Assessing the Memorability of Familiar Vocabulary for System Assigned Passphrases

by

Noopa Jagadeesh

A thesis submitted to the
School of Graduate and Postdoctoral Studies in partial
fulfillment of the requirements for the degree of
Master of Science in Computer Science

Faculty of Science
University of Ontario Institute of Technology (Ontario Tech University)
Oshawa, Ontario, Canada
August 2021

THESIS EXAMINATION INFORMATION

Submitted by: **Noopa Jagadeesh**

Master of Science in Computer Science

Thesis title: Assessing the Memorability of Familiar Vocabulary for System Assigned Passphrases

An oral defense of this thesis took place on August 13th, 2021 in front of the following examining committee:

Chair of Examining Committee :	Patrick Hung Professor, Faculty of Business and Information Technology
Research Supervisor :	Miguel Vargas Martin Professor, Faculty of Business and Information Technology
Examining Committee Member:	Julie Thorpe Associate Professor, Faculty of Business and Information Technology
Thesis Examiner:	Jeremy Bradbury Associate Dean, Graduate and Postdoctoral Studies; Associate Professor

The above committee determined that the thesis is acceptable in form and content and that a satisfactory knowledge of the field covered by the thesis was demonstrated by the candidate during an oral examination. A signed copy of the Certificate of Approval is available from the School of Graduate and Postdoctoral Studies

Abstract

Text-based secrets are still the most commonly used authentication mechanism in information systems. Initially introduced as more secure authentication keys that people could recall, passphrases are tokens consisting of multiple words. However, when left to the choice of users, they tend to choose predictable natural language patterns in passphrases, resulting in vulnerability to guessing attacks. System-assigned authentication keys can be guaranteed to be secure, but this comes at a cost to memorability. In this study we investigate the memorability of system-assigned passphrases from a familiar vocabulary to the user. The passphrases are generated with the Generative Pre-trained Transformer 2 (GPT-2) model trained on the familiar vocabulary and are readable, pronounceable, sentence like passphrases resembling natural English sentences. Contrary to expectations, following a spaced repetition schedule, passphrases as natural English sentences, based on familiar vocabulary performed similarly to system-assigned passphrases based on random common words.

Keywords: Authentication; System-assigned passphrase; Memorability; GPT-2

Author's Declaration

I hereby declare that this thesis consists of original work of which I have authored. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize the University of Ontario Institute of Technology (Ontario Tech University) to lend this thesis to other institutions or individuals for scholarly research. I further authorize the University of Ontario Institute of Technology (Ontario Tech University) to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for scholarly research. I understand that my thesis will be made electronically available to the public.

_____ Noopa Jagadeesh

Statement of Contributions

I hereby certify that I am the sole author of this thesis. I have used standard referencing practices to acknowledge ideas, research techniques, or other materials that belong to others. Furthermore, I hereby certify that I am the sole source of the creative works and/or inventive knowledge described in this thesis.

Acknowledgements

I would like to express my deep gratitude to my research supervisor Dr. Miguel Vargas Martin, for the continuous guidance, motivation, support, patience, and immense knowledge and providing me the opportunity to do this research. I am honored and privileged to work and study under his supervision.

I would like to thank Dr. Julie Thorpe for her valuable inputs on the research.

I would also like to give a special thanks to my supervisory committee members, and my external examiner, for their valuable input throughout this thesis editing process.

Last but not least, my heartfelt gratitude to my husband Rajith for his patience and wholehearted support, to my parents for their unconditional love, and to all my friends. The completion of this work would not have been possible without them.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Authentication is the process of confirming who a user claims to be. Authentication is a critical component of almost every computer security architecture. Everyone now has a computer and manage a number of online accounts, such as online banking, personal e-mail, instant messaging and social networks. This means that most people need to manage specific individual credentials in order to authenticate themselves to different systems, with each system having a different rule for what is acceptable and not. Used in both corporate and personal settings, passwords have been the most popular means of authentication for many decades, despite many attempts to replace them [11], and they continue to dominate web authentication. A typical internet user has the complex task of creating and recalling passwords for many different accounts. Most users struggle with this task, thereby adopting insecure password practices [22][33][10] or frequently having to reset their passwords. Allowing users to create their own passwords often leads to users picking passwords that are too easy for an attacker to guess [37][64], resulting in security breaches and loss of privacy for users. There are numerous real-world examples of password breaches

[57].

To prevent users from picking passwords that are weak, of poor quality and easily guessable, system administrators adopt password-composition policies (e.g., requiring passwords to exceed a certain minimum length, include symbols and numbers and that they are not words found in a dictionary). These password composition policies force users to create more secure passwords by limiting the space of user-created passwords to exclude easily guessed passwords and thus make passwords more difficult for an adversary to guess [52][32]. Use of password-composition policies also result in a significant reduction in password reuse. However, many users struggle to create and remember their passwords under strict password-composition policies [53].

With a system-assigned password, a password is automatically generated by the authentication system and assigned to a user [64]. Removing user choice and having system assigned passwords, make passwords more secure. Such system-assigned passwords can be guaranteed to be sufficiently difficult to guess, however, most users find these passwords hard to remember, type [36] and therefore, tend to write them down.

Multi-word passphrases may be a promising improvement over passwords. A passphrase is a password composed of a sequence of words, intended to increase password length and therefore security, while retaining memorability [31], however subjected to an increased rate of typographical errors. Resistance to targeted impersonation, throttled guessing, unthrottled guessing, and leaks from other verifiers [11] are some of the security advantages of system-assigned passphrases. Shay et al. [51] found that the use of system-assigned passphrases comes at a cost to memorability. Rather than memorizing, users tend to write down or otherwise store both passwords and passphrases when they are system assigned. It has been suggested that if people paired a system-assigned passphrase with a story, it would improve memorability [38]; however, studies indicate that even this was not a

successful strategy [51].

Prior research has made attempts to adopt techniques from cognitive psychology to help users memorize system-assigned secrets. These make use of cognitive processes which operates on individual's mental contents to generate some responses. These mental contents are formed by different representations and encodings which an individual makes from external stimuli, rules, knowledge, images, scenes and identical content originating from either long-term or short-term memory [16]. In *CuedR* [2], users can choose from multiple types of cues (visual, spatial, and verbal) to aid in the recognition of images of six objects, each corresponding to one letter of a system-assigned secret. In *spaced repetition*, information is learned in multiple sessions that are separated by increasing intervals of time. This memorization technique was introduced by Bonneau and Schechter [12] to reliably implant a 56-bit authentication secret in users brains. In *implicit learning*, learning and memorization happens unintentionally without awareness. In this approach the assigned passwords don't need to be explicitly remembered. Instead, it employs implicit memory of users. *Chunking* is the process of breaking down long strings of information into smaller units or chunks, that are meaningful and are easier to store in memory. Huh et al. [27] studied the effect of chunking on system-generated PIN's and found that chunking overall improves memorability. *Learning Through Games* is another memorization technique based on cognitive psychology. To train users to type a random key sequence in quick order, Bojinov et al. [8] used implicit memory for authentication, using a game similar to one used in psychological research to study implicit memory formation [48]. A computer game helped authors plant an authentication secret in a user's brain such that the users are unaware of their secret, and thus incapable of revealing the password due to coercion thereby providing a measure of resistance against "rubber-hose" attacks. Haque et al. [25] used the *Method of Loci* and achieved 86% recall success rate for a login session one week

after the training. This technique involves identifying a few landmarks (also known as loci) in some familiar place and then forming a graphic visual image of placing each item to be remembered on each of these landmarks. By mentally re-walking through these landmarks, the items can be later retrieved during recall period.

With the goal of improving the usability and memorability of system-assigned passphrases, we propose in this work a creation of system generated passphrase based on a familiar vocabulary to the users. As a proof-of-concept, this thesis describes the results of a 500-participant study on the memorability and usability of system-assigned passphrases generated from a vocabulary familiar to the user. For the purpose of the study we generated passphrases from a story familiar to the user. The passphrases are generated with the Generative Pre-trained Transformer 2(GPT-2) model [1] and are readable, pronounceable, natural English sentence like passphrases. The reason for generating passphrases using GPT-2 over using phrases directly from music lyrics, movies, literature or similar public available material is to prevent dictionary attacks. The idea is to have a user-specific vocabulary, e.g., generated from user-created text content, such as emails or blog post or tweeted messages; users may also choose a pre-generated dictionary specific with their interests e.g., vocabulary from a favorite poet or books. This user-specific vocabulary is then used to train a GPT-2 model, that generates readable, pronounceable, natural English sentence like passphrases.

1.1 Motivation

Our work was motivated from a proof-of-concept prototype -*Myphrase* [55] which is a multi-word password scheme that encourages users to use words that are more personal. A *Myphrase* passphrase consists of randomly chosen multiple words from a user-created/selected

dictionary. Here the authors believed that *Myphrase* passphrases may retain security advantages of random phrases without being too difficult to remember, as the dictionary is created from a familiar/personal vocabulary. However, to the best of our knowledge this hypothesis was never tested.

1.2 Main Contributions

We hypothesize that following a spaced repetition schedule, passphrases as natural English sentences, based on familiar vocabulary are easier to recall than passphrases composed of random common words. A familiar vocabulary is a user-specific vocabulary or a vocabulary that is personalised, e.g., generated from user-created text content, such as emails or blog post or tweeted messages; user chosen pre-generated dictionary specific with their interests e.g., vocabulary from a collection of favorite poems or books. Spaced repetition, is a memorization technique where information is learned in multiple sessions that are separated by increasing intervals of time.
The main contributions of our work are as follows:

1. We present a novel approach for system-assigned passphrases that harnesses the power of Generative Pre-trained Transformer models such as GPT-2.

2. We discuss the security goals to be achieved when designing a system-generated passphrase based on personal/familiar vocabulary.

3. Through an online user study with 500 participants on Amazon Mechanical Turk, we test the hypothesis - following a spaced repetition schedule, passphrases as natural English sentences, based on familiar vocabulary are easier to recall than passphrases composed of random common words.

Would a passphrase crafted in plain natural English according to the style and feel of a familiar vocabulary be more memorable than a random passphrase? We conducted a test with 500 participants in Amazon Mechanical Turk to find, at least partially, quantitative answers to such question. We used a generative pre-trained transformer (GPT) network to generate passphrases based on three of the most popular books (according to the top 15 books downloaded from Project Gutenberg library) in such a way that the generated passphrases do not appear in these books but still preserve the style and feel of these writings. To measure the quality of the generated passphrases, we built a sentence similarity score heat map from a generated sample of passphrases, showing that passphrases are rather dissimilar amongst themselves. Furthermore, we built a bidimensional representation of an embedding of the generated passphrases which shows that the passphrases tend not to form clusters.

In terms of security, given the limitations of the familiar vocabularies used in our study, we were only able to provide as much as 16.8 bits of entropy. It is important, however, to keep in mind that our study is a proof-of-concept aimed at testing memorability, although a real implementation should provide sufficient security to protect from at least throttled online attacks but ideally from offline attacks. We are aware that linguistically correct passphrases are susceptible to parts-of-speech (POS) attacks; and while the books and the GPT-2 generated passphrases did exhibit important POS distribution similarities, we removed those passphrases that don't have a tag-sequence that is either unique from each other or with the tag-rules from n-grams. After this, we ranked passphrases using joint probabilities of sequences. Our security analysis also provides insight as to the marginal guesswork for these books even though the generated passphrases wouldn't be textually taken from the books but from the GPT-2 output.

Our study showed that following a spaced repetition schedule, passphrases as natu-

ral English sentences, based on familiar vocabulary performed similarly to system-assigned passphrases based on random common words. Nonetheless, the familiar vocabulary passphrases used in our test were 5, 6 and 7 words long, while the random word passphrase was only 4 words long. Even with a longer passphrase, the familiar vocabulary passphrase was equally remembered compared to the random word passphrase.

We believe that our work tests the viability of familiar vocabularies as a new security paradigm, with the potential of improving memorability while maintaining similar security. We study a long-held belief that natural English passphrases are more memorable than random words, bringing together artificial intelligence and passphrase authentication. Our user-study with 500 participants on Amazon Mechanical Turk showed that following a spaced repetition schedule, passphrases as natural English sentences, based on familiar vocabulary performed similarly to system-assigned passphrases based on random common words. Our study showed that system-assigned passphrases based on random common words performed just as well in terms of memorability and recall over system-assigned passphrases from a familiar vocabulary. Future work may reveal that the power of natural language generative pre-trained networks may catch up with passphrases, rendering them less secure. Or, on the other hand, further studies may find that artificial intelligence can in fact be effectively harnessed to improve passphrase security.

Chapter 2

Background and Related Work

2.1 User-Chosen Secrets

Many techniques have been suggested by organizations to help users construct stronger passwords. One technique is to have passwords follow a password-composition policy, so as to force the user to construct passwords based on strict rules, thereby making the password stronger. AlFayyadh et al. [4] investigated a set of password composition policies that a user would have to comply with when selecting passwords for their various online services, by studying the password policies of different institutions, companies and websites. They found several inconsistencies in policy requirements. They found that these password policies have diverging password requirements for service that have the same authentication assurance level. Inflexible strict policies often lead to user frustration [53] and places a heavy burden on users, both in creating a new password and also in learning them [28], often leading to construction of vulnerable passwords with predictable patterns [53][62]. Schechter et al. [49] proposed replacing the complex password policies by allowing users to

choose any password they want, as long as the chosen password is not already too popular with other users and is not an target for a statistical guessing attack.

Another measure for encouraging users to create stronger passwords is the use of password meters. A password meter proactively check the strength of password at the time of its creation by the user and provide a visual feedback on password strength [59]. It also provides suggestions to users to assist them in creating stronger passwords [58]. A password strength meter is considered accurate if their score correctly reflect password strength. High accuracy is one important aspect that impacts the security of a password strength meter. Golla et al. [24] identified that the password strength meters used in practice are less accurate than academic proposals, and they couldn't see any significant improvement of password meter accuracy when comparing with password meters from five years ago.

Skillen et al. [55] proposed a proof-of-concept called *Myphrase* which is a multi-word password scheme that encourage users to use words that are more personal or familiar to them. For this they proposed the creation of user-specific dictionary generated from user-authored content such as sent emails and blogs to construct a master passphrase by randomly selecting words from this constructed dictionary. Users can also choose from a pre-generated dictionary that aligns with their interests. They proposed two variants of passphrase construction - random sequence and connected discourse. In a random sequence, words are chosen uniformly across the dictionary, and in connected discourse, the words chosen are tagged using a part-of-speech engine and inserted appropriately into sentence templates. They then salt the master passphrase with the site's URL domain to create website-specific passwords.

2.2 System-Assigned Secrets

Shay et al. [51] experimented on the memorability and usability of random system assigned passphrases. They conducted an online study with 1,476 participant on the usability of three and four word system-assigned passphrases in comparison to system-assigned passwords composed of five to six random characters, and an eight character system assigned pronounceable passwords. Their study concluded that usability of system-assigned passphrases are similar to system-assigned passwords of similar entropy. They also experimented on using random system assigned passphrases, by encouraging users to imagine a scene that links each word in the passphrase. In their paper they called this condition as *pp-nouns-instr*. In this condition participants were assigned four nouns, randomly sampled with replacement from a dictionary containing 181 most common nouns. Participants were then asked to construct a scene that includes all of the words in their password phrase, assuming that imagining a scene would improve the memorability of the assigned passphrase. However, their results were not very encouraging as, out of the eleven variations on passphrase and password conditions they tested, eight conditions showed greater successful login on first try when compared to *pp-nouns-instr*. This makes us think how well people can construct a memorable scene when just asked to imagine one without given any directions.

Bonneau et al. [12] demonstrated the brain's ability to learn and later recall random full 56-bit system-assigned secret. This 56-bit code was encoded as either 6 words or 12 characters. They found that 88% of users were able to recall their passphrase after 3 days of completion of the initial study after they had stopped using their security codes. In this study users choose their own password. After receiving a user's self chosen password, authors added a new field in the login page into which users must type a random security

code, which was displayed. With each login a one-third second delay was added to encourage users to type in the assigned security code from the memory. The main limitation of this study however was the quite long training period. The participants were required to login into a website 90 times over up to 15 days, which they did at an average rate of nine logins per day.

Blocki et al. [7] provided evidence that the password strengthening mechanism of Bonneau and Schechter [12] could be improved by adopting the Person-Action-Object (PAO) story mnemonic, using a rehearsal schedule in $1.5 \times increasingintervals$. They asked their participants to imagine a story based on a photo of a scene that was shown, a user-chosen famous person from a predefined list, and a randomly selected action-object pair that together served as their secret passphrase. An example mentioned in the paper was, Bill Gates—swallowing—bike on a beach. Following a spaced repetition schedule over a period of 100+ days, participants were asked to recall the action-object pairs (e.g., swallowing—bike) when prompted with the associated scene-person pairs (e.g., Bill Gates—beach). The study identified that most of the forgetting happened in the first test period (12 hours) and 89% of users who remembered their passphrase during the first test period successfully remembered them in every subsequent round.

Joudaki et al. [30] showed how to improve the usability of system-assigned passphrases using implicit learning techniques. Their system design utilizes two implicit learning techniques: contextual cueing and semantic priming. Contextual cueing is a 2-dimensional spatial arrangement of distractors, and semantic priming is displaying semantically-related prime words. Users were assigned a 4-word system assigned passphrase and each word in the passphrase is presented in a display surrounded by 31 semantically related words. In the user study conducted, the authors were able to improve recall rates and login times for 4-word system-assigned passphrases.

Al-Ameen et al. [2] proposed providing users with a series of cues to aid recall of system-assigned secret. They called their system *CuedR* and it allows the users to choose from multiple cues (visual, verbal, and spatial) the one that best fit their learning process. Their pilot study showed that this method holds promise as all users recalled their phrase after 1 week within three attempts. In CuedR, six keywords were randomly assigned to the user each from a distinct portfolio (e.g., animals, fruits, or vehicles), where each portfolio presents 26 keywords. To aid memorability, the scheme offered graphical, verbal, and spatial cues corresponding to each of the assigned keyword.

Mnemonic is a learning technique that helps improve recall of information, by encoding information by abbreviation to a rhyme, or a pattern. Jeyaraman and Topkara [29] proposed a system that automatically creates a mnemonics for a given system-assigned, text-based passwords to make them more memorable. The goal of their automatic mnemonic generation is to automatically generate natural language sentences that convey some news or a story thereby providing semantic content to encode a given password. However, they used a manually created corpus - Reuters Corpus Volume 1 (RCV1) for extracting the sentence that corresponds to the assigned password. Their experimental results were promising, suggesting that automatic mnemonic generation technique make text-password systems more usable. Woo et al. [63] used abbreviations of passphrases as mnemonics, which was created from the first letters of passphrase words. They proposed the use of mnemonics in two ways 1) hint-mnemonics and 2) guide-mnemonics. Hint-mnemonics can be used to improve recall. Here as a user creates a passphrase, the system creates a hint-mnemonic and stores it with the passphrase. For example, if a user choose a passphrase "The rainbow has seven colors" then the resulting hint mnemonic becomes "TRHSC". During authentication, the system prompts the user for their passphrase, and displays the hint-mnemonic. Hint-mnemonics improved recall by 30–36% after three days, and

by 51–74% after seven days, when displayed as user hints during authentication. Guide-mnemonics is used during passphrase creation. Here a guide-mnemonics is generated by randomly choosing letters from the English alphabet. The user is then prompt to generate a passphrase matching this mnemonic. For example, the system may generate a guide-mnemonic "TQBFJOTLD" and the user may input a matching passphrase like "the quick brown fox jumped over the lazy dog". Guide-mnemonics reduced the use of common phrases from 50% to under 5% when user were asked to generate passphrases, that match a given mnemonic. Kuo et al.'s study on vulnerability of mnemonic passwords in 2006 [34], showed that users base their mnemonic passwords on phrases that can be found on the Internet and their 400,000-entry dictionary cracked 4% of mnemonic password.

Doolani et al. [20] developed the *LociMotion*, to help users learn a 56-bit strong system-assigned password in a single session through the method of loci and a cup-and-ball game. Their dynamically-generated video clip contains a virtual tour of a house, that has twelve unique landmarks (also known as loci), such that each location is associated to a specific letter of the password. The idea behind the cup-and-ball game is to intercept a falling ball into a cup by moving the cup under the ball. To move the cup left and right, a user must type a specific chunk of their assigned password before the ball could hit the ground. A user study showed that *LociMotion* offers a recall success rate of 99%, 96%, and 81% after 1, 4, and 18 days, respectively.

2.3 GPT-2 and Language Modeling

Language model (LM) is a machine learning model that can predict based on the words already observed in the sequence, the probability of the next word in the sequence. Let (x_1, x_2, ... , x_3) be a sequence of tokens and we want to learn a probability distribution of

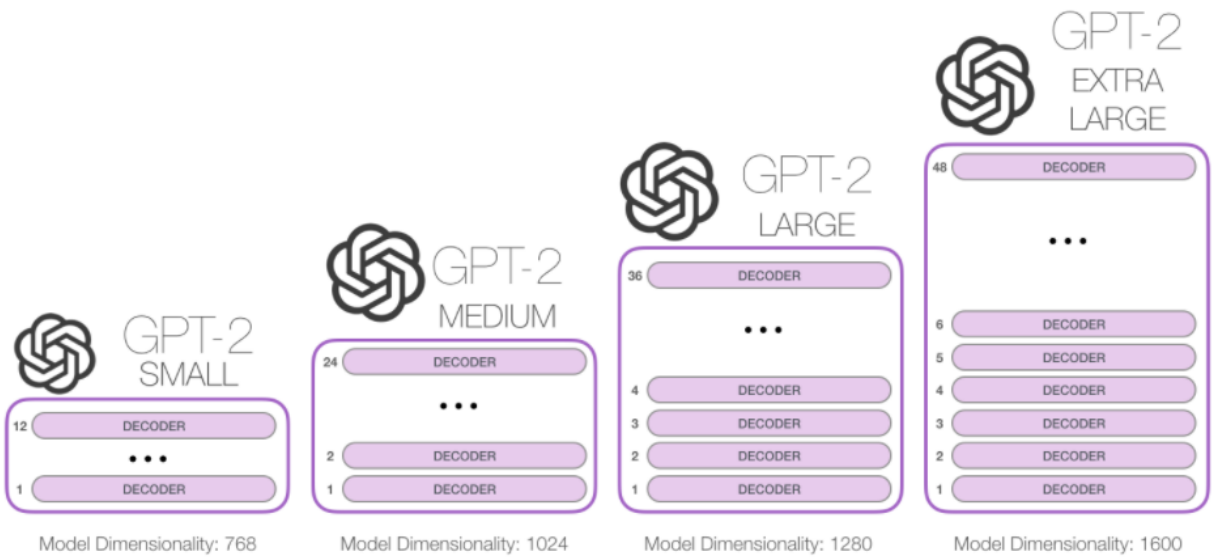

Figure 2.1: Variants of GPT-2 model [3].

the form (p_1, p_2, ... , p_3) [5]. The joint distribution is usually evaluated using the chain rule as:

$$P(x) = \prod_t P(x_t|x_{<t}) \tag{2.1}$$

where $x_{<t}$ represents all the tokens before the current token.

Smartphone keyboards are an example of LM as they suggest the next word based on what is currently typed. In February 2019, OpenAI released a paper [44] describing GPT-2, a model for text-generation based on the Transformer architecture [60] and trained on huge amounts of text data. With the objective to predict the next word, given all of the previous words within some text; the GPT-2 model was trained on a dataset of 8 million web pages. This dataset is called WebText and is a 40GB dataset [40]. In the staged release OpenAI has released four flavors of GPT-2 models. The small model with 117M parameters, a 345M version, a 762M and a 1.5B model.

Recurrent neural networks (RNN) are computationally expensive, and due to its sequential nature, training of RNN's cannot be parallelized. The transformer model overcomes this limitation by neither using recurrent nor convolutional neural networks and only using attention mechanisms, which has allowed for text-generation models that are much less computationally expensive [60]. The original transformer model is made up of a stack of encoder and decoder. However, GPT-2 is built using only transformer decoder blocks. The main distinguishing factors between the different GPT-2 model sizes is how high the stack of decoder block is, as shown in Figure 2.1.

Despite significant advances in language modeling, what is the best decoding strategy for text generation still remains an open question. The decoding methods, mainly include:

- Greedy search
- Beam search
- Top-K sampling
- Top-p sampling

Greedy search selects the word with the highest probability as its next word because of which it misses the high probability words hidden behind a low probability word as shown in Figure 2.2. Here the greedy search starts from word "The", and chooses "nice" as the next word as it has the highest probability between the words "dog","nice" and "car". Continuing this process, the algorithm generate the word "The", "nice", "woman" with a overall probability of $0.5 \times 0.4 = 0.2$.

Beam search expands upon the greedy search and reduces the above mentioned risk of missing hidden high probability word sequences. The Beam search in parallel expands

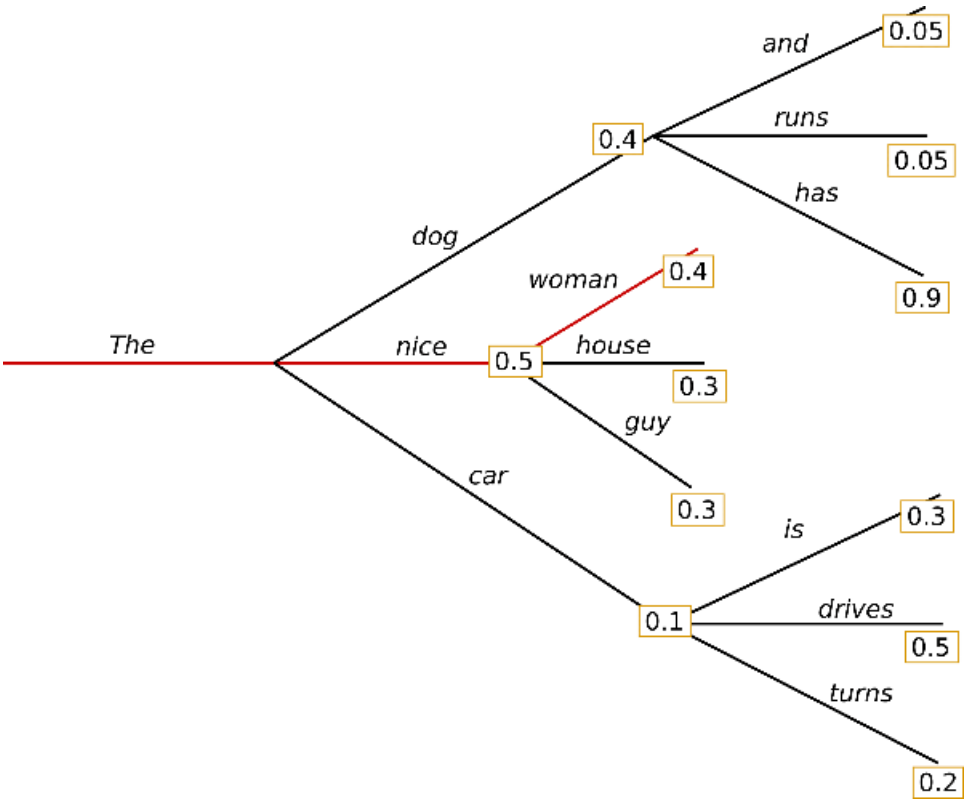

Figure 2.2: Greedy Search Decoding Strategy. Reproduced with permission [61].

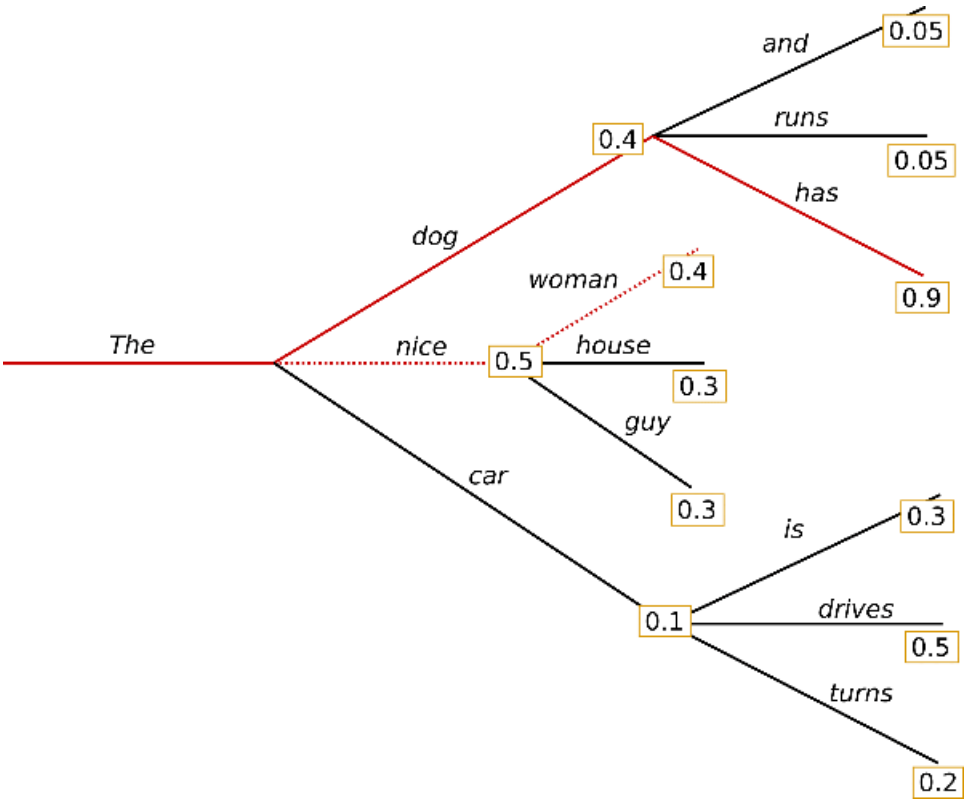

Figure 2.3: Beam Search Decoding Strategy. Reproduced with permission [61].

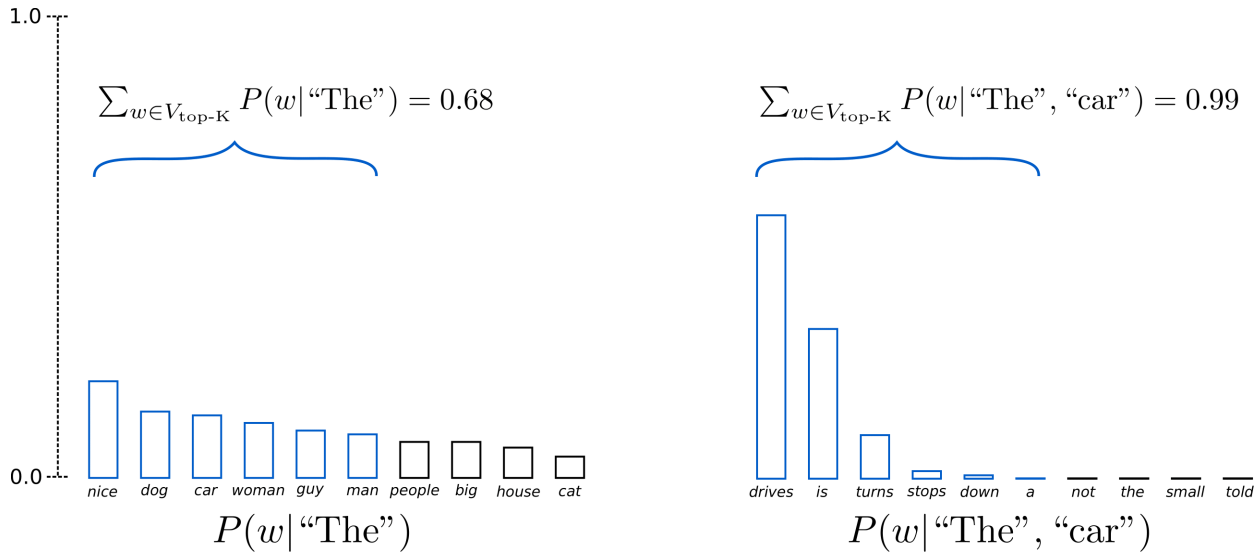

Figure 2.4: Top-K sampling. Reproduced with permission [61].

all possible next steps and keeps the k most likely where k is referred as the number of ‘beams’ and is denoted as num_beams. num_beams we search with is a user-specified hyper-parameter. Figure 2.3 illustrates Beam search with num_beams = 2. At time step 1, the beam search keeps track of two sequences “The”, “nice” and the sequence “The”, “dog”. At time step 2, the beam search reaches the word sequence “The”, “dog”, “has” which has a higher probability 0.4×0.9=0.32.

Top-K (top_k) sampling introduced by Fan et al. [21] is a simple yet very powerful sampling scheme. In Top-K sampling, the top k next words are filtered after which the probability mass is redistributed among only those k most likely next words. One of the reasons for GPT-2’s success in story generation is the adaptation of top_k sampling scheme. However, a limitation of Top-K sampling is that it does not dynamically adapt to the number of words that are filtered from the next word probability distribution because of which some words might be sampled from a very sharp distribution, whereas others from

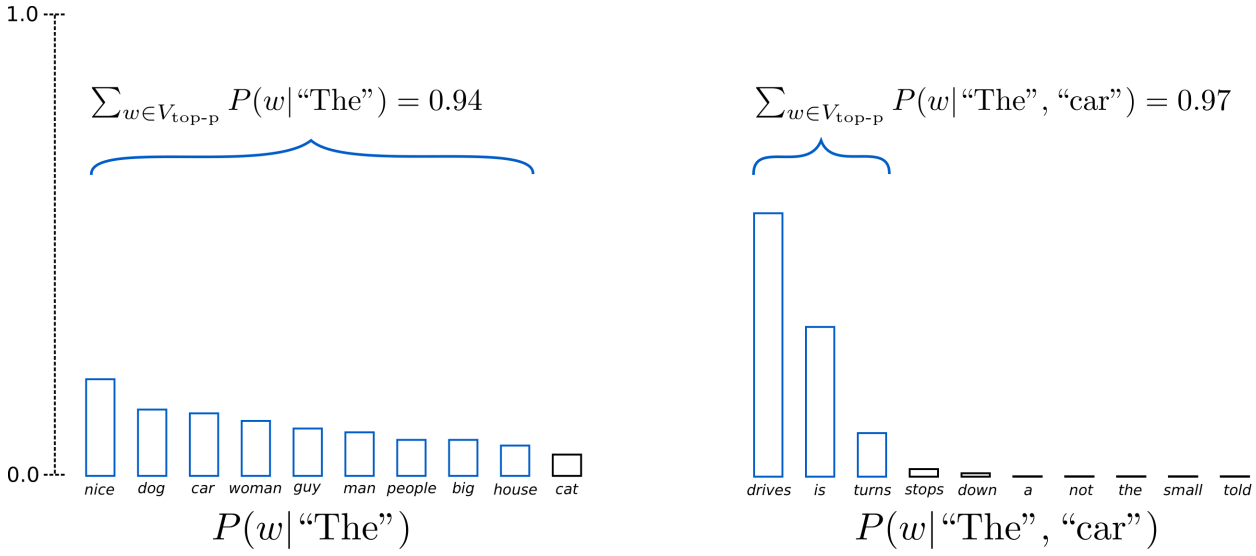

Figure 2.5: Top-p sampling. Reproduced with permission [61].

a much more flat distribution as shown in Figure 2.4. Here k is set to 6. In time step 1, Top-K eliminates the possibility to next words “people”, “big”, “house” and “cat”(left of Figure 2.4), which also seem like reasonable candidates. Here, the words are sampled from a flat distribution where every word seems to to be reasonable candidates. On the other hand, in time step 2 the Top-K sampling includes the arguably poor word choices “down”, “a” , “not”, “the”, “small” and “told” (right of Figure 2.4).

Instead of sampling only from the fixed most likely k words, Top-p (nucleus,top_p) sampling chooses from the smallest subset of the vocabulary whose cumulative probability exceeds the probability p. This way, the number of words in the set can dynamically increase and decrease according to the next word’s probability distribution [26] as shown in Figure 2.5.

2.4 Transfer learning

Deep neural network models needs to be trained on a very large dataset and the knowledge gained from this dataset is compiled as "weights" of the network. However, these large datasets are not always attainable which is a common challenge especially in NLP tasks. Traditional learning is isolated and occurs purely based on the datasets and the specific tasks. No knowledge gained through traditional learning is retained and transferred from one model to another. Training a neural network model is also very expensive, in terms of resources and time. Training a large and effective deep learning model that could make a plausible prediction requires millions of data points. These limitations of traditional learning motivated researchers to search for the possibility of knowledge transfer using large trained models [5]. Transfer learning is a machine learning method where a model developed for a task is reused as the starting point to solve a similar problem. A pre-trained model is a saved network that was previously trained on a large dataset. We can either use the pre-trained model as it is or use transfer learning to customize this model to a given task. Fine-tuning is a Sequential transfer learning that unfreeze few of the top layers of a frozen pre-trained model base and jointly train both the newly-added layers and the last layers of the base model. This allows us to "fine-tune" the higher-order feature representations in the base model, making them more relevant for the specific task.

The Python code to download the smaller version of GPT-2 model and the TensorFlow code to load the downloaded model and generate predictions was open-sourced on GitHub [41] by OpenAI. By utilizing transfer learning and building upon OpenAI's GPT-2 text generation model, Ressmeyer et al. [47] trained a model for the domain-specific task of generating similar tweets based on a given account which proved to be more successful than using an Long Short Term Memory(LSTM). Using transfer learning to convert GPT-

2 to their domain specific task, the model has shown to be able to generate seemingly human-authored tweets in response to the prompt that makes sense both grammatically and semantically. Budzianowski et al. [14] adapted the pretrained GPT-2 model to multi-domain task-oriented dialogues. Lee et al. [35] fine-tuned 345M GPT-2 pre-trained model for generating patent claims. With just few training steps they were able to generate patents that were coherent and complicated.

Chapter 3

Study Design

The goal of our work is to determine, will assigning a passphrase that resembles natural English sentences and are readable, pronounceable and are based on a vocabulary familiar to the user, help users better recall their assigned passphrase or not. For the purpose of generating a system-assigned passphrase that matches the above mentioned criterion's we first needed a vocabulary, familiar to the users. For the purpose of our study we obtained this familiar vocabulary from three popular books. We then trained a GPT-2 model on the familiar vocabulary, thereby generating passphrases that are readable, pronounceable, sentence like passphrases resembling natural English sentences.

In this section we discuss our two study conditions and how we generated passphrases for each of the study conditions. We conducted our two-part online user study of system-assigned authentication secret passphrases using Amazon's Mechanical Turk (M-Turk) framework. The study was approved by the Research Ethics Board (REB) at our institution. After participants consented to participate in the research study, we assigned each participant to a particular study condition. In the first part of the study, participants

were assigned a passphrase and were asked to memorize the assigned passphrase. Six hours later, participants were invited to return back, log in using their assigned passphrase, and complete a survey. We followed a space repetition schedule of one day for six days where participants were asked to recall the assigned passphrase. In this chapter, we give an overview of our study design and experimental conditions.

3.1 Study Overview

We recruited participants through Amazon's M-Turk. Participants needed to be at least 18 years old and we limited our data collection to participants from the United States or Canada. We compensated them 65 cents for completing the first part of the study and an additional 50 cents for completing each round of the second phase of the study (the compensation was prorated in accordance with the minimum wage in our university's location).

Our study is divided into two parts. Part 1 of the study is called as *Memorization Phase* and Part 2 of the study is called the *Recall Phase*. The *Recall Phase* is a multiple round phase, and is composed of six rounds. We refer to the first recall round as *Day#1 Recall*, second recall round as *Day#2 Recall* until the sixth recall round *Day#6 Recall*. During *Memorization Phase* we assigned participants a system generated passphrase. We asked them to memorize this assigned passphrase, and informed them that they will have to periodically return to the study and login to a dummy website using the assigned passphrase. The phase of the study were participants return back to check if they still remember the passphrase is the second part of the study which is referred as *Recall Phase*. During the *Recall Phase* participants were asked to login to a dummy website using the assigned passphrase. During the *Memorization Phase* participants were informed that this

was a memory study and they should not write down the words that we ask them to memorize. They were told that they will be paid for each completed rehearsal phase — even if they forgot the entire passphrase or words of the passphrase.

During the *Memorization phase*, we assigned participants a secret passphrase in one of two conditions, described in Section 3.2. After being assigned the secret, participants were required to login to a dummy website using the assigned secret. After three unsuccessful attempts, participants were displayed their secret. Six hours after completing the *Memorization phase* of the study, participants received an email through M.Turk asking them to return for the first round of *Recall Phase*(*Day#1 Recall*). To begin the Recall phase, participants were asked to login using their secret to the same dummy website. After three incorrect attempts, we showed participants their secret. In order to select the number of attempts threshold we followed the work of Al-Ameen et al. [2], the *CuedR* system which provided participants with three attempts to login. Once they had logged in, participants completed a survey about how they had remembered their assigned passphrase and their sentiments toward the assigned passphrase. This completes the *Day#1 Recall*. One day after the memorization phase, the *Day#2 Recall* study was conducted. It had the same procedure as *Day#1 Recall*, except that *Day#2 Recall* didn't have a survey. Only those participants who returned to complete *Day#1 Recall* were sent an invitation to participate in the *Day#2 Recall*. This procedure continued until *Day#6 Recall* which is the last day for the multiple round recall phase. The participants who reach till *Day#6 Recall* are those who haven't dropped out from the study along any of the recall rounds. Blocki et al. [7] reported that much of the forgetting of system-assigned passphrases happens in the first 12 hours. This is why we chose our *Day#1 Recall* to be 6 hours after the *Memorization Phase*, after which participants have to return every day for six consecutive days to complete the six rounds of *Recall phase*.

3.2 Conditions

We assigned participants to one of four experimental conditions, which are summarized in Table 3.1. The conditions varied in the passphrase assigned to the participants. Participants were unable to modify their assigned passphrase or to obtain a replacement. We required that participants enter words separated by spaces and in the same order they were assigned. All assigned passphrase were case-sensitive and they might resemble English like sentences depending on the passphrase condition.

Condition name	Length	Story	Assigned Passphrase
Random Passphrase	4	NA	information lays early site
Based on Familiar Vocabulary	5	Pride and Prejudice	Darcy knows my style perfectly
	6	The Adventures of Sherlock Holmes	Holmes investigations have always been indirect
	7	Alice in Wonderland	Alice was suppressed by wings of thunderstorm

Table 3.1: A summary of experimental conditions.

3.2.1 Random Passphrase Conditions

This condition is similar to *pp-sentence* by Shay et al. [51] which offered 30 bits of entropy. Participants were assigned passphrases of the form "noun verb adjective noun," where nouns, verbs, and adjectives are chosen from separate 181-word dictionaries as provided in [51]. We used the same 181-word dictionary provided by Shay et al. Although these passphrases were unlikely to make semantic sense due to the random selection of words, some of them could resemble natural English sentences. To identify the randomness of the passphrases generated in this condition, we computed the perplexity score. The perplexity score for the passphrase "information lays early site" is 214,885.92. A high perplexity indicates low probability which further indicates highly random sentence. Perplexity is the

inverse of the probability of the test set (as assigned by the language model), normalized by the number of word tokens in the test set. Assuming the test corpus has N tokens, w_1 ... w_N, and if the LM assigns probability $P(w_1 \ldots w_{i-N})$ to the test corpus, its perplexity, $PP(w_1 \ldots w_N)$, is defined as:

$$PP(w_1...w_N) = P(w_1...w_N)^{-\frac{1}{N}} \tag{3.1}$$

$$= \sqrt[N]{\frac{1}{P(w_1...w_N)}} \tag{3.2}$$

A LM with lower perplexity is better because it assigns a higher probability to the unseen test corpus. We used pretrained GPT as Language Model to calculate the Perplexity score to a sentence in *Random Passphrase Conditions*.

3.2.2 Passphrase Condition based on Familiar Vocabulary

In this condition we assign participants with a passphrase generated from a familiar vocabulary. For this we asked participants to select a story they are most familiar, from a list of three stories during the *Memorization phase*. We then assigned them with a system generated passphrase based on the vocabulary of the selected story. The three stories they chose from are *Alice in Wonderland, The Adventures of Sherlock Holmes , Pride and Prejudice*. These three are among the top 15 books downloaded from the Project Gutenberg library (which contains over 60,000 free eBooks) and they also have at least one movie based on the story.

We had to keep the length of passphrases based on Familiar vocabulary to be greater than four as shown in Table 3.1. This was because passphrases less than four, from the three stories selected appeared more as random words rather than pronounceable English

sentences and could not capture the style in which the story was written. We wanted to ensure that the passphrases generated from the stories not only are familiar in context of its vocabulary, but also resembles natural English sentences.

OpenAI's GPT-2 is a pre-trained, transformer-based language model that can be used for various NLP tasks such as text generation, translation, and data summarization. It is trained with the goal of predicting the next word given all previous words of some text [44]. By utilizing transfer learning and by fine-tuning GPT-2 text generation model, we sought to train a model for the domain-specific task of generating short stories in the style of a given book. Fine tuning GPT-2 has now become the best practice and have demonstrated best results in several language task [43],[35],[47]. In this work, our experiments are based on the 345M GPT-2 model. We forked Neil Shepperd's OpenAI's repo [54] which contains additional code to allow fine tuning the existing OpenAI model on custom datasets. For our experiment, we built a baseline model with temperature=0.7, top_p=0.9, top_k=40. Top-p sampling used in combination with Top-K, can avoid very low ranked words while allowing for some dynamic selection. Higher temperatures work better (e.g. 0.7 - 1.0) to generate more interesting text. The generated text are of length greater than 300 to accurately represent the context of the story on which the model is trained on and to generate realistic and coherent output.

Bonneau et al. [12] demonstrated user's ability to remember randomly assigned 56-bit (encoded as either 6 words or 12 characters) secret through spaced repetition. Bonk et al. [9] created StoryPass which asked users to create a sentence-like passphrase of at least 7 words in length. A 7-word long passphrase are difficult to be guessed due to the lack of tables for larger n-gram sizes. COCA [18] corpus the largest, genre-balanced corpus of American English has only up to 2, 3, 4, and 5 n-grams. Figure 4.4 on marginal guesswork shows that 2, 3 and 4-grams have lower marginal guesswork in bits compared to 5-grams.

Keeping these factors into consideration, for our study we choose passphrases that are 5, 6 and 7 words long.

To simulate the use of familiar vocabulary, we used three classic literary pieces: *Alice in Wonderland* (Lewis Carroll, 1965), *Pride and Prejudice*(Jane Austen, 1813), *The Adventures of Sherlock Holmes* (Arthur Conan Doyle, 1892). The books for these stories were downloaded from Project Gutenberg [1]. We used GPT-2 to generate 300+ words text for each of three books. We then cleaned the generated text to produce passphrases of 5, 6 and 7 words long. Table 3.2 shows the short story generated by GPT-2 from the book *Alice in Wonderland*. To produce passphrases from the GPT-2 generated short story, we performed text cleaning. To start with, each lines from the GPT-2 generated text was extracted. We then split each line into a sequence of words that forms a candidate passphrase until a punctuation - comma, colon, semicolon or period - is found. All those sequences, with word count less than 5 were discarded from the candidate pool of passphrase. The conjunction '*and*', definite article '*the*', indefinite article '*a/an*' were then removed. The word count of each candidate passphrase were then taken and those less than 5 or greater than 7 were removed from the candidate pool of passphrases. In the final step we randomly replaced personal pronouns (I, me, she, he, them, it) to character names in the book. This data cleaning operation generates passphrases of 5 , 6 or 7 words long. Table 3.3 shows the passphrases generated from Table 3.2. We then ranked the generated passphrases using conditional probability of n-grams. Assuming an attacker wants to guess all n-gram combinations, the best list for an attacker with unlimited resources would be the bigrams, because all of the word combinations that appear in bigram also appear in all the other n-grams. Before assigning a different user with a passphrase generated from the same book, we compute passphrase similarity score to ensure the newly assigned passphrase is

[1] https://www.gutenberg.org/

sufficiently different from the already assigned passphrases based on the same book. We used universal sentence encoder [17] in order to derive the semantic representation and the similarity of passphrases. We used the universal sentence encoder transformer based version 4, which is available at Google Tensorflow-hub [2]. For the generated passphrase in Table 3.3, Figure 3.1 and Figure 3.2 shows the sentence embedding and similarity scores respectively.

[2]https://tfhub.dev/google/universal-sentence-encoder/4

Alice was silent.

The Pigeon flew away in a straight line, and was just in front of it when it came: it was all but out of sight, and the moment Alice appeared on the other side, she was suppressed by the wings of the thunderstorm, and was only a few yards off the Queen's head.

Alice was not a bit hurt, and she flew away in a great hurry, from one end of the court to the other; and as the hall was very hot, and all the players were engaged in dancing, Alice appeared on the fiddle: she was a good deal frightened at the creature's appearing, but she did not dare to move, so ran down the other side of the court, and began dancing round and round the court, always in a hurry: among the raving players was another of the soldiers, and this time he was in livery, with the words "DRINK ME," done and ready.

"" Said the Queen, "who demands of it so much of anything, that he's always ordering people to laze about in the sun!"

And Alice was immediately ordered out of the court; for, you see, as she was only nine inches high, she could not afford to be any more cuddly: and, by the time she had been foraging for roots and leaves, she was getting up and walking off to the side of the court.

"Get to your places!" said the King, with a sudden emphasis. "What are you doing running?"

"I'm a farmer," said Alice: "I run them"

"There's no such thing!" said the King. "You're a serpent."

Alice was thoroughly puzzled, and was looking down at her hands in great curiosity. "What are they doing?" she asked the Gryphon.

"They're doing

Table 3.2: GPT-2 generated text for *Alice in Wonderland*.

Pigeon flew away in straight line
Gryphon was all but out of sight
moment Alice appeared on other side
Alice was suppressed by wings of thunderstorm
was only few yards off Queens head
Alice was not bit hurt
Cheshire Cat flew away in great hurry
from one end of court to other
as hall was very hot
all players were engaged in dancing
but Bill did not dare to move
so ran down other side of court
began dancing round round court
among raving players was another of soldiers
this time Dormouse was in livery
Alice was immediately ordered out of court
white rabbit was only nine inches high
Get to your places said King
What are you doing running
Theres no such thing said King
What are they doing Duchess asked Gryphon

Table 3.3: Passphrases generated for *Alice in Wonderland*.

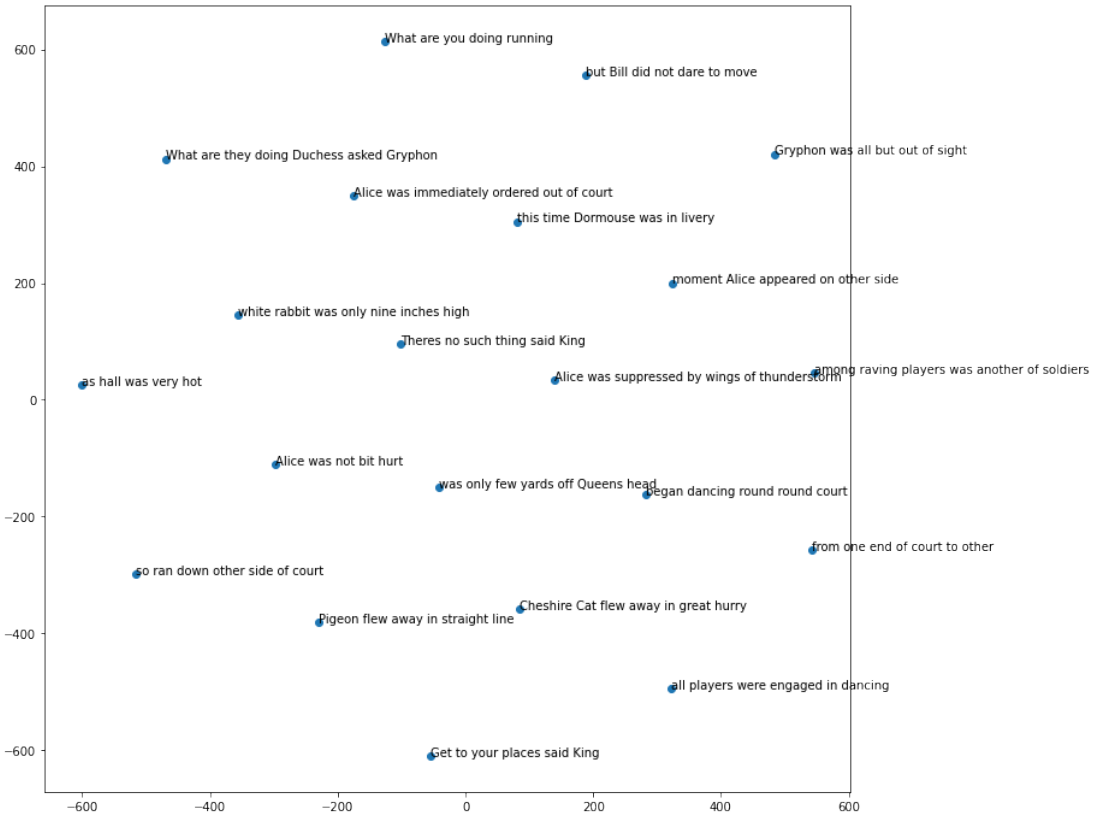

Figure 3.1: Sentence Embedding of passphrases in Table 3.3 using embeddings from the universal sentence encoder [17].

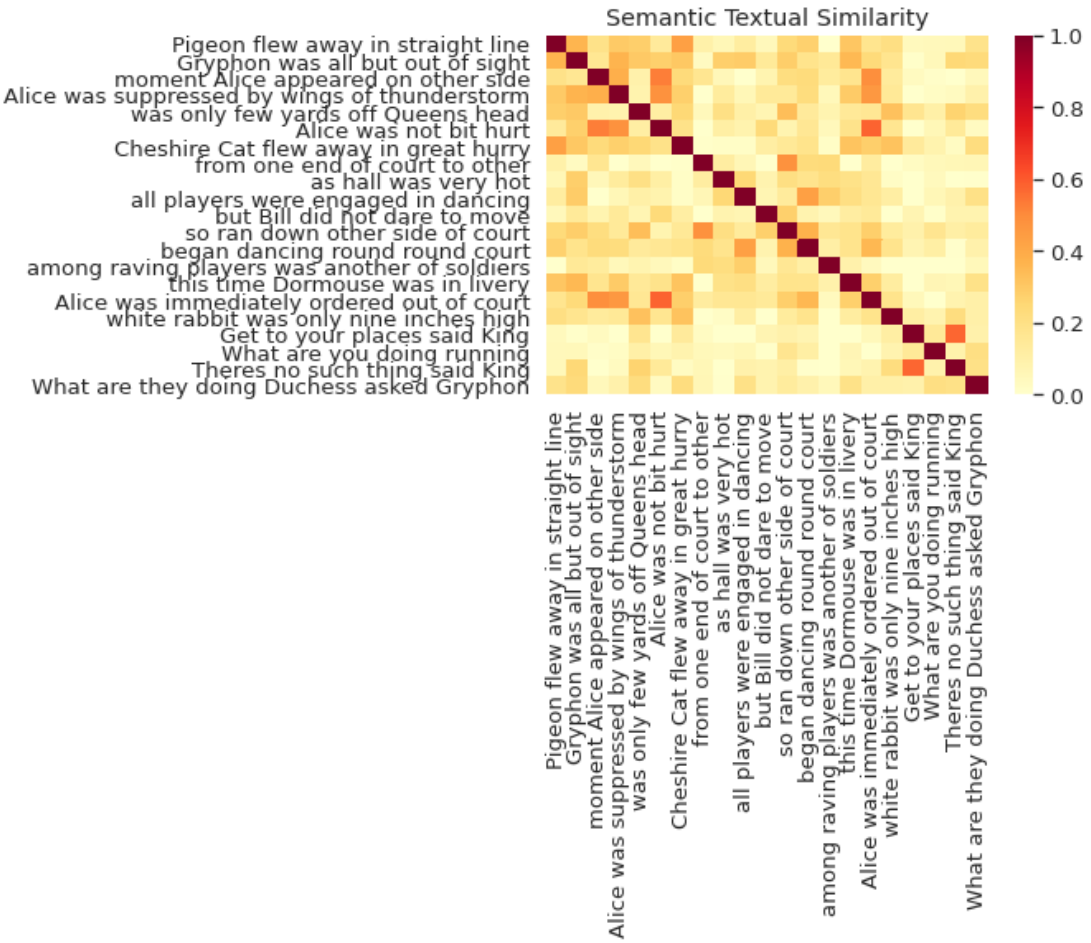

Figure 3.2: Sentence similarity scores of passphrases in Table 3.3 using embeddings from the universal sentence encoder [17].

Chapter 4

Word Selection and Security Goals

4.1 Word Selection

Bonneau et al. [13] studied the patterns in human choice of passphrases, based on Amazons now discontinued PayPhrase service. The study showed that a user tends to choose linguistically correct phrases because of which even a four words long passphrase probably has less than 30 bit of security. Their results suggest that users fail to choose phrases made of completely random words, but are influenced by the probability of a phrase occurring in natural language. Examples of linguistically correct phrases that occurs in natural language are "*all rights reserved*" or "*property of their respective*". Even though these are multi-word passphrases they are still insufficient to improve security because of their linguistic correctness. Sparell et al. [56] showed linguistically correct passphrases can be cracked using Markov process.

Blanchard et al. [39] proposed a method to guide users word choice when creating a passphrase, to make more memorable and more secure passphrases. Participants were

asked to select six words, in the order of their choice from a randomly generated set of words presented as an array. Their study indicated that passphrases created by choosing words from an imposed set of random words are more memorable than automatically generated passphrases.

POS tagging is the method of tagging a word with its corresponding part-of-speech like adjective, noun, verb, adverb, etc., following the language's grammatical rules that are further constructed on the basis of the context of occurrence of a word and its relationships with other words in a sentence. Rao et al. [45] investigated the effect of structural (grammatical) patterns on password security. They used Parts-of-Speech (POS) tagging to model the grammatical structures. Their password cracking algorithm automatically combines multiple words using their POS tagging framework to generate password guesses. They found a decrease in password search space due to the presence of grammar (also referred as tag-rule in the paper) and that this decrease can be as large as 50% for a password of length five words. An attacker aware of such a distribution can reduce their guessing effort by enumerating values for the grammatical structure and ignoring all other structures. Their work showed that two passwords of equal length may have different strength depending on its structure. Building on the idea of Rao et al. [45], we checked if our generated set of passphrases in Table 3.3 follow any common grammatical structure with each other and with the source text from which it is build.

For the book *Alice in Wonderland* after removing the unnecessary space there are 15 POS tags, including punctuation. We used SpaCy[1], an open-source software library, developed for performing simple to advanced Natural Language Processing tasks to perform POS tagging.

[1]https://spacy.io/usage/linguistic-features

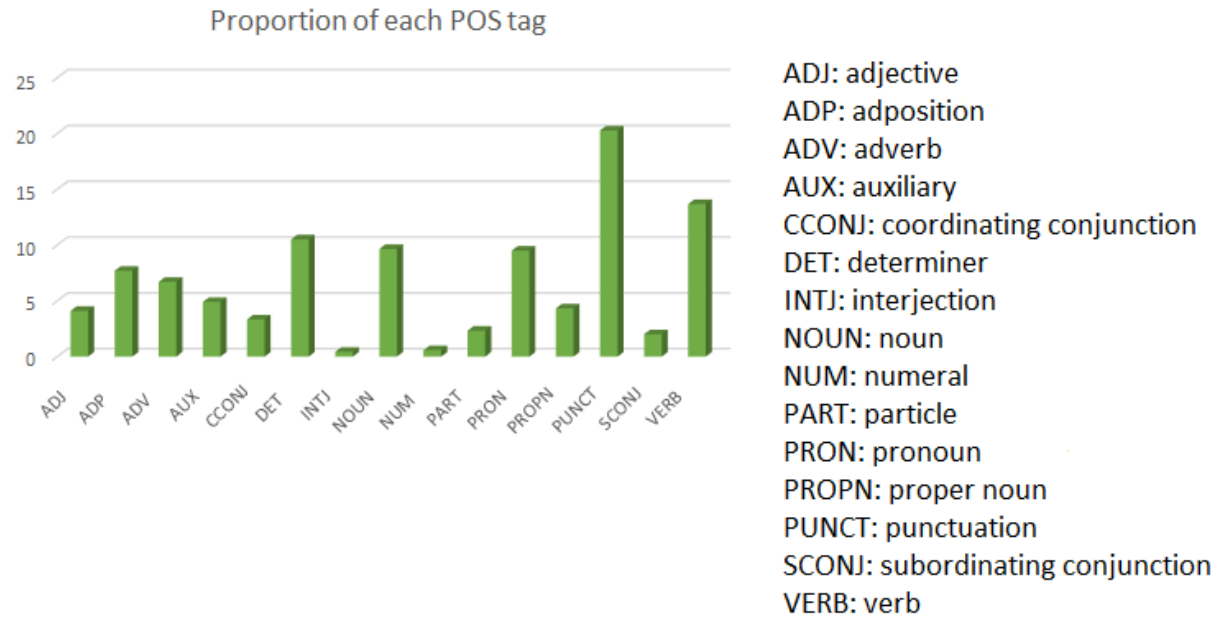

Figure 4.1: Proportion of POS-tag for the book *Alice in Wonderland*.

Let T be the set of all POS tags for *Alice in Wonderland*.

$$T = \{\text{ADJ, ADP, ADV, AUX, CCONJ, DET, INTJ, NOUN, NUM, PART, PRON, PROPN, PUNCT, SCONJ, VERB}\}$$

Figure 4.1 shows the proportion of each POS tag in the book *Alice in Wonderland*. Following punctuation's the next highest POS tag for the book is verb, followed which was determiner. We also checked the POS tag distribution of the short story shown in Table 3.2, which was generated by GPT-2 for the book *Alice in Wonderland*, based on which we have generated the passphrase. Figure 4.2 shows a similar POS-tag distribution for the generated story with the original book *Alice in Wonderland*.

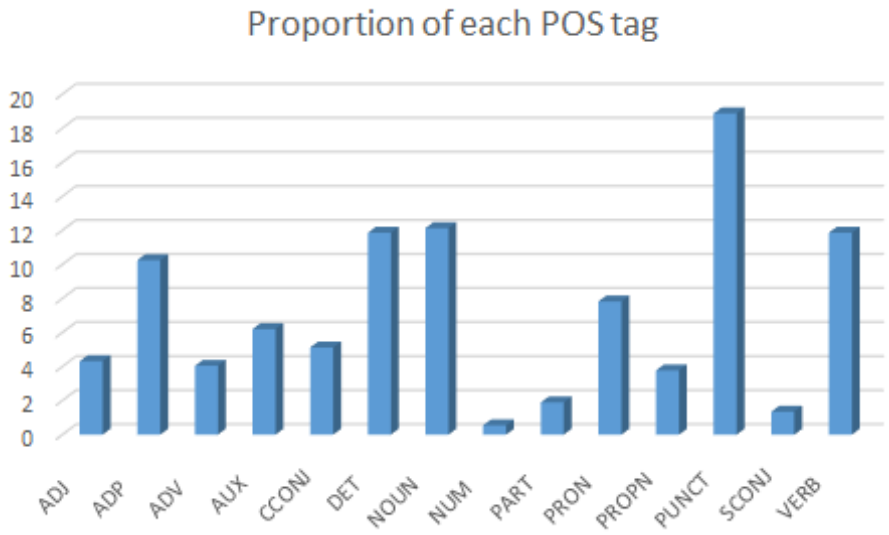

Figure 4.2: Proportion of POS-tag for the generated story in Table 3.2 using GPT-2 for the book *Alice in Wonderland*.

We removed the punctuation and created 5-gram, 6-gram and 7-gram of word and POS tag pairs for the book *Alice in Wonderland*. Given a sequence of words ($word_1$ $word_2$ . . .$word_n$), a POS tagger can output a sequence of tags, one tag per word (($word_1$, tag_1) ($word_2$, tag_2) . . . ($word_n$, tag_n)). Finally, we extracted the set of unique tag-rules from the word tag pair in each n-gram. In Figure 4.3 we group the tag-rules by their search space size. We observe that majority of tag-rules have a very small or negligible search space. For example, in tag rule of length seven (7-gram), 99% of tag rules have less than 2.8 bits of strength. This is because almost 99% of tag-rules have only one tag sequence. This implies that for 7-grams, only one sequence of words are generated by 99% of tag rules. Because of this limitation of our chosen vocabulary, we remove from Table 3.3 all those passphrases that don't have a tag-sequence that is either unique from each other or with the tag-rules from n-grams, thereby creating Table 4.1. This would make sure that

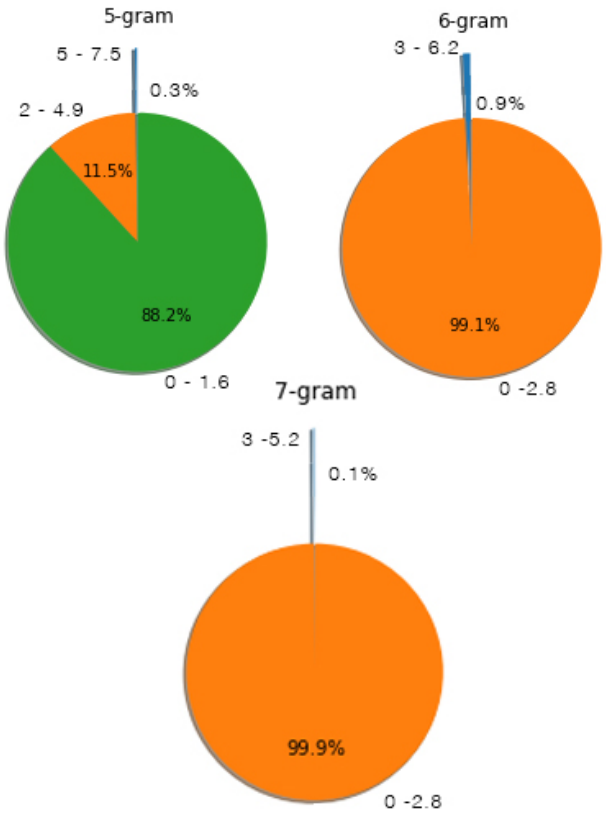

Figure 4.3: Tag-rules of length 5 to 7 grouped by the size of their search space (in bits) for the book *Alice in Wonderland*. The search space of a tag-rule, t_s, is the number of word sequences it generates, which in bits is $log_2 count(S(t_s))$. Numbers outside the pie chart indicate the range of bits. Numbers inside the pie indicate the percentage of tag-rules with those many bits [45].

our generated passphrase don't follow any specific tag-rule thereby offering an entropy of $log_2 14^7 = 26.6$ bits for a seven word long passphrase. The ideal scenario is to have the tag-rules divide the password search space evenly and with every search space offering high bits of strength. As observed, the passphrase "all players were engaged in dancing" and "What are you doing running" from Table 3.3 are missing from Table 4.1. This is because these two phrases had a tag sequence that was not unique and was similar with a tag-rule in the 6-gram and 5-gram respectively in the book *Alice in Wonderland*.

The presence of semantic patterns could reduce even further the search space of passphrases. Bonneau et al. [13] showed how the presence of semantic patterns in passphrases decreases guessing efforts to an extend that even a 4-word phrases can now be less than 30-bit of security. Bonk et al. [9] used a *Passphrase Ranking Algorithm*, based on n-grams, to estimate the number of guesses it would take to crack a user's passphrase. To analyse the semantic effects of a generated passphrase we created one dictionary for 2-gram, 3-gram, 4-gram and 5-gram corpus of *Alice in wonderland* with n-grams in each dictionary sorted by increasing order of frequency. We then computed the joint probability of each of the generated passphrases in Table 4.1 using 2-gram, 3-gram, 4-gram and 5-gram along with Laplace smoothing and generated a score for each passphrase as shown in Algorithm 1. During the calculation of joint probability for 2-gram, 3-gram, 4-gram and 5-gram, if the numerator of all terms in Equation 4.2 is one and is due to Laplace Transform, then the joint probability of that n-gram is discarded. We used Laplace smoothing to prevent a language model from assigning zero probability to these unseen events. Laplace smoothing add one to all the n-gram counts, before we normalize them into probabilities. All the counts that used to be zero will now have a count of 1, the counts of 1 will be 2, and so on.

An n-gram as the sequence of n words. A 2-gram (or bigram) is a two-word sequence

Pigeon flew away in straight line
Gryphon was all but out of sight
moment Alice appeared on other side
Alice was suppressed by wings of thunderstorm
was only few yards off Queens head
Alice was not bit hurt
Cheshire Cat flew away in great hurry
from one end of court to other
as hall was very hot
but Bill did not dare to move
so ran down other side of court
began dancing round round court
among raving players was another of soldiers
this time Dormouse was in livery
Alice was immediately ordered out of court
white rabbit was only nine inches high
Get to your places said King
Theres no such thing said King
What are they doing Duchess asked Gryphon

Table 4.1: Passphrases for *Alice in Wonderland* after removing non-unique tag sequences.

of words like "please turn", and a 3-gram (or trigram) is a three-word sequence of words like "The Three Musketeers". The general equation for an n-gram approximation to the conditional probability of the next word in a sequence is:

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-N+1}^{n-1}) \tag{4.1}$$

Given the bigram assumption for the probability of an individual word, we can compute the probability of a complete word sequence as:

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k|w_{k-1}) \tag{4.2}$$

where

$$P(w_k|w_{k-1}) = \frac{C(w_{k-1}w_k)}{C(w_{k-1})} \tag{4.3}$$

Algorithm 1: Score for generated passphrase

procedure BESTNGRAM PROBABILITY(C)

C = generated passphrase

a= probability of C using 2-gram

b= probability of C using 3-gram

c= probability of C using 4-gram

d= probability of C using 5-gram

score= MAX(a,b,c,d)

The passphrases in Table 4.1 are then ranked in increasing order of score, with rank one for that passphrase with minimum score. This procedure helps to identify any generated passphrases that have a high probability of being guessed and thus can be removed from the list of generated passphrases. Table 4.2 shows the generated passphrase arranged in

increasing order of their score. Rank 1 is for the passphrase with the minimum joint probability of their word sequence.

Rank 1: Alice was suppressed by wings of thunderstorm
Rank 2: among raving players was another of soldiers
Rank 3: Cheshire Cat flew away in great hurry
Rank 4: Pigeon flew away in straight line
Rank 5: began dancing round round court
Rank 6: Alice was immediately ordered out of court
Rank 7: white rabbit was only nine inches high
Rank 8: this time Dormouse was in livery
Rank 9: Alice was not bit hurt
Rank 10: from one end of court to other
Rank 11: was only few yards off Queens head
Rank 12: Gryphon was all but out of sight
Rank 13: so ran down other side of court
Rank 14: Theres no such thing said King
Rank 15: What are they doing Duchess asked Gryphon
Rank 16: moment Alice appeared on other side
Rank 17: but Bill did not dare to move
Rank 18: Get to your places said King
Rank 19: as hall was very hot

Table 4.2: Ranked Passphrases for *Alice in Wonderland*.

4.2 Security Goals

We use the book *Alice in Wonderland* to illustrate security goals. The book *Alice in Wonderland* has 2565($\approx 2^{11}$) unique tokens. The expected number of guesses for a passphrase of 5 words from this book is $(2^{11})^5/2 = 2^{54}$ which is $\gg 10^6$ and thus has negligible risk of a successful online attack [23].

Entropy is defined as the average amount of information that is generated by each character in a piece of text [50]. In terms of passwords/passphrases entropy can be considered a measure of the difficulty of guessing a password/passphrase [15]. For a passphrase of 5 words from *Alice in Wonderland*, the entropy in bits is $log_2 2565^5 = 56$ bit.

Marginal guesswork, is defined in [42] as "the optimal number of trials necessary to be guaranteed a certain chance of guessing a random value in a brute-force search". Marginal guesswork determines the amount of effort it would take to guess a certain number of passphrases under a known probability distribution. Bonneau et al. [13] used marginal guesswork model to measure the guessing difficulty of a distribution. The marginal guesswork to measure the maximum cost of determining the value of X , with a α probability of success is defined as [42]:

$$w_\alpha(X) = min\left\{ i \;\middle|\; \sum_{j=1}^{i} p_{[j]} \geq \alpha \right\} \tag{4.4}$$

Figure 4.4 shows the marginal guesswork for the book *Alice in Wonderland*. Marginal guesswork is calculated by ordering the n-grams that would be used by the attacker in order of the highest probability to the lowest probability, then summing the probabilities in descending order. This can determine at what point in guessing effort a certain percentage of n-grams would be guessed. The Success Rate(α) in Figure 4.4 is the probability

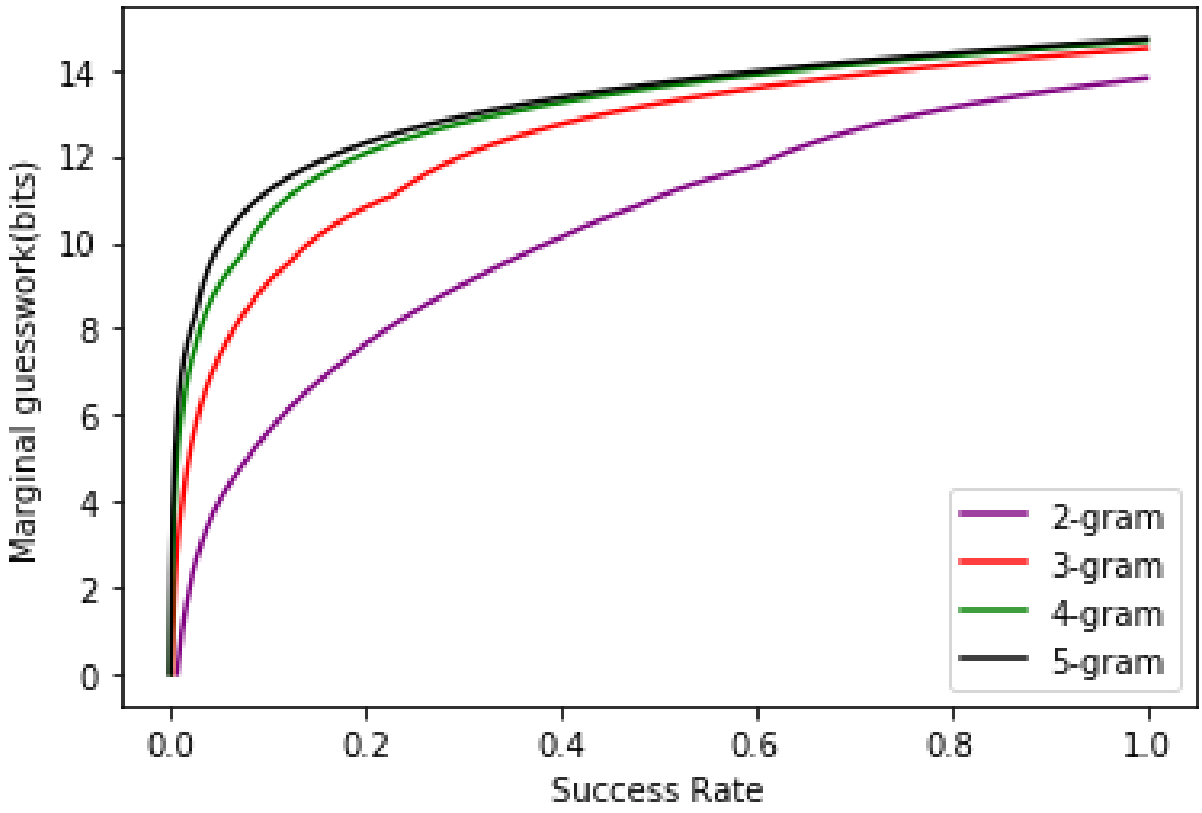

Figure 4.4: Marginal Guesswork for the book *Alice in Wonderland*.

of n-gram guessed for *Alice in Wonderland*. The marginal guesswork for *Alice in Wonderland* wouldn't withstand an unthrotted attack. However, we are not constructing our passphrases from n-grams of the selected book instead we are generating a new short story based on the selected book and then generating a passphrase based on that new short story. We are also ranking our passphrase based on the sequence probability as shown in Table 4.2. This helps us to blacklist and thereby eliminate those passphrases which are guessable according to n-grams. Our work is a proof-of-concept to a broader idea about using familiar vocabulary to improve memorability.

Figure 4.5 shows the marginal guesswork for the books *Pride and Prejudice* and *The Adventures of Sherlock Holmes*. Clearly the books *Pride and Prejudice* offers better marginal guesswork when compared to *The Adventures of Sherlock Holmes* and the book *Alice in Wonderland*.

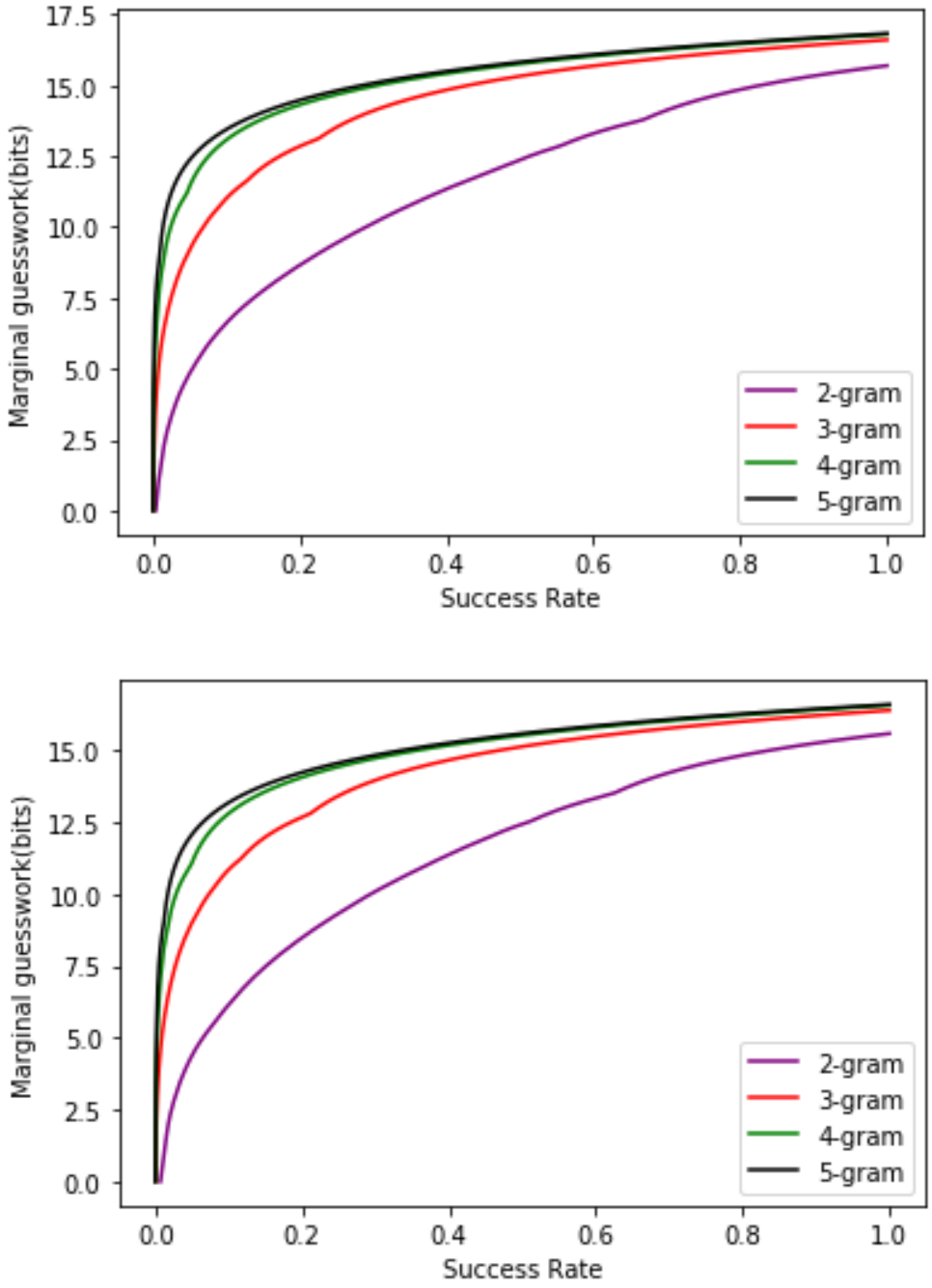

Figure 4.5: Marginal Guesswork for the book *Pride and Prejudice* (top) *The Adventures of Sherlock Holmes* (bottom).

Chapter 5

Results

In this section, we present the results of our study.

5.1 Study Data

One of the primary challenges in analyzing the results from any multi-phase user study is that, some participants have to be dropped from the study because they were unable to return for one of their rounds of recall phase in a timely manner, or have left the study. It is hard to determine how many of those dropped participants would have been able to remember their passphrase, if returned. We use the following notations from [7].

Notation: NumSuccessfulReturned*(C, i)* denote the total number of participants from study condition C who survived (passed) through recall $i - 1$ and returned for recall i. NumRemembered*(C, i)* denote the number of participants who remembered their passphrase during recall i with < 3 incorrect attempts. NumSurvived*(C, i)* denote the number of participants who also remembered their passphrase during every prior recall

Recall Condition	***i*=0**	***i*=1**	***i*=2**	***i*=3**	***i*=4**	***i*=5**	***i*=6**
Random	250	148, 94, 63.51%	89, 87, 97.75%	96, 82, 85.42%	101, 77, 76.24%	107, 74, 69.16 %	92, 67, 72.83%
Familiar Vocabulary	250	162, 105, 64.81%	94, 87, 92.55%	96, 70, 72.92%	99, 65, 65.66%	98, 61, 62.24%	88, 52, 59.09%

Table 5.1: NumSuccessfulReturned*(i)*, NumSurvived*(i)*, NumSurvived*(i)*/NumSuccessfulReturned*(i)*.

rounds. NumSuccessful*(i)* denote the total number of participants who failed recall i – 1 but passed recall i. Because the study condition C is often clear from the context, we will omit it from the notations. The conditional probability that a participant remembers their passphrase during recall i given that they have survived through recall i-1 and returned for recall i is

$$\frac{NumSurvived(i)}{NumSuccessfulReturned(i)}$$

Table 5.1 shows the conditional probability of random and familiar vocabulary conditions during each of the six recall rounds. Table 5.1 also shows how many participants who had never failed before returned in each recall rounds represented by NumSuccessful*(i)*. i= 0 in Table 5.1 indicates the *Memorization Phase*. We had 250 participants for the Random Passphrase condition and 250 participants in the familiar vocabulary condition. i= 1 indicates *Day#1 Recall* and i= 6 indicates *Day#6 Recall*. Table 5.2 summarizes the success and failure rate at each of the recall rounds. Droupout*(C,i)* indicates those participants who participated in recall i-1 but did not return for recall i. For $i = 1$ the previous phase is *Memorization Phase*, denoted as $i = 0$ in Table 5.2.

Random							
$i =$	No. of Participants	Num Remembered	Failed Login	Success Rate(%)	Failure Rate(%)	Drop out	Num Successful
0	250						
1	148	94	54	63.51	36.49	102	
2	136	101	35	74.26	25.74	12	14
3	130	106	24	81.54	18.46	6	11
4	122	110	12	90.16	9.84	8	10
5	119	102	17	85.71	14.29	3	2
6	106	98	8	92.45	7.55	13	7

Familiar Vocabulary							
$i =$	No. of Participants	Num Remembered	Failed Login	Success Rate(%)	Failure Rate(%)	Drop out	Num Successful
0	250						
1	162	105	57	64.81	35.19	88	
2	146	113	33	77.40	22.60	16	26
3	124	104	20	83.87	16.13	22	12
4	119	104	15	87.39	12.61	5	8
5	110	103	7	93.64	6.36	9	5
6	94	87	7	92.55	7.45	16	2

Table 5.2: Success and Failure rate of (top) Random Vocabulary and (bottom) Familiar Vocabulary.

5.2 Storage

During the first round of recall phase of the study, we asked participants if they wrote down their secrets, either on paper or electronically, reassuring them that their compensation would not be affected by their response. We consider a participant not to have stored his or her secret if the participant affirms of typing their passphrase from memory. The participants who haven't stored their assigned secret are most relevant when evaluating the memorability of secrets and hence we removed from our results those participants result who have reported to have recorded their assigned passphrase. Out of the participants who returned for first round of recall phase, 20 participants from the random vocabulary condition and 21 participants from the familiar vocabulary condition have reported to have recorded or stored their secret.

5.3 User Sentiment

In the first round of recall phase of the study, we asked participants to indicate their agreement, from "strongly disagree" to "strongly agree," with the statements "learning my passphrase was [annoying / difficult / fun]." The use of these three question to evaluate user sentiments was adapted from Shay et al. [51]. An overview of this result is shown in Figure 5.1.

5.4 Story Selection

As discussed in Section 3.2.2, we asked participants in the familiar vocabulary condition of the study to select from three stories, one story they are most familiar with. It is

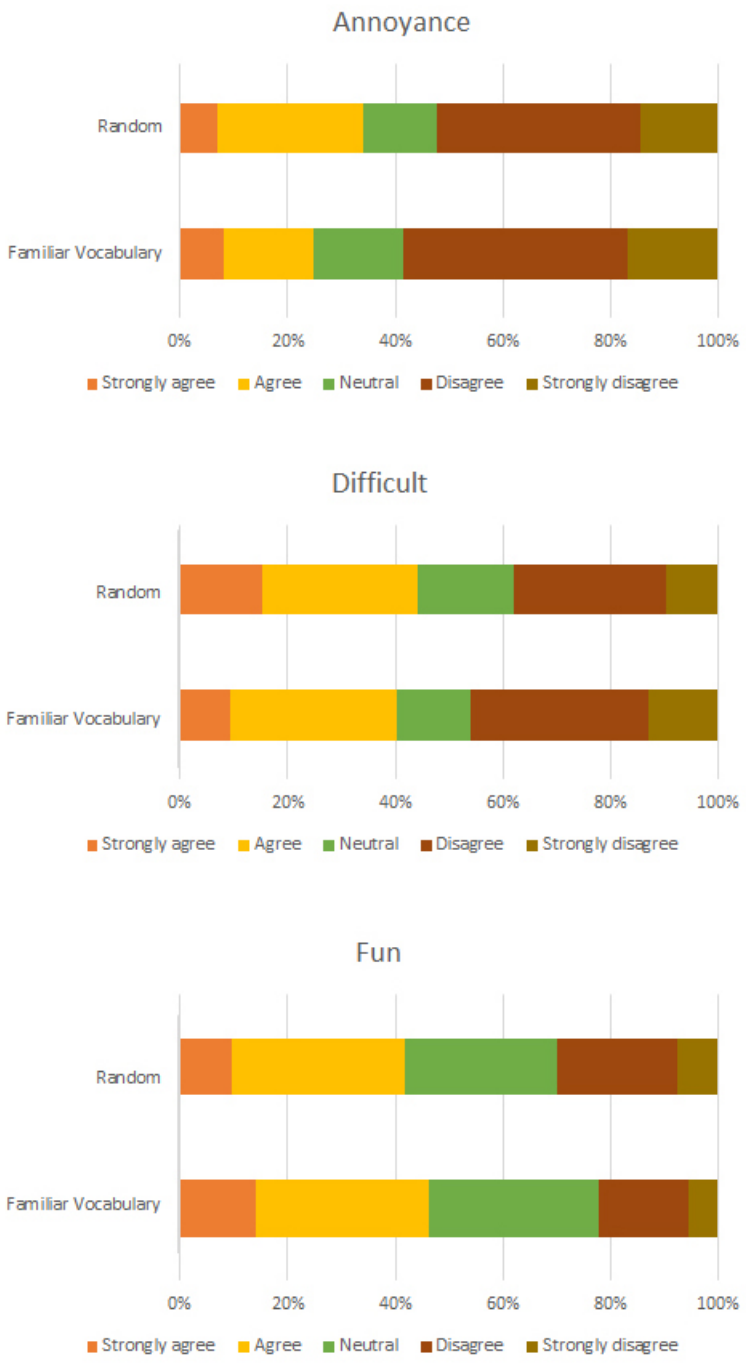

Figure 5.1: Response data on annoyance, difficulty, and fun.

based on the vocabulary of this selected story we assign a passphrase to the participants. Participants who selected the story *Alice in Wonderland* were assigned the passphrase *Alice was suppressed by wings of thunderstorm*, who selected the story *Sherlock Holmes* were assigned the passphrase *Holmes investigations have always been indirect* and, for those who selected the story *Pride and Prejudice*, they were assigned the passphrase *Darcy knows my style perfectly.* 67.90% participants selected the story *Alice in Wonderland*, 21.60% selected *Sherlock Holmes* and 10.50% selected the story *Pride and Prejudice.* During the first recall round we asked three questions related to the selected story, to the participants in the familiar vocabulary condition. The first question asked was if the participants have read the chosen story or have watched a related movie. The second question was if the participants imagined a scene related to the passphrase assigned or words in the passphrase and the third question was if the imagined scene was related to the story they had chosen. 22 participants reported that they haven't read any of these three book nor watched any movie related to the these stories. Since we couldn't provide a familiar vocabulary for these 22 participants, we removed their data from our study, leaving us 162 participants for *Day#1 Recall.* 53.09% of participants imagined a scene related to the passphrase assigned or words in the passphrase, out of these participants 67.44% imagined a scene related to the story they selected.

5.5 Typographic Error Analysis

Developed by Google, Bidirectional Encoder Representation from Transformers (BERT) [19] is a state of the art technique for natural language processing. BERT uses transformer architecture, an attention model to learn word embeddings. There are two versions of BERT. BERT Base has 12 encode layers with 768 hidden units and 12 attention heads,

and BERT Large, has 24 encode layers with 1024 hidden units and 16 attention heads. Bao et al. [6] worked on different ways to measure similarity between embedding vectors generated by BERT, mixing Euclidean distance with cosine similarity. We use a pre-trained BERT model [46] to compute a dense vector representation along with cosine similarity to measure the similarity between the assigned passphrase from familiar vocabulary conditions and the failed logins.

For the 5-word passphrase based on the book *Pride and Prejudice* there were six failed login attempts in total during the course of six recall rounds, as shown in Table 6.1 (The failed login column in Table 6.1 adds to six for *Pride and Prejudice*). Out of these six failed login attempts, it was just one login in which the particular participant completely forgot the passphrase and was not able to recall any one word in the 5-word passphrase with all three attempts. Since we had six failed login attempts and each login allows up to three attempts, we collected $6 \times 3 = 18$ incorrect ways in which the assigned passphrase was typed. Table 5.3 summarizes the similarity score of each of these with the assigned passphrase -"Darcy knows my style perfectly". Two of the incorrect passphrases had a similarity score greater than 95% with the assigned passphrase. These two passphrases are "darcy knows my style perfectly" and "Darcy matches my style perfectly" with a similarity score of 99% and 96% respectively with the assigned passphrase. We also included in Table 5.3, under the column 'completely dissimilar' the total number of attempts in which a incorrect passphrase with no single word in the assigned passphrase was typed.

Number of Incorrect Passphrases and their Similarity score with Assigned Passphrase											
Familiar Vocabulary	**> 95%**	**95% - 90%**	**89% - 85%**	**84% - 80%**	**79% - 75%**	**74% - 70%**	**69% - 60%**	**59% - 50%**	**49% - 40%**	**39% - 20%**	**completely dissimilar**
Pride and Prejudice	2	5	2	2	4						3
Sherlock Holmes	7	13	2	2	4	13	12	23	7		31
Alice in Wonderland	86	42	35	29	24	8	13	8	10	22	8

Table 5.3: Similarity Score between the assigned passphrase and the incorrectly typed passphrase.

For the 6-word passphrase based on the book *Sherlock Holmes* there were 38 failed login attempts in total during the course of six recall rounds as shown in Table 6.1. Out of these 38 failed login attempts, there were 10 login in which the particular participants completely forgot the passphrase and was not able to recall any one word in the 6-word passphrase with all three attempts. In 22 of the 38 failed login attempts, participants remembered the phrase - "Holmes investigations" from the passphrase "Holmes investigations have always been indirect". We collected 114 incorrect passphrases for the 6-word passphrase and their similarity score with the assigned passphrase is shown in Table 5.3.

For the 7-word passphrase based on the book *Alice in Wonderland* there were 95 failed login attempts in total during the course of six recall rounds as shown in Table 6.1. Out of these 95 failed login attempts, it was just one login in which the particular participant completely forgot the passphrase and was not able to recall any one word in the 7-word passphrase with all three attempts. We collected 285 incorrect passphrases for the 7-word passphrase and computed their similarity score with the assigned passphrase is shown in Table 5.3 and Figure 5.2.

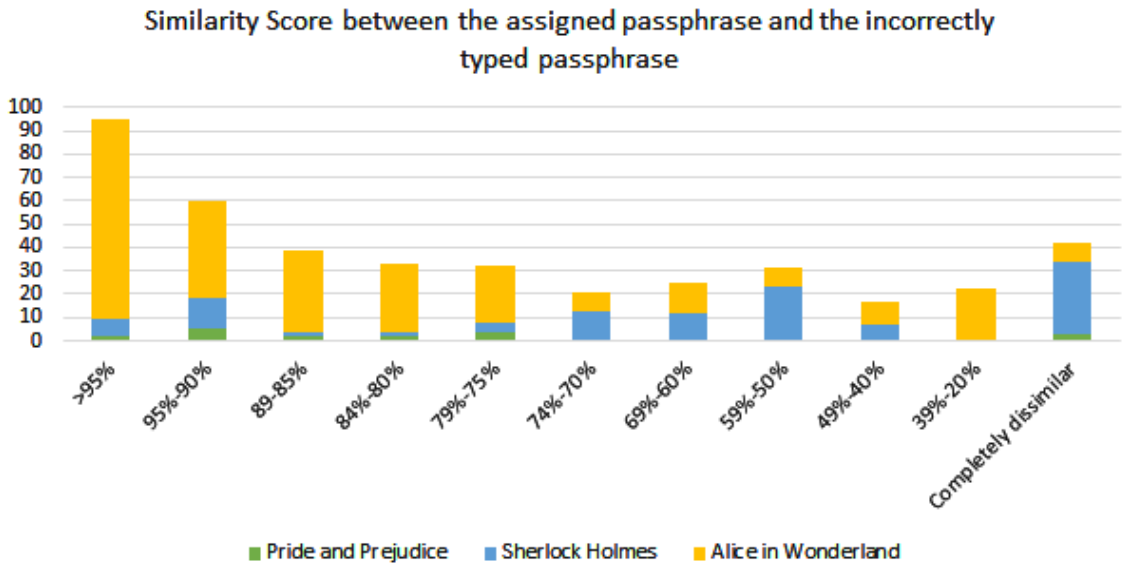

Figure 5.2: Similarity Score between the assigned passphrase and the incorrectly typed passphrase.

Chapter 6

Discussion and Conclusion

This work presents a new authentication method based on familiar vocabulary. With a user study we investigated the memorability of system-assigned passphrases from a familiar user vocabulary. The highlights of our work are presented below.

6.1 Discussion and Conclusion

We were able to identify a familiar user vocabulary. For conducting the experiment we chose three stories, of which we had to assume that participants would be familiar with at least one of them and would have either read a book related to one of these stories or have watched a movie related to one of the stories. Out of the participants who returned for *Day#1 Recall*, only 22 participants reported, to not have read a book or watched a movie related to the story they had chosen to construct the passphrase. We removed from our study the data from these 22 participants. This indicates for the remaining participants we were able to assign a system-assigned passphrase based on their familiar vocabulary.

However, we acknowledge the fact that these stories might not be the most familiar and the most recent vocabulary to the participants.

The central idea of our study is to investigate the impact of familiar vocabulary to generate passphrases that are memorable. For our study we selected three stories as this was the best way we could have known users familiar vocabulary in advance. In real-life scenarios this familiar vocabulary would be more personalized and could come from the ten books read, or from user-authored content such as sent emails and blogs.

Passphrases based on familiar vocabulary was reported to be more fun, less difficult and less annoying compared to random passphrases. However, conducting a paired t-test on the difficulty, fun and annoyance of random and familiar vocabulary condition we find that the difference is not significant. We believe that this has to do with the length of the passphrases assigned in the two conditions. Random passphrases were four word long whereas passphrases based on familiar vocabulary were five, six and seven words long depending on the story selected as shown in Table 3.1 (please refer to Section 3.2.2 for a discussion on passphrase length).

Table 5.2 indicates a similar success rate in memorizing a familiar vocabulary passphrase and a random passphrase, even though the word lengths do not match (please refer to Section 3.2.2 for a discussion on passphrase length). When conducted a paired t-test on the Success Rate of random and familiar vocabulary condition in Table 5.2 we find no significant difference. A paired t-test on the login success rate of random passphrase and familiar vocabulary passphrase indicates that the success rate of familiar vocabulary is not significantly greater than the random condition. The system-assigned passphrases created from a set of random words performed just as well in terms of memorability over system-assigned passphrases from a familiar vocabulary.

In order to identify whether failed participants in the familiar vocabulary passphrase are able to memorize their passphrase over time, we identified NumSuccessful*(i)* to denote the total number of participants from study condition *Familiar Vocabulary* who failed recall $i-1$ but passed recall i. As summarized in Table 6.1, the participants who failed the $i-1$ recall round, are eventually memorizing their passphrase at the next recall rounds for all three stories. We also investigated from Table 6.1 whether it is the participants who are failing in recalling the passphrase, the only ones dropping out from the study. However no such relationship exists.

Table 5.3 summarizes how similar are the incorrect passphrases typed by the participants with the assigned passphrase in the familiar vocabulary condition. There were incorrect passphrases in all three familiar condition which had no single word similar to the assigned passphrases. However, keeping that aside, for the story *Pride and Prejudice* even the least similar incorrect passphrase has a similarity score of 0.77 with the actual assigned passphrase of five words. There is a clear indication that as the length of the assigned passphrase increases the participants who failed one or more recall rounds are having less similar passphrase in their memory, than to the one assigned. For the story *Sherlock Holmes* there was no such incorrect passphrase that had a similarity score less than .40 (without considering the completely dissimilar cases) with the assigned passphrase. The least similar incorrect passphrase has a similarity score of 0.41 with the actual assigned passphrase of six words. However, for the seven word passphrase based on the story *Alice in Wonderland*, there were 22 incorrect passphrases that had a similarity score less than 0.39 with the assigned passphrase with the least similarity score of 0.24 for the incorrect passphrase "alice has her tea".

Our study showed that following a spaced repetition schedule, passphrases as natural English sentences, based on familiar vocabulary performed similarly to system-assigned

Pride and Prejudice							
i =	No.of Partici-pants	NumRemembered *(i)*	Failed Login	Success Rate(%)	Failure Rate(%)	Drop out*(i)*	NumSuccessful *(i)*
1	17	14	3	82.35	17.65		
2	16	14	2	87.5	12.5	1	2
3	14	13	1	92.86	7.14	2	1
4	14	14	0	100	0	0	1
5	11	11	0	100	0	3	NA
6	8	8	0	100	0	3	NA

The Adventures of Sherlock Holmes							
i =	No.of Partici-pants	NumRemembered *(i)*	Failed Login	Success Rate(%)	Failure Rate(%)	Drop out*(i)*	NumSuccessful *(i)*
1	35	18	17	51.43	48.57		
2	32	23	9	71.88	28.12	3	6
3	28	22	6	78.57	21.43	4	2
4	26	21	5	80.77	19.23	2	2
5	23	22	1	95.65	4.35	3	1
6	18	18	0	100	0	5	1

Table 6.1: Success and Failure rate of three stories in Familiar Vocabulary.

Alice in Wonderland							
i =	No.of Partici-pants	NumRemembered *(i)*	Failed Login	Success Rate(%)	Failure Rate(%)	Drop out*(i)*	NumSuccessful *(i)*
1	110	73	37	66.36	33.64		
2	98	76	22	77.55	22.45	12	18
3	82	69	13	84.15	15.85	16	9
4	79	69	10	87.34	12.66	3	5
5	76	70	6	92.10	7.89	3	4
6	68	61	7	89.70	10.29	8	1

Table 6.1: Success and Failure rate of three stories in Familiar Vocabulary (continued).

passphrases based on random common words. However, we believe that system-assigned passphrases based on familiar vocabulary has a potential to be explored further as a new authentication method.

6.2 Limitations

Our current methodology has its limitations. The length of the passphrase in the random passphrase condition was less than the familiar vocabulary condition. To be able to construct passphrases that resembled natural language sentences from familiar vocabularies, we had to include prepositions and connecting words, resulting in passphrases of five, six and seven words. Also, passphrases less than four, from the three stories selected could not capture the style in which the story was written. We wanted to ensure that the passphrases generated from the stories not only are familiar in context of its vocabulary, but also its

style. The random passphrase was only four words long as it did not contain propositions or any connection words. We could not increase the random passphrase length as that would further increase its entropy further adding to the already existing imbalance in the security in bits of both conditions.

Another limitation to our methodology is the difficulty to compare passphrases of equal bits of security. Since the passphrase in the random vocabulary conditions is of length four and is chosen randomly from a 181-word dictionary, it had an entropy of 30 bits. But, for the three stories selected, the maximum Marginal guesswork in bits obtained for 5-grams is 14.7 bits for Alice in Wonderland, 16.8 bits for Pride and Prejudice and 16.5 bits for The Adventures of Sherlock Holmes. To even attain a 20-bit marginal guesswork, we will need to combine at least ten books that can together generate 900,000 or more 5-grams. It was also not possible to decrease the entropy of random vocabulary conditions to match the security in bits of familiar vocabulary condition, as this would mean we will have to further reduce the number of words in random condition passphrase which would make the length of both the conditions further unbalanced.

For our study we are also forced to choose books whose scripts are available for free downloads. Gutenberg not only give the statistics of how many times a book was downloaded, but also give its script for free download. Even though we may get the most popular books from Amazon, the corresponding free download may not be available of that book. This made the choice of stories we could provide to the participants limited. The three stories participants chose from are *Alice in Wonderland*, *The Adventures of Sherlock Holmes*, *Pride and Prejudice*. Even though these three are among the top 15 books downloaded from the Project Gutenberg library (which contains over 60,000 free eBooks) and they also have at least one movie based on the story, these are relatively much older books published at least 120 years ago written in a very distinctive style. The choice of the book could

also have effected our study. It is also not necessary that the chosen story was the one the participants are most familiar with and it need not be the most recent vocabulary of the participants which could have also impacted the results.

6.3 Future Work

For future work, it would be interesting to see what could be done with more advanced natural language tools during a security analysis. We rank our passphrase based on the sequence probability as shown in Table 4.2 for the book *Alice in Wonderland*. This is to help blacklist and thereby eliminate those passphrases which are guessable according to n-grams. However, the ideal threshold to blacklist the generated passphrase to reach a sufficient theoretical security is yet to be determined and is proposed as a future work.

How to collect rich user familiar vocabularies needs further investigation. We chose three familiar stories to generate familiar vocabularies only for the sole purpose of conducting our user study as a proof-of-concept. For our study we selected three stories as this was the best way we could have known users familiar vocabulary in advance.

What remains to be compared is memorability of equal length passphrases from familiar vocabulary versus random words; for example, we could remove prepositions and stop words from familiar vocabulary sentences. Although this comparison would fall beyond the hypothesis of this work as the resulting passphrases will not look like natural English sentences, and thus could be undertaken as a future work.

References

[1] Pretrained models. `https://huggingface.co/transformers/pretrained_models.html`.

[2] Mahdi Nasrullah Al-Ameen, Matthew Wright, and Shannon Scielzo. Towards making random passwords memorable: leveraging users' cognitive ability through multiple cues. *ACM Conference on Human Factors in Computing Systems (CHI)*, pages 2315–2324, 2015.

[3] Jay Alammar. The illustrated gpt-2 (visualizing transformer language models). `https://jalammar.github.io/illustrated-gpt2/`, August 12 2019.

[4] Bander AlFayyadh, Per Thorsheim, Audun Jøsang, and Henning Klevjer. Improving usability of password management with standardized password policies. In *The Seventh Conference on Network and Information Systems Security.* —SAR-SSI 2012, 2012.

[5] Zaid Alyafeai, Maged Saeed AlShaibani, and Irfan Ahmad. A survey on transfer learning in natural language processing, 2020.

[6] Wei Bao, Hongshu Che, and Jiandong Zhang. Will_go at semeval-2020 task 3: An accurate model for predicting the (graded) effect of context in word similarity based

on bert. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, 2020.

[7] Jeremiah Blocki, Saranga Komanduri, Lorrie Cranor, and Anupam Datta. Spaced repetition and mnemonics enable recall of multiple strong passwords. In *22nd Annual Network and Distributed System Security Symposium*. NDSS, 2015.

[8] Hristo Bojinov, Daniel Sanchez, Paul Reber, Dan Boneh, and Patrick Lincoln. Neuroscience meets cryptography: Designing crypto primitives secure against rubber hose attacks. In *Proceedings of the 21st USENIX Security Symposium*, 2012.

[9] Christopher Lockett Bonk. Storypass: A system study for memorable secure passphrases. Master's thesis, 2014.

[10] Joseph Bonneau. The science of guessing: analyzing an anonymized corpus of 70 million passwords. In *Security and Privacy (SP), 2012 IEEE Symposium on. IEEE*, pages 538–552. IEEE, 2012.

[11] Joseph Bonneau, Cormac Herley, Paul C. van Oorschot, and Frank Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *IEEE Symposium on Security and Privacy*, pages 553–567. IEEE, 2012.

[12] Joseph Bonneau and Stuart Schechter. Towards reliable storage of 56-bit secrets in human memory. In *USENIX Security Symposium*, pages 607–623, 2014.

[13] Joseph Bonneau and Ekaterina Shutova. Linguistic properties of multi-word passphrases. In *USEC*, 2012.

[14] Pawel Budzianowski and Ivan Vulic. Hello, it's gpt-2 – how can i help you? towards the use of pretrained language models for task-oriented dialogue systems. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, 2019.

[15] William E. Burr, Donna F. Dodson, , and W. Timothy Polk. Electronic authentication guideline. Technical report, 2006.

[16] John B. Carroll. *Human cognitive abilities: A survey of factor-analytic studies*. Cambridge University Press, Cambridge, New York, 1993.

[17] Daniel Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, and Chris Tar. Universal sentence encoder. 2018.

[18] Mark Davies. Corpus of contemporary american english. `https://www.english-corpora.org/coca/`.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[20] Jayesh Doolani, Matthew Wright, Rajesh Setty, and S M Taiabul Haque. LociMotion: Towards learning a strong authentication secret in a single session. In *CHI Conference on Human Factors in Computing Systems (CHI '21)*. ACM, 2021.

[21] Angela Fan, Yann Dauphin, and Mike Lewis. Hierarchical neural story generation. In *Conference of the Association for Computational Linguistics (ACL)*, 2018.

[22] Dinei Florencio and Cormac Herley. A large-scale study of web password habits. In *Proceedings of the 16th international conference on World Wide Web*, pages 657–666. ACM, 2007.

[23] Dinei Florêncio, Cormac Herley, and Paul C. Van Oorschot. An administrator's guide to internet password research. In *USENIX Conference on Large Installation System Administration*, pages 35–52, 2014.

[24] Maximilian Golla and Markus Durmuth. On the accuracy of password strength meters. In *ACM CCS*, pages 1567–1582, 2018.

[25] S M Taiabul Haque, M N Al-Ameen, Matthew Wright, and Shannon Scielzo. Learning system-assigned passwords (up to 56 bits) in a single registration session with the methods of cognitive psychology. In *USEC*, 2017.

[26] Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. 2019.

[27] Jun Ho Huh, Hyoungschick Kim, Rakesh B. Bobba, Masooda N. Bashir, and Konstantin Beznosov. On the memorability of system-generated pins: Can chunking help? In *SOUPS*, 2015.

[28] Philip Inglesant and M. Angela Sasse. The true cost of unusable password policies: password use in the wild. In *ACM Conference on Human Factors in Computing Systems 2010*, pages 383–392. ACM, 2010.

[29] Sundararaman Jeyaraman and Umut Topkara. Have the cake and eat it too - infusing usability into text-password based authentication systems. In *21st Proc. ACSAC*, pages 473–482, 2005.

[30] Zeinab Joudaki, Julie Thorpe, and Miguel Vargas Martin. Reinforcing system-assigned passphrases through implicit learning. In *ACM Conference on Computer and Communications Security (CCS '18)*, pages 1533–1548. ACM, Oct. 2018.

[31] Mark Keith, Benjamin Shao, and Paul John Steinbart. The usability of passphrases for authentication: An empirical field study. *International Journal of Human Computer Studies*, 65(1):17–28, 2007.

[32] Saranga Komanduri, Richard Shay, Patrick Gage Kelley, Michelle L. Mazurek, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, and Serge Egelman. Of passwords and people: Measuring the effect of password-composition policies. In *CHI2011: Conference on Human Factors in Computing Systems*, 2011.

[33] Hennie Kruger, Tjaart Steyn, B. Dawn Medlin, and Lynette Drevin. An empirical assessment of factors impeding effective password management. *Journal of Information Privacy and Security*, 4(4):45–59, 2008.

[34] Cynthia Kuo, Sasha Romanosky, and Lorrie Faith Cranor. Human selection of mnemonic phrase-based passwords. In *SOUPS '06: Proceedings of the 2nd Symposium on Usable Privacy and Security*, pages 67–78. ACM, 2006.

[35] Jieh-Sheng Lee and Jieh Hsiang. Patent claim generation by fine-tuning openai gpt-2. 2019.

[36] Michael D. Leonhard and V.N. Venkatakrishnan. A comparative study of three random password generators. In *IEEE EIT*. IEEE, 2007.

[37] MattBishop and Daniel V. Klein. Improving system security via proactive password checking. *Computers & Security*, 14(3):233–249, 1995.

[38] Randall Munroe. xkcd: Password strength, 2012.

[39] Ted Selker Nikola K. Blanchard, Clément Malaingre. Improving security and usability with guided word choice. *34th Annual Computer Security Applications Conference – ACSAC*, 2018.

[40] OpenAI. Better language models and their implications. `https://openai.com/blog/better-language-models/`, February 14 2019.

[41] OpenAI. Gpt-2. `https://github.com/openai/gpt-2`, 2019.

[42] John O. Pliam. On the incomparability of entropy and marginal guesswork in brute-force attacks. In *in Progress in Cryptology-INDOCRYPT 2000*, 2000.

[43] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. Technical report, 2018.

[44] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 2019.

[45] Ashwini Rao, Birendra Jha, and Gananand Kini. Effect of grammar on security of long passwords. In *ACM Conference on Data and Application Security and Privacy (CODASPY'13)*, 2013.

[46] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bertnetworks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Associations for Computing Linguistics, 2019.

[47] Ryan Ressmeyer, Sam Masling, and Madeline Liao. Deep faking" political twitter using transfer learning and gpt-2. Technical report, Stanford, CA, USA, 2019.

[48] Daniel J. Sanchez, Eric W. Gobel, and Paul J. Reber. Performing the unexplainable: Implicit task performance reveals individually reliable sequence learning without explicit knowledge. *Psychonomic Bulletin Review 17*, pages 790–796, 2010.

[49] Stuart Schechter, Cormac Herley, and Michael Mitzenmacher. Popularity is everything: A new approach to protecting passwords from statistical-guessing attacks. In *Proc. HotSec'10*, 2010.

[50] C. E. Shannon. Prediction and entropy of printed english. *Bell System Technical Journal*, 30(1):50–64, 1951.

[51] Richard Shay, Patrick Gage Kelley, Saranga Komanduri, Michelle L. Mazurek, Blase Ur, Timothy Vidas, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. Correct horse battery staple: Exploring the usability of system-assigned passphrases. In *Proceedings of the eighth symposium on usable privacy and security*, page 7. ACM, 2012.

[52] Richard Shay, Saranga Komanduri, Adam L Durity, Phillip(Seyoung) Huh, Michelle L. Mazurek, Sean M Segreti, Blase Ur, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. Designing password policies for strength and usability. *ACM Transactions on Information and System Security*, 18(4):13, 2016.

[53] Richard Shay, Saranga Komanduri, Patrick Gage Kelley, Pedro Giovanni Leon, Michelle L. Mazurek, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. Encountering stronger password requirements: user attitudes and behaviors. In *SOUPS'10*, 2010.

[54] Neil Shepperd. gpt-2. `https://github.com/nshepperd/gpt-2`, month =mar, lastaccessed =March 2, 2021,.

[55] Adam Skillen and Mohammad Mannan. Myphrase: Passwords from your own words. Technical report, Montreal, Quebec, Canada, 2013.

[56] Peder Sparell and Mikael Simovits. Linguistic cracking of passphrases using markov chains. *IACR Cryptology ePrint Archive*, 2016.

[57] Dan Swinhoe. The 15 biggest data breaches of the 21st century, January 2021.

[58] Blase Ur, Felicia Alfieri, Maung Aung, Lujo Bauer, Nicolas Christin, Jessica Colnago, Lorrie Faith Cranor, Henry Dixon, Pardis Emami Naeini, Hana Habib, Noah Johnson, and William Melicher. Design and evaluation of a data-driven password meter. In *Proc. CHI*, 2017.

[59] Blase Ur, Patrick Gage Kelley, Saranga Komanduri, Joel Lee, Michael Maass, Michelle L. Mazurek, Timothy Passaro, Richard Shay, Timothy Vidas, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. How does your password measure up? the effect of strength meters on password creation. In *Proc. USENIX Security*, 2012.

[60] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *In Advances in neural information processing systems*, pages 5998–6008, 2017.

[61] Patrick von Platen. How to generate text: using different decoding methods for language generation with transformers. `https://huggingface.co/blog/how-to-generate`, March 18 2020.

[62] Matt Weir, Sudhir Aggarwal, Michael Collins, and Henry Stern. Testing metrics for password creation policies by attacking large sets of revealed passwords. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, pages 162–175. ACM, 2010.

[63] Simon S. Woo and Jelena Mirkovic. Improving recall and security of passphrases through use of mnemonics. In *Proceedings of the 10th International Conference on Passwords (Passwords)*, 2016.

[64] Moshe Zviran and William J Haga. A comparison of password techniques for multilevel authentication mechanisms. *The Computer Journal*, 36(3):227–237, 1993.

APPENDIX

Survey Questionnaire

Following are the survey questions asked to participants during the first day of the recall phase of the study.

1. Which of the stories are you most familiar with?
( ) Alice in Wonderland
( ) Pride and Prejudice
( ) Adventures of Sherlock Homes

2. Have you read this chosen book or watched a related movie based on the book (this question will not appear for the controlled group)?
( )Yes
( )No

3. Learning my passphrase was annoying.
( ) Strongly agree

( ) Agree

( ) Neutral

( ) Disagree

( ) Strongly disagree

4. Learning my passphrase was difficult.

( ) Strongly agree

( ) Agree

( ) Neutral

( ) Disagree

( ) Strongly disagree

5. Learning my passphrase was fun.

( ) Strongly agree

( ) Agree

( ) Neutral

( ) Disagree

( ) Strongly disagree

6. Describe anything you did to help yourself remember your passphrase.

7. How did you just enter your passphrase for this study (please be honest; you get paid regardless, and this will help our research)?

( ) I typed it in from memory

( ) It was stored in my browser

( ) I cut and pasted it from a text file

( ) I looked it up in the place I had recorded it earlier and then I typed it in

( ) I use a password manager that filled it in for me

( ) I prefer not to answer

( ) It was automatically filled in

( ) I forgot my passphrase

( ) Other

8. Did you imagine a scene related to the words or letters in your passphrase, to help you remember it?

( ) Yes

( ) No

9. Was this scene related to the book / movie chosen in question three.

( ) Yes

( ) No

10. The passphrase was easy to learn.

( ) Strongly agree

( ) Agree

( ) Neutral

( ) Disagree ( ) Strongly disagree

11. It was difficult to remember some or all of the words or order of the words in the assigned passphrase.

( ) Strongly agree

( ) Agree

( ) Neutral

( ) Disagree

( ) Strongly disagree

12. If you have any additional feedback about your passphrase or this survey, please enter your comments here.