents a tool to help the readers on Chrome browser, which is also under the umbrella of this related article. This related work has the strength to help the readers understand the usage of Chrome Extension through reasoning. My work has the strength of solving a distinguished problem through technology rather than focusing on the general topic.

6. CONCLUSIONS

Through observation, I see the usage of making a Chrome extension that can help readers read articles from a webpage. The purpose of the Chrome extension is to make the users' life more convenient, increase the users' productivity, and create a more inclusive environment, especially for non-English speakers. The design of the Chrome extension is to automatically scan the uncommon words in an article through web scraping, and then highlight the words and provide the definition with a mouse hovering over the word. To further understand a word in a context, the program also provides a natural language processing aspect with the deep analysis feature, which can often lead to finding the correct definition of a word in a given context. Philosophy class is an example of the Chrome extension's application since it often reads articles from a webpage across a variety of topics [14]. I conducted an experiment on myself, seeing the Chrome extension's impact on convenience, productivity, and an inclusive environment. Through tracking the time I spent on my philosophy homework for 4 weeks, I can see an increase of productivity of about 21%, which is a result of the distraction that I can prevent from looking up the word and the convenient accessibility of the definition. Similarly, through an experiment conducted on my philosophy classmates, most people reflected positively on all aspects of the topic. Moreover, the

application is perfectly designed for English learners based on the experiment. The Chrome extension has proven itself to be effective to solve the challenge brought by reading many articles online, and the effectiveness varies inversely with the English fluency of the reader.

Even though the program has proven to be effective, there is an issue regarding processing speed especially when it comes to a website that includes a lot of uncommon words. Also, some websites have technology that prevents web scraping for security issues, in which the Chrome extension may not apply properly. The user interface is very limited corresponding with the concise feature, but more customizing options are limited to the user. There is also a limitation that comes from the cloud server and database that the Chrome extension is based on, which is a limitation to speed and the number of users.

The speed issue can be improved by giving the users a choice to scan less uncommon words in an article; it can also be improved by finding a quicker accessing method to the definition or through a better algorithm based on Big O notation for a large webpage. Definitions, servers, and databases can be customized to further complement the Chrome extension, increasing the stability of the program [15].

REFERENCES

[1] Coiro, Julie. "Rethinking online reading assessment." Educational Leadership 66.6 (2009): 59-63.
[2] Baron, Robert S. "Distraction-conflict theory: Progress and problems." Advances in experimental social psychology 19 (1986): 1-40.
[3] Carlini, Nicholas, Adrienne Porter Felt, and David Wagner. "An evaluation of the google chrome extension security architecture." 21st USENIX Security Symposium (USENIX Security 12). 2012.
[4] Ragunath, P. K., et al. "Evolving a new model (SDLC Model-2010) for software development life cycle (SDLC)." International Journal of Computer Science and Network Security 10.1 (2010): 112-119.
[5] Claessens, Brigitte JC, et al. "A review of the time management literature." Personnel review (2007).
[6] Stevenson, Julie S., Gordon C. Bruner, and Anand Kumar. "Webpage background and viewer attitudes." Journal of advertising research 40.1-2 (2000): 29-34.
[7] Florian, Lani, and Kristine Black-Hawkins. "Exploring inclusive pedagogy." British educational research journal 37.5 (2011): 813-828.
[8] Sanosi, Abdulaziz B. "The effect of Quizlet on vocabulary acquisition." Asian Journal of Education and e-learning 6.4 (2018).
[9] Caspar, D. L. D., et al. "Proposals." Cold Spring Harbor Symposia on Quantitative Biology. Vol. 27. Cold Spring Harbor Laboratory Press, 1962.
[10] Maltzman, Irving, Seymore Simon, and Leonard Licht. "Verbal conditioning of common and uncommon word associations." Psychological Reports 10.2 (1962): 363-369.
[11] Walsh, Maureen, Jennifer Asha, and Nicole Sprainger. "Reading digital texts." Australian Journal of Language and Literacy, The 30.1 (2007): 40-53.
[12] Manning, Christopher D., et al. "The Stanford CoreNLP natural language processing toolkit." Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations. 2014.
[13] Ok, Min Wook, and Kavita Rao. "Digital tools for the inclusive classroom: Google chrome as assistive and instructional technology." Journal of Special Education Technology 34.3 (2019): 204-211.
[14] Chia, Robert. "Philosophy and research." Essential skills for management research (2002): 1-19.
[15] Berg, Kristi L., Tom Seymour, and Richa Goel. "History of databases." International Journal of Management & Information Systems (IJMIS) 17.1 (2013): 29-36.