verKNACKular:

An Intersection of Implicit Learning and Spaced Repetition Systems

[Student Name Redacted]

[Institution Name Redacted]

AP Research Spring 2022

Word Count: 4,702

verKNACKular:

An Intersection of Implicit Learning and Spaced Repetition Systems

Due to the ever increasing connectivity of our global economy over one billion people attempt to learn a foreign language everyday (Hagiwara & Settles, 2017). This constant demand from new second language learners (L2 Learners) necessitates constant improvement in the field of linguistics, and to the tools that are available to facilitate this learning.

Literature Review

Learning Methods

Spaced Repetition. Spaced Repetition is a memorization technique used to shift memories from your short term memory to your long term memory (Hanson and Brown, 2019). This technique is implemented by spacing out study sessions and focusing on information that the user is most likely to forget next. The forgetting curve determines what information a user is most likely to forget next, and is the graph of the rate at which stored information degrades over time. Spaced Repetition counters the forgetting curve by forcing the user to review the terms they are most likely to forget, right before they forget them. This is effective because it employs both the spacing effect, which states that if something is review multiple times over a long period (ex: 15 minutes a day for 15 days) instead of all at once (ex: 75 minutes on 1 day) that information will be better retained, despite the same total time input (Settles & Meeder, 2016). This effect is combined with the lag effect which states that, if something is reviewed at exponentially increasing intervals (ex: 1 hour, 1 day, 5 days, 2 weeks, 1 month, 6 months, etc…) that information will be better retained over time then if that information was reviewed at stagnant intervals over the same time period with a fraction of the time requirement (Settles & Meeder, 2016). This combination of increased retention and reduced study time has cemented spaced repetition as favorite among the L2 learners (Hanson and Brown, 2019).

This learning method can be employed in many ways. It can be employed physically with systems such as the Leitner system, or digitally in a Spaced Repetition System. When using the

Leitner system, a user would create a series of flashcards, as well as a series of boxes, to sort the retention level of each flashcard. The user would then put all their new flashcards into the first box, review them, and then move any cards they were able to correctly recall into the next box. The further along in the series of boxes a flash card is, the longer you should wait before reviewing it. In the event that the user gets a recall wrong they move that card back to the first box (Smolen et al., 2016). Within the Leitner system this length of time was handled by creating ever increasing box sizes, and once a box became full you would review its content (Hanson and Brown, 2019). They would then repeat this process each day until the information is no longer necessary to review.

While this system can be effective it can also be exceedingly difficult to manage, especially as the number of flashcards increases and the time intervals between each review increase. For this reason most current users of Spaced Repetition employ a program called a Spaced Repetition System to modulate what information they should next review. These systems can come in many forms but in essence they allow the user to create the same series of flashcards they would create for the Leitner system, but remove the burden of managing which flashcards they should next review. Additionally they allow for more precise control over the timing of review intervals.

Though the impact of SRS's is clear, a major concern is user interest. Most studies have reported the utilization of applications such as Anki to fall significantly after only a short time due to the process of SRS assisted learning being grueling (Hanson and Brown, 2019). Because of this, increasing user engagement could increase the effectiveness and user retention of SRSes.

Implicit Vocabulary Learning. Implicit Vocabulary Learning, or Immersion learning is often considered the most efficient method for vocabulary acquisition (Ellis, 2006). This method involves simply immersing oneself in the language with which they wish to expand their vocabulary (Ellis, 2006). While this can come in many forms (books, shows, conversation), its greatest effect is seen in activities in which the learning directly engages with the words such as

during conversations or when reading (Ellis, 2006). This method works because when we see a word in a vacuum, such as on a vocab sheet or a flashcard, it's exceedingly difficult for our brain to recall that word for later use (Ellis, 2006). But, when we see a word with context —such as in a sentence— our brain not only has to attempt to understand the sentence as a whole, which forces it to spend more time focused on the word, but it will also see how the word is used in context (Ellis, 2006). This gives it a chance to attempt to decode how the word is used in context and allows the learner to better construct sentences with the word in the future (Ellis, 2006). While this technique has been proven to be effective, a major reason it is not widely used is that continually finding new content that is both in your target language, and contains frequent use of the vocabulary you are attempting to learn takes time that many L2 learners simply do not have.

Background Technology

Machine Learning. Machine learning is a subfield of Artificial Intelligence (AI) that simulates countless attempts at a specific task in an attempt to train a computer to behave in a specific manner (Dai et al., 2009). This methodology is highly effective for teaching an AI to complete specialized tasks such as the object detection needed in some self-driving cars, text translation for programs like Google Translate, or the vocal recognition used in virtual assistants like Siri. Though powerful, these models can take billions of data points and immense computational power to train. Though there are methods to circumvent this cost.

Transfer Learning. Transfer learning is one such method. Transfer learning is a method in which you train an AI based on a combination of a dataset and a previously trained AI with a similar function rather than just a dataset. This method can allow for significantly reduced computational costs and allows for greater sample efficiency (the degree a model benefits from each data point) because it allows the model to reuse sections of the old model that work to save the same problem rather than recreating them (Dai, Jin, Xue, Yang, and Yu, 2009).

Natural Language Processing. Natural Language Processing (NLP) is a subfield of Artificial Intelligence that uses massive amounts of data to create predictive algorithms that allow computers to generate and interpret text (Grosz, 1982). With the ever increasing complexity of these models many critics are concerned about the potential negative effects of unchecked use of these models (Ponti et al., 2021). Most concerns steam from their potential applications as distributors of miss information (Ponti et al., 2021). Despite this there are also many potential uses for NLP. Some of the current research and applications of NLP are in the sub-fields of: Text2Text generation, conversational AI, Translation, and Text Classification.

Chatbots. Chatbots are a type of program that attempts to simulate a human conversation (Haristiani, 2019). This generally takes place over text-like messages, but is being expanded to include vocal conversations with products such as Alexa, Siri, and Google Assistant. While basic chatbots are often unrealistic on their own, when combined with NLP models they can carry semi-realistic conversations on almost any topic. Though the conversations are impressive it should be noted that they often drift off topic over time due to incorrect assessment of keywords and improper training among other factors (Haristiani, 2019).

Similar Research

Throughout the field of linguistics and the sector of L2 learning there are numerous examples of applications that attempt to leverage spaced repetition or implicit learning to improve vocab retention. Prime examples of these tactics being used can be found in programs such as Anki, a popular SRS that allows a high level of user control over its algorithm and the content they review (see Appendix A); Duolingo, the most popular lesson based language learning app which is popular due to its ease of use, high level of user intractability, and the way it gamifies its process (see Appendix B) (Settles & Meeder, 2016). Though programs such as these are similar to verKNACKular there are a few key differences.

Anki. For pure Spaced Repetition Systems like Anki, a common complaint is the high initial setup that the user must engage in. Part of this setup is the entering of the Expression

(what the user first sees when a card is shown) and Meaning (the answer the user checks with after their attempted recall), this can either be done manually by the user, or by downloading a “Deck” off of the internet. This system could theoretically show the user the word in one context, as long as the Expression of the card is a sentence in the user's target language. But this context is limited by the fact that it remains static throughout the card's lifetime, unless a user decides to go back and edit the Expression to include a new sentence. Because of this Anki is unable to effectively use implicit learning.

Duolingo. Alternately apps such as Duolingo provide context for their users in the form of graphics, sentences, and audio (see Appendix B). This information is passed to the user in a fashion that is similar to that of a SRS, this information is also spaced into individual lessons which are designed to be completed once, and upon having reached a level of competence that Duolingo deems sufficient, you unlock a new set of words/concepts to tackle (Settles & Meeder, 2016). Due to this model, despite there being an SRS system handling some of the information passed to the user, the actual benefit of an SRS is not seen by the vast majority of its users, this is because without recurring and increasing review intervals the user will see neither benefits of the lag effect or the forgetting curve.

Gap

The field of L2 learning has been heavily explored throughout the years, but due to the rapidly expanding and relatively new field artificial intelligence, an avenue to make rapid improvements has been opened. While the intersection of these fields has been studied, there still remains many gaps to be filled with combinations of differing learning methodologies and new AI technologies. One such gap is with the combination of Chatbots and SRS’s to create a more interactive, and potentially more effective, learning experience. While each of these concepts has had their merit proven in separate cases, they have yet to be tested in tandem.

Spaced Repetition Systems have been shown to greatly increase vocab retention over long periods of time when utilized correctly (Hanson and Brown, 2019). This effectiveness has been proven in many studies, yet despite its proven benefits I have been unable to find almost any research into ways in which we can improve upon SRS's.

This lack of expansion on SRS's is the core of my gap. To fill this gap I decided to address what I believe to be a glaring weakness of SRS's, the flash card element. While flashcards can be a powerful tool for studying/learning, due to the isolated way in which they present information, they do not gain any of the benefits of implicit learning, yet another effective L2 learning method (Ellis, 2006). This gap in the research on the intersection of implicit learning and Spaced Repetition Systems is the gap I attempted to fill.

Research Question

After a deep dive into the literature of both Spaced Repetition Systems and Implicit Learning, I decided to fill this gap by asking the question, "How can Artificial Intelligence be used to merge the benefits of Spaced Repetition Systems and Implicit Learning?" I approached this question by using AI text generation models to generate context for vocabulary provided by the user. This context was created with the intent to stimulate greater immersion with the vocabulary and thus increase long term retention.

Implicit Spaced Repetition System

Due to the lack of a term to describe a system that combines Spaced Repetition and Implicit Learning the research decided to define a new type of system known as a *Implicit Spaced Repetition System* (iSRS). An iSRS is a program that uses a Spaced Repetition

System to handle the next item of review, and simultaneously provides some form of context to supplement the definition. This context could be semantic, auditory, or visual.

Method

Implementing Context Generation

Over the course of my investigation into this unexplored avenue of Spaced Repetition and Implicit Learning I identified two possible ways to integrate these two learning methodologies.

Firstly, creating a chatbot that holds a conversation with the user, and over the course of the exchange attempts to inject words from the SRS list into the conversation. Over time, as it detects that the user correctly responds to the words, or when it sees the user correctly use a word from the list, it automatically increases the review interval of the word.

Secondly, creating a Spaced Repetition System that functions as normal, but instead of outputting a flashcard for the user to review, it runs the words the user is attempting to learn through a text2text model and generates sentences for the user to translate back into english.

While there were numerous factors in my decision, in the end it came down to two major points, ease of implementation and assumed benefit. Due to both development complexity and the increased rate at which users see the vocabulary they actually intend to review, I decided to go with the Sentence Generation Model rather than the Chatbot.

Choosing a Framework

A framework is a programming library that a developer can use to create the front end of their application. It allows them to easily build web and app elements through the use of the frameworks API rather than having to program them from scratch. This is beneficial as it can drastically reduce development time, which was a major concern due to my limited research window. When choosing the framework I wanted to work with I had two major points in mind.

Firstly, scalability and accessibility. In this context both scalability and accessibility are in regard to user interaction and access. I wanted a framework that could be accessed on almost any device and provided a high degree of polish to end user experience. For this I looked to three major candidates: React Native, Flutter, and Streamlit. React Native and Flutter are both designed to be deployed as standalone applications, with the benefit of creating one unified code base that can be deployed across multiple platforms. This was enticing to me as being able to deploy the app to Apple, Android, and Windows devices —among several others— would allow the application to reach a greater number of users; increasing accessibility. In contrast to this, Streamlit was designed exclusively for web development, I was somewhat opposed to using Streamlit as creating an application seemed to be the industry norm. This is mainly because it allowed for offline use and easily separated user profiles, both things that L2 learning applications greatly benefit from.

Secondly, development language. Despite its shortcomings, Streamlit did have one major advantage over the competition, this being its development language. While React Native and Flutter programs are programmed in Javascript and Flutter/Dart respectively, Streamlit is programmed in Python, a language that, not only am I much more comfortable developing in, but is also used for a majority of AI research (Parmenter, 2021). This meant that I would have access to a greater variety of AI/ML models, increasing my odds of finding one that was applicable to my research.

Because of this increased pool of models, and the time I would save by developing in a language with which I was comfortable. I eventually decided that the tradeoffs for using Streamlit were worth it and settled on using it for the remainder of my development.

Translation Model

When selecting the translation model for this layer I had 3 primary factors in mind: usability, accuracy, and cost. During my search, I found that while both the DeepL Translator and Google Translate were both well regarded for their translation accuracy , DeepL was cited as being overall more accurate (Coldewey & Lardinois, 2017). Since they were both available free of charge until use exceeded a certain threshold, it came down to their usability. As I began to test both models I quickly learned that DeepL's API was much simpler to implement. Due to time constraints and the slightly increased accuracy, I decided to use the DeepL Translator over Google Translate.

The Process/Results

Application Flow

My research focused on creating a bridge between the learning methods of Spaced Repetition Systems, and Implicit Vocabulary Learning. And, how artificial intelligence could be used to support this bridge. After several attempts, the program I devised to fulfill this role works as follows.

First, the user enters a new word; this word is then stored in a word object (see Appendix C) which keeps track of: the word itself, the words current interval, and the words last recall date. This object is then passed into a deck handler object (see Appendix D). Deck handler objects are in charge of storing words the user is attempting to learn, keeping track of which words need to be reviewed next, and handling the decision making behind when to swap words between these states.

Next, each deckhandler randomly selects a word from its list of available words; these words are then fed into the sentence generation layer. This layer is in charge of creating context for the words within the SRS. It then takes in three words from the SRS (one noun, one adjective, and one verb) and passes this list into the keytotext model. The keytotext model then returns a sentence containing these words.

Finally, these sentences are then run through the Deepl translation layer. This layer translates the sentence from english to the users target language. This translated sentence is then displayed to the user who can enter what they believe the original sentence was. The system will then compare the user's answer with the original sentence, and the SRS will update the status of all words used in the generation of the sentence to reflect whether or not the user was correct.

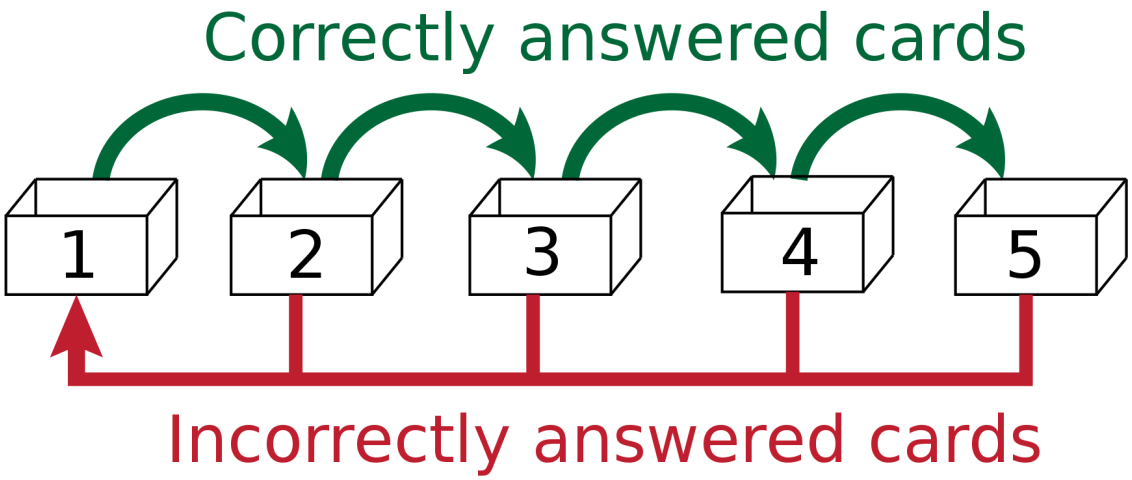
Specifics of the Spaced Repetition System

Within the SRS each part of speech has its own deckhandler, each deckhandler object is its own mini SRS. Currently the program is configured to have lists of: nouns, adjectives, and verbs. This list could be expanded to hold any part of speech, though minor edits to the program would be required. This separation of word types was implemented because, after moderate testing, queries to the key2text-new model that had one noun, one adjective, and one verb were found to provide the best results.

While there are many types of SRS's, I modeled mine after one of the most common systems known as the *Leitner System* (Smolen et al., 2016). This system works by sorting each word into different "Boxes", in this context a box is just the interval of practice. When the SRS goes to select the next word it determines the available words by comparing the date of each word's last retrieval (a correct use or comprehension of the word) to the current date. Then, if this difference is greater than the interval that has been assigned to the word, it is moved back into circulation (see Figure 1).

Figure 1

Simple Example of the Leitner System (Zirguezi, 2012).



After every retrieval the last recall date is reset to the current date, and on a successful retrieval the interval is moved up by one, while on an unsuccessful retrieval the interval is reset to zero. My program has four intervals which have spacing presets of 1, 5, 10, and 20 days respectively. The time intervals of each box can be edited by the user if they feel they need more or less time between each interval, additionally, the number of boxes can be edited if the user wants a more dynamic setup, though minor edits to the program would be necessary. This system allows the user to create properly spaced recalls along their own forgetting curve, hopefully increasing long term retention of L2 vocabulary (Hanson and Brown, 2019).

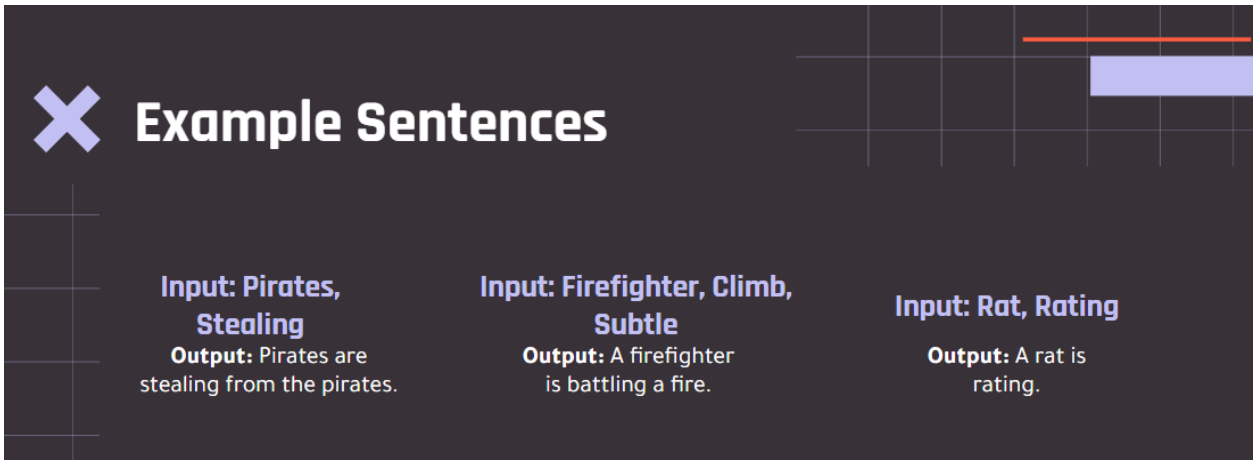
Analysis/Discussion

Though I was unable to test the application on a large scale during the development process I did run a standard series of tests to confirm functionality of the program, during this time I encountered several interesting complications.

One recurring aspect I encountered in testing was that, often, either due to the inputs received or inaccuracies in the model, the program would generate sentences that, despite being grammatically correct, were obviously illogical when viewed by a human. Some examples can be seen in Figure 2. While at first this was concerning, I soon realized that due to the nature of implicit learning, realism was not necessarily a factor of importance, and furthermore some studies suggest that it is easier to remember concepts/terms that we connect to more absurdist statements/ideas.

Figure 2

Example Sentences outputted by the key2text-new.



Additionally, when testing I noticed that because of the way the model was trained there were times when I would pass in a list of three words and the sentence would be missing one or more of those words. While there could be many causes of this, I eventually came to the conclusion that, when constructing a sentence there were times that the algorithm was electing to ignore certain words within the list. Whether this was simply due to the way in which key2text-new was trained, the architecture of the T5 model it is based on, or some other factor is still unknown to me. Though originally I was concerned by this, I soon realized that this problem could be circumvented if we ran a check on each sentence before translation to assert which words were present. At which point we could update only the words we found, rather than

updating all the words that were passed into the keytotext model. Though due to this trend going undetected until very late in the development cycle I was unable to actually implement this minor change.

One final complication I encountered when testing this model was that, after a sentence has been generated with a word or with a set of words; if those same words were inputted again there was a chance the same sentence would be used. This problem confused me recently when Gagan Bhatia —the creator of the key2text-new model— put out a statement explaining that, “The models have been trained in such a way that there are only one sentence for one set of keywords. That is why the same inputs generate the same outputs.” While this is obviously problematic, I realized that in the long term it is only somewhat detrimental to the effectiveness of the program, this is because the chances of the chosen words being the same decreases everytime another word is added. Because of this, the problem will only be encountered by a small number of users, but I would strongly recommend that if future researchers attempt to create a similar program, they train their model in such a way that it does not allow for this to occur.

Limitations, Implications, and Directions for Future Research

Non-Technical Limitations and Implications

Due to both the nature of my research and the time constraints provided I was unable to test the iSRS program I created. Because of this lack of testing the results and findings presented within my research cannot provide a concrete resolution on the validity and effectiveness of an iSRS over a normal SRS. Future research could address these limitations by running a study that compares how students perform in a L2 course when using a traditional SRS, say anki, versus how they perform when using an iSRS such as the one I created.

Another limitation of this study was the fact that it only works for English users. This is because the system is only trained to make sentences out of English words and thus if you do not speak English you can not add words to the system to learn. This problem could be

addressed by future researchers with a new sentence generation model that was trained to create sentences in multiple languages.

The final limitation of this study is the fact that, while I do have some degree of experience in the field of programming, I have nowhere near the years of experience and training that would be required to develop a market level application. Because of this there were many times throughout my process where I had to remove or reconfigure planned features to both make up for lost time or simply due to the enormity of the task at hand. Because of this the application I developed is more along the lines of a minimum viable product. This means that it fulfills all basic functionalities that would be required to get user feedback and test the effectiveness of such an application, but is not at a point where it could be deployed to users. Despite this there is still one missing component that I was unable to implement within the timeframe, this feature being cloud storage. This means that while the system is fully functional, it does not have any way to store information, thus a user must leave the application open on their computer or their information will not be stored. This could be fixed by implementing a cloud storage solution such as MongoDB to upload user data so that they can come back to the website and complete reviews at any time, in any place. Though it should be noted that this did not in any way impact my testing of the application as I simply modified the machines perceived date and time when testing.

Technical Limitations and Implications

One technical limitation of this study was the lack of publicly available high quality sentence generation models. Due to the time restraints, computational cost, and technical knowledge required to train a model myself, I was unable to create a custom model for verKNACKular, this forced me to search for open source models that could effectively generate sentences from a list of words. Eventually, I found the key2text-new model which could generate sentences with a fair degree of accuracy. Though key2text-new was accurate, future researchers might improve this system by training a model that was specifically designed to

work within the realm of an iSRS or by training the model on a larger and more precise dataset which might allow for a reduction in nonsensical sentences and missed words.

Another problem that arose from my use of translation models was that, while the models had a high level of accuracy, there were times when the DeepL model would choose a synonym instead of the actual target word. This would obviously be problematic as if a word is unknown to the user it would make it impossible to correctly answer the question. This problem is simply due to the technical complexity of translating languages. Currently I can think of no way to address this problem, though due to the low number of occurrences this problem should not greatly impact the systems overall performance.

Future Directions

Because of the proven effectiveness of the Implicit Learning and Spaced Repetition Systems I am confident that, despite my inability to gather data during this study, an intersection of these two ideas in an iSRS has the potential to greatly increase vocab retention of L2 learners who properly implement one into their study routine.

While I demonstrated one example of an iSRS with my implementation in verKNACKular, I'm certain that many other implementations, such as the chatbot, could potentially leverage this intersection of learning theories to an even greater degree. I believe that this research will open the door into future investigations on the use of iSRSes, and I would strongly encourage future researchers to attempt to innovate, or incorporate, the principles and techniques I used in verKNACKular in their own research.

References

Ellis, N. C. (1995). THE PSYCHOLOGY OF FOREIGN LANGUAGE VOCABULARY acquisition: IMPLICATIONS FOR CALL. *Computer Assisted Language Learning*, *8*(2-3), 103-128. https://doi.org/10.1080/0958822940080202

Grosz, B. J. (1982). Natural language processing. *Artificial Intelligence*, *19*(2), 131-136. https://doi.org/10.1016/0004-3702(82)90032-7

Ponti, E. M., Reichart, R., Korhonen, A., & Vulic, I. (n.d.). Isomorphic Transfer of Syntactic Structures in Cross-Lingual NLP. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. https://doi.org/10.18653/v1/P18-1142

Seibert Hanson, A. E., & Brown, C. M. (2019). Enhancing l2 learning through a mobile assisted spaced-repetition tool: An effective but bitter pill? *Computer Assisted Language Learning*, *33*(1-2), 133-155. https://doi.org/10.1080/09588221.2018.1552975

Settles, B., T. laflair, G., & Hagiwara, M. (2020). Machine learning–driven language assessment. *Transactions of the Association for Computational Linguistics*, *8*, 247-263. https://doi.org/10.1162/tacl_a_00310

Hagiwara, M. & Settles, B. (2017). *3 habits of successful language learners.* TechCrunch. https://techcrunch.com/2017/03/05/3-habits-of-successful-language-learners/?guccounter=1&guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce_referrer_sig=AQAAAKp4R6ez_yNnIuPiu_X6C5prgaT5N6FnaK2ZCBgx4S2xgO0qpXtNVGie7ZB9DHhiLVJ1ooT0iJW3jH381Kx7_sT-do2ZBMS3aQCRjGp5DzratfTfi55OJsb7AK-0-Cx9m__sjRbbm_Bdtq5pR5fRYgdAAlnG6sK7xfOAFHoyqm4_#:~:text=Roughly%201.2%20billion%20people%20worldwide%20are%20currently%20learning%20a%20foreign%20language.

Settles, B. & Meeder, B. (2016). *A Trainable Spaced Repetition Model for Language Learning*. duolingo research. https://research.duolingo.com/papers/settles.acl16.pdf

Smolen, P., Zhang, Y., & Byrne, J. H. (2016). The right time to learn: mechanisms and

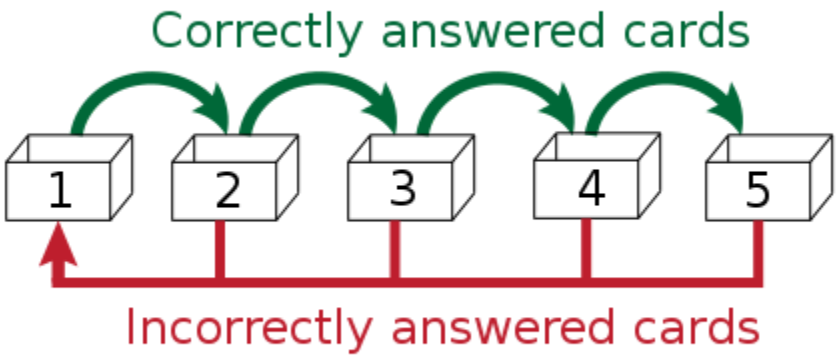
optimization of spaced learning. *Nature reviews. Neuroscience*, *17*(2), 77–88. https://doi.org/10.1038/nrn.2015.18

Dai, W., Jin, O., Xue, G.-R., Yang, Q., & Yu, Y. (2009). EigenTransfer: a unified framework for transfer learning. *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*. https://doi.org/10.1145/1553374.1553399

Parmenter, L. (2021). 9 Best Programming Languages for AI [2022 Project Guide]. *Springboard.* https://www.springboard.com/blog/data-science/best-programming-language-for-ai/#:~:text=Python%20is%20one%20of%20the,and%20TensorFlow%20facilitate%20deep%20learning.

Coldewey, D & Lardinois, F (2017). *DeepL schools other online translators with clever machine learning.* TechCrunch. https://techcrunch.com/2017/08/29/deepl-schools-other-online-translators-with-clever-machine-learning/

Zirguezi (2012). *Leitner system alternative* [



]. Wikipedia. https://commons.wikimedia.org/wiki/File:Leitner_system_alternative.svg

@misc{bhatia,

title={keytotext},

url={https://github.com/gagan3012/keytotext},

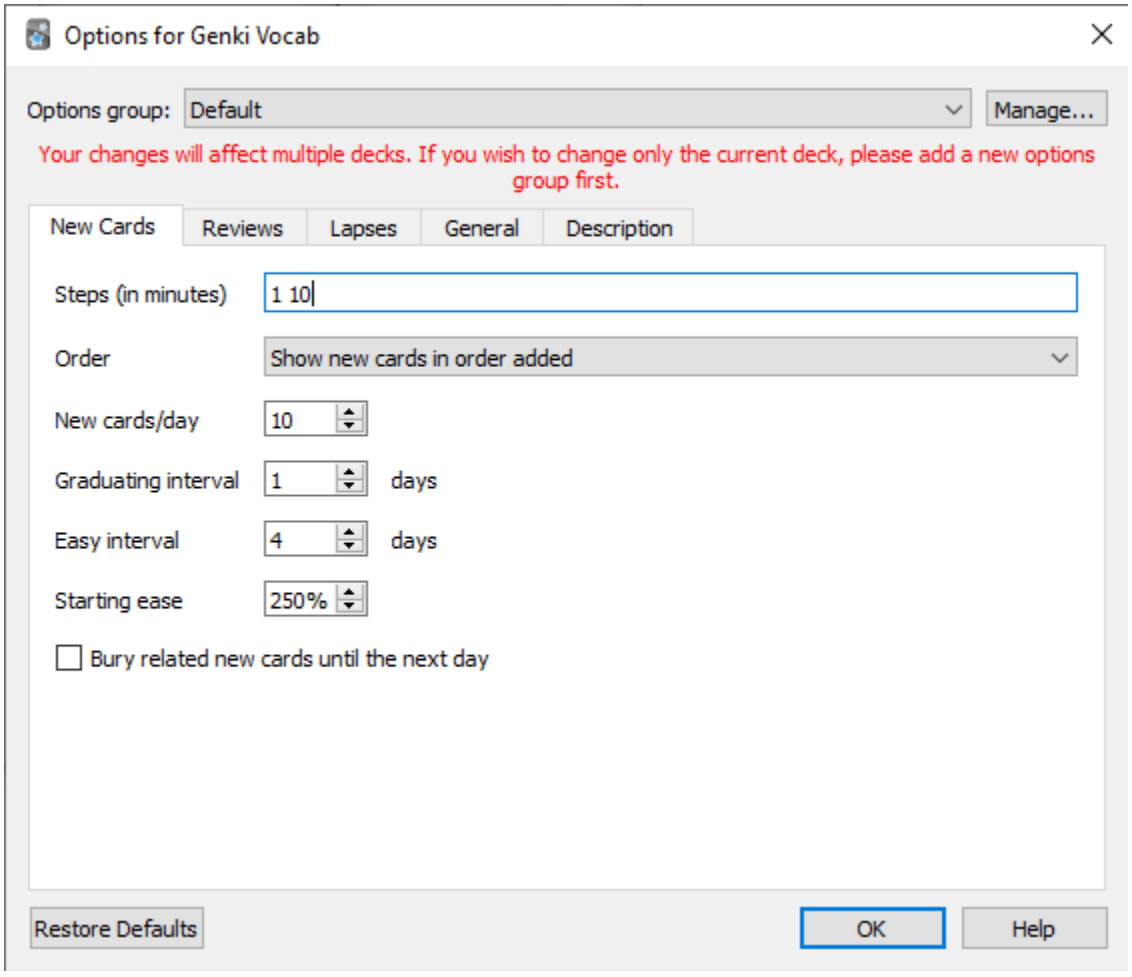
journal={GitHub},

author={Bhatia, Gagan}

}[1]

[1] The author recognizes that this citation is not formatted in the same manner as the others, but after direct contact with the author of the keytotext model he requested that the citation be included with this specific formatting.

Appendix A: Image of Anki's settings menu



Appendix B: Example of Duolingo's gamified questions

Appendix C: Code used in the Creation of Word Objects

VocabWord.py > VocabWord

```python
import datetime

#file --VocabWord.py--
class VocabWord:
    word = ''
    currentInterval = 0
    lastRecallDate = 0

    def __init__(self, myWord):
        self.word = myWord
        self.currentInterval = 0
        self.lastRecallDate = datetime.datetime.now()

    #returns word
    def getword(self):
        return self.word

    #precondition: correct is a bool
    def updateInterval(self, correct):
        self.lastRecallDate = datetime.datetime.now()
        if correct:
            if self.currentInterval <= 3:
                self.currentInterval += 1
        else:
            self.currentInterval = 0

    def daysSinceLastRecall(self):
        x = datetime.datetime.now()-self.lastRecallDate
        return x.days
```

Appendix D: Code used in the Creation of DeckHandler Objects

DeckHandler.py > ...

```python
import random
from VocabWord import VocabWord

class DeckHandler:
    currentReviews = []
    pendingReviews = []
    #interval times are in days, for each number you add to the list the number of total boxes increases
    intervalTimes = [1,5,10,20]

    #init
    def __init__(self):
        self.currentReviews = []
        self.pendingReviews = []
        self.intervalTimes = [1,5,10,20]

    #adds a new word to the list of words
    def addWord(self, newWord):
        theWordsWeCreate = VocabWord(newWord)
        self.currentReviews.append(theWordsWeCreate)

    #takes in the index of a word and moves it from the current reviews to pending reviews, additionally now updates the interval for convience
    def reviewed(self, currentIndex):
        self.currentReviews[currentIndex].updateInterval(True)
        self.pendingReviews.append(self.currentReviews[currentIndex])
        self.currentReviews.pop(currentIndex)

    #moves all words that need to be reviewed from pending to current reviews
    def recentReviewsCheck(self):
        for i in self.pendingReviews:
            if i.daysSinceLastRecall > self.intervalTimes[i.currentInterval]:
                self.currentReviews.append(i)
                self.pendingReviews.remove(i)

    #gets a random review from the list of current reviews
    def getNextReview(self):
        r = random.randrange(0, len(self.currentReviews))
        return self.currentReviews[r].word
```