What is in Your Password? Analyzing Memorable and Secure Passwords using a Tensor Decomposition

Youjin Shin
The State University of New York, Korea
Incheon, South Korea
Stony Brook University, NY, USA
youjin.shin.1@stonybrook.edu

Simon S. Woo*
Department of Applied Data Science
SKKU Institute for Convergence
Sungkyunkwan University, Suwon, South Korea
swoo@g.skku.edu

ABSTRACT

In the past, there have been several studies in analyzing password strength and structures. However, there are still many unknown questions to understand what really makes passwords both memorable and strong. In this work, we aim to answer some of these questions by analyzing password dataset through the lenses of data science and machine learning perspectives. We use memorable 3,260 password dataset collected from prior IRB-approved user studies over 3 years and classify passwords into three strength groups using online and offline attack limits. Then, we apply a *tensor decomposition* to analyze password dataset by constructing a 3rd-order tensor with passwords' syntactic and semantic features. In particular, we used PARAFAC2 tensor decomposition to uncover the main characteristics and features that affect password strength. We quantitatively identified the underlying factors that are more frequently observed in strong and memorable passwords. We hope that our finding can validate widely accepted advice for creating strong passwords and provide useful insights to design a better password suggestion system.

CCS CONCEPTS

• Security and privacy → Authentication; **• Computing methodologies → Factorization methods**.

KEYWORDS

Password, Authentication, Tensor Decomposition, PARAFA2, Classification

ACM Reference Format:
Youjin Shin and Simon S. Woo. 2019. What is in Your Password? Analyzing Memorable and Secure Passwords using a Tensor Decomposition. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313690

1 INTRODUCTION

Although there have been several attempts to find strong alternatives to textual passwords, passwords are still widely used for many applications to authenticate users due to simplicity and convenience. However, textual passwords require two seemingly conflicting security and memorability requirements in order to serve today's user authentication needs. That is, users need to create both memorable and strong passwords. However, there is a clear tension between memorability and security. Strong passwords such as random strings are difficult to remember, while memorable passwords are easy to be guessed. Therefore, creating both strong and memorable passwords is a challenging task for a user. We, as security researchers, have to help users create both memorable and strong passwords, based on qualitative evidence and quantitative measurements.

To address these issues, several research [7, 9, 10, 19, 23, 25, 33, 34] have tackled the problems of analyzing and understanding the detailed structures of passwords and have shed lights on how people create memorable passwords. Also, Ur et al. [30] have uncovered some of users' misconception about strong passwords and users' detailed strategy to create strong passwords through small-scale lab study. However, prior research has heavily relied on leaked passwords dataset, which does not provide information on what passwords are memorable. Also, a small-scale qualitative study lacks on generalizing results based on quantitative measures. In addition, there are limited research in understanding the structures of passwords using the latest machine learning and data analysis techniques to identify latent factors that influence the creation of strong and memorable passwords.

In this research, we aim to understand syntactic and semantic features that strongly influence the creation of both strong and memorable passwords, using a *tensor decomposition.* A tensor decomposition is widely exploited and adopted in data mining, statistics, bioinformatics, psychology, physics, computer vision, and signal processing areas as a powerful tool to model a wide variety of heterogeneous, multiaspect, and higher-order data [24]. Especially, a tensor decomposition can effectively extract useful latent information out of multiaspect data tensors. In this work, in order to complement prior research, we consider passwords as multidimensional tensors and perform a tensor decomposition to uncover and validate the characteristics of both memorable and strong passwords. We leverage the memorable password dataset collected from prior IRB-approved user study by Woo et al. [36] as a baseline password dataset. Furthermore, we measure password strength in guess numbers using Monte-Carlo sampling method [13] and classify them to weak, medium, and strong strength group based on online (10^6 guesses) and offline guessing limit (10^{14} guesses) [14]. This allows us to analyze the characteristics of both memorable and

*Corresponding Author

strong password structures using a tensor. Our key contributions are summarized as follows:

- We are the first to model passwords as tensors and apply a tensor decomposition to extract useful underlying features from passwords.
- We further provide the explainability on the syntactic and semantic features that predominately occur in strong and memorable passwords.
- Through distance analysis, we show and validate our results with findings from prior research.

Our quantitative finding supports that password length is the most dominating feature in explaining strong passwords. On the other hand, the predominant use of dictionary words are shown in weak memorable passwords. Our work provides the confirmation and validation of common advice in creating strong passwords, and can help guide users create both strong and memorable passwords.

2 RELATED WORK

Jakobsson and Dhiman [19] attempted to better understand the password structures. However, their research is limited in capturing diverse password patterns. Also, much research has investigated distribution of characters and types of mangling patterns [9, 10, 23, 34]. On the other hand, Ur et al. [30] conducted the small scale qualitative lab studies to uncover the rationale behind average users' misconception about weak and strong passwords. Although they provided very useful insights, their main focus is on the qualitative study. Rao et al. [25] found that people tend to use grammatical structures for long passwords and passphrases. They found that some Part-of-speech (POS) tags are more frequently used than others. In addition, Vera et al. [33] presented the first framework for automatically segmenting, classifying, and capturing the semantic essence of passwords. They employed the natural language processing techniques to effectively segment passwords into more meaningful semantic segments. However, their analysis is based on the leaked password datasets and cannot explain the specific factors influence the strength and memorability of passwords. From our understanding, no research efforts have made to systematically understand both syntactic and semantic latent factors that influence the construction of both strong and memorable passwords from a data science perspective. In this work, we employ a *tensor decomposition* to achieve these goals.

A tensor is a multi-dimensional array [24]. Due to its ability to express multimodal or multiaspect data, it is a very powerful tool in applications that inherently create such data. In mathematics, geometric vectors and scalars can be considered as the simplest tensors. A 1st-order tensor is a vector, a 2nd-order tensor is a matrix, and a 3rd-order tensor is a cube. In general, Nth-order tensor $\boldsymbol{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is represented by the outer product $\circ$ of N vector spaces as follows:

$$\boldsymbol{X} = a^1 \circ a^2 \circ \cdots \circ a^N, \tag{1}$$

where a^N is the vector in Nth dim. A rank (order) in a tensor indicates the number of components in the decomposed matrices and every tensor can be expressed as a sum of a rank-1 tensor [1, 22].

Expressing a tensor as a sum of a rank-1 tensor was first proposed by Hitchcock [17, 18]. Practical algorithms have begun to appear in the late 20th century. In 1970, Carroll and Chang [8] proposed a canonical decomposition (CANDECOMP) and Harshman [16] suggested parallel factor decomposition (PARAFAC), which is an extended version of 2-way factorization for higher-order data. Since CANDECOMP and PARAFAC have same concept, the CANDECOMP/PARAFAC decomposition formulated by Kiers [20] has been widely used. It decomposes Nth-order data into a linear sum of rank-1 tensor as described in Fig. 1.(a) and a 3rd-order tensor can be decomposed into three component matrices A, B, and C.

Harshman [16] suggested PARAFAC2 model to relax the limitation of CANDECOMP/PARAFAC model. Whereas CANDECOMP /PARAFAC decomposition requires the identical factors across a parallel set of matrices, PARAFAC2 requires only one identical factor mode (dimension) and relaxes the other factor matrices to vary. We use PARAFC2 for our password dataset analysis.

The Tucker decomposition is another commonly-used tensor decomposition, which was first introduced by Tucker [27] and refined later [28, 29]. It has a core tensor, which can be viewed as compression of original tenser $\boldsymbol{X}$. For a 3rd-order tensor, it decomposes a tensor into a core tensor and three matrices which have different ranks as shown in Fig. 1.(b).

Applellof and Davidson [6] pioneered the use of CANDECOMP/ PARAFAC model to extract information from a chemical system. Andersson and Bro [5] dedicated developing practical description and application of tensors. And Acar *et al.* [2, 3] was the first case to apply a tensor decomposition to data mining. They analyzed online chatroom data to understand how social groups evolved in cyberspace. They constructed a 3rd-order tensor with $user \times keyword \times time$ spaces. De Lathauwer and Vandewalle [12] applied the Tucker decomposition for dimensionality reduction in higher-order signal processing. Pioneers of the use of the Tucker decomposition in computer vision are Vasilescu and Terzopoulos [31]. They extended conventional Singular Values Decomposition (SVD) to N-mode SVD for a tensor. They also proved that it has significantly better performance than standard Principal Component Analysis (PCA) in image recognition [32].

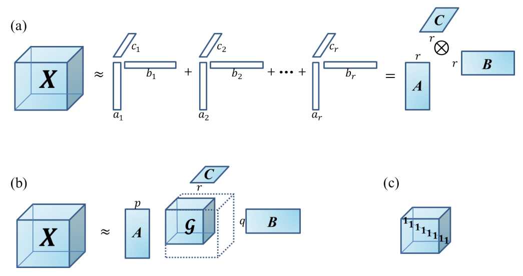


Figure 1: Examples of different tensor decomposition methods from a given 3rd-order tensor: (a) CANDECOMP/PARAFAC (CP) decomposition, where$X$ is an approximated sum of r rank-1 tensors. (b) Tucker decomposition, where X is a rank-(p, q, r) tensor and g is a core tensor. (c) A super-diagonal core tensor with ones in all diagonal values. If the core tensor of Tucker decomposition is a super-diagonal and all of the ranks are same, it can be thought as same as CANDECOMP/PARAFAC decomposition.

3 TENSOR DECOMPOSITION OVERVIEW

Tensor decomposition is one of the most effective unsupervised methods to extract features and characteristics of Nth-dimensional data and is capable of classifying data in the absence of training labels. Also, tensor decomposition can offer more accurate results by keeping the Nth-order structure of data without any feature mapping such as kernel trick as shown by other research [4, 11, 21].

Tucker decomposition: A given 3rd-order tensor $\boldsymbol{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, the Tucker decomposition is described in Fig. 1.(b). and defined as follows:

$$\boldsymbol{X} \approx \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} \mathbf{g}_{pqr} a_p \circ b_q \circ c_r = [\![\mathbf{g}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!], \tag{2}$$

where $a_p \in \mathbb{R}^{I_1}$ for p=1,. . ., P, $b_q \in \mathbb{R}^{I_2}$ for q=1,. . ., Q, and $c_r \in \mathbb{R}^{I_3}$ for r=1,. . ., R. g is a core tensor.

CANDECOMP / PARAFAC decomposition: As briefly discussed in Section 2, CANDECOMP/PARAFAC decomposition is a special case of the Tucker decomposition and can be constructed as follows:

$$\boldsymbol{X} \approx \sum_{r=1}^{R} \lambda_r a_r \circ b_r \circ c_r = [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!], \tag{3}$$

where R is rank, λ is weights $a_r \in \mathbb{R}^{I_1}$, $b_r \in \mathbb{R}^{I_2}$, and $c_r \in \mathbb{R}^{I_3}$, for r=1,. . .,R. For details of numerical tensor calculus, see Hackbusch's book [15]. CANDECOMP/PARAFAC decomposition can be considered as a special case of the Tucker decomposition where a core is super-diagonal and P = Q = R in Eq. 2. A super-diagonal is illustrated in Fig. 1.(c) [21].

PARAFAC2 decomposition: On the other hand, PARAFAC2 is a weak constraint version of PARAFAC. A given 3rd-order tensor, it is expressed by Acar and Yener [4] as below:

$$\boldsymbol{X}_k = \mathbf{A}\mathbf{D}_k\mathbf{B}^\top + \mathbf{E}_k, \tag{4}$$

where $\boldsymbol{X}_k$ represents a 3rd-order tensor, and $\mathbf{A}$ and $\mathbf{B}$ are the component matrices in the first and second modes, respectively. $\mathbf{D}_k$ is a diagonal matrix, whose diagonal elements are the kth row of the 3rd-component matrix $\mathbf{C}$. $\mathbf{E}_k$ contains residuals. Whereas PARAFAC must apply identical factors across the matrices, PARAFAC2 allows factors of matrices to be different but keeps only one factor invariance. Mathematically, it can be expressed by Acar and Yener [4] as following:

$$\begin{gathered} \boldsymbol{X}_k = \mathbf{A}_k\mathbf{D}_k\mathbf{B}^\top + \mathbf{E}_k \\ \textit{subject to } {\mathbf{A}_k}^\top\mathbf{A}_k = \Phi, k = 1, 2, \ldots, K, \end{gathered} \tag{5}$$

where ${\mathbf{A}_k}^\top\mathbf{A}_k = \Phi$ is orthonormal constraint and imposed to improve the uniqueness properties by Harshman [16] and all components are same as above.

4 DATASET

We leveraged the password data collected over three years from various authentication research by Woo et al. [35]. In the course of the study, these passwords were created and were successfully recalled after two days. The dataset includes more than 3,200 passwords. We used these successfully recalled passwords to understand features, which make passwords strong and memorable. First, we measured the strength of memorable passwords, using the Monte Carlo method by Dell'Amico, and Filippone [13]. We trained the guessing algorithm with a total of 21 million leaked passwords. Based on the estimated passwords' strength, we classified each password into the weak, medium or strong category, using the estimated number of guesses for online (10^6 guesses) and offline (10^{14} guesses) attack limits as boundaries between weak, medium, and strong password strength categories [14]. Number of passwords (%) in each strength group are summarized in Table 1. We found that majority of these collected passwords can withstand the online attacks, indicating these are fairly realistic passwords than now easily crackable leaked passwords from RockYou. After classifying memorable passwords into three different strength groups, we generated both *syntactic* and *semantic* features. Then, we applied the PARAFAC2 tensor decomposition to understand the latent syntactic and semantic features among different password strength groups.

Table 1: Memorable password dataset, categorized into three different strength groups

Strength Category	No. of Passwords (%)
Weak (guesses $< 10^6$)	109 (3.34%)
Medium ($10^6 \leq$ guesses $< 10^{14}$)	2,276 (69.82%)
Strong (guesses $\geq 10^{14}$)	875 (26.84%)
Total	3,260 (100%)

Syntactic Feature Generation: We generated 12 syntactic features for each password. First, we calculate the length of password (label: length) and the number of digits (label:D), special chars (label: S), and uppercase letter (label: UL). If there are more than three character classes among digits, special chars, uppercase, and lowercase letters, we define it as a 3class8, which is one of the widely used password composition policy. The number of class flips (change) is counted, when a character changes its class from one to another. Also, we counted the total number of flips (label: tFlips) as well as the following six sub flip cases: lowercase letter to special char (label: flipLS), lowercase letter to digit (label: flipLD), digit to special char (label: flipDS), digit to lowercase letter (label: flipDL), special char to lowercase letter (label: flipSL), and special char to digit (label: flipSD). We extract these features from each password.

Semantic Feature Generation: We used Vera et al.'s semantic segmentation parser [33] to segment the password and label each segment, which uses CLAWS7 tag set using Part-of-speech (POS) tagger by University Centre for Computer Corpus Research on Language (UCREL) [26]. In addition to CLAWS7 tag set, we have the following tags: "special*n*" and "char*m*", which represent the n consecutive special symbols and the m consequence characters. We extracted a total number of 114 different semantic features (tags). Through these initial tag generation process, we obtain a total of 67 features from 12 syntactic and 55 semantic features.

4.1 Pre-processing and Feature Refinement

Before actually constructing a tensor using the dataset with 67 features, we performed the following two pre-processing methods to make analysis much more reliable: 1) near zero variance removal and 2) normalization, which are popular pre-processing techniques.

1) Removing Near Zero Variance features: In one feature, if the majority of values do not vary much, then this feature cannot offer much useful information. And this would have an adverse effect on statistical analysis, since the variance is too small. These features with near zero variance values need to be eliminated prior to actual data analysis. In particular, we remove the near zero variance features for the following cases: the percentage of unique values is less than 10% and the frequency ratio is greater than 19 (95/5). After removing near zero variance features using these two methods, we obtained a total of 31 features (12 syntactic and 19 semantic features) out of initial 67 features, where 12 syntactic tags are *length, D, S, UL, 3class8, flipLS, flipLD, flipDS, flipDL, flipSL, flipSD*, and *tFlip*. And 19 semantic tags include *article, verb, adjective, pronoun, nn, nn1, nn2, np1, npm, num, char1, char2, char3, special1, special2, number1, number2, number3*, and *number4*.

2) Feature Normalization: Since features are distributed in different ranges at different scale (i.e.: length, number of symbols), we should normalize each feature individually. We applied min-max normalization, which is one of the well-known feature scaling techniques as shown in Eq. 6. Min-max normalization changes all values into the range between 0 and 1, where the minimum value is mapped to 0 and the maximum value is mapped to 1.

$$Z = \frac{x - min(x)}{max(x) - min(x)}, \tag{6}$$

where *min* and *max* are the minimum and maximum value from the given x.

5 EXPERIMENTS

We have three pre-processed datasets for each weak, medium, and strong password group with its 31 features. Since we have 109, 2276, and 875 passwords in *weak, medium*, and *strong* strength group respectively, the size of the *weak* dataset is 109×31, the *medium* dataset is 2276×31, and the *strong* dataset is 875×31, where 31 is the number of features in each password group. Since our dataset has different size of rows across the strength and only one equivalent feature dimension, it cannot be a cube and we need to build up a 3rd-order tensor in different row size as shown in the left side of Fig. 2. The final dimension of our tensor becomes I[3] × J × K = (109, 2276, 875) × 31 × 3 and each dimension represents passwords (I[3]), syntactic and semantic features (J=31), and levels of password strength (K=3) from *weak, medium*, and *strong* as shown in Fig. 2.

Next, we decompose this 3rd-order password tensor into component matrices, **A**, **B**, and **C**, using PARAFAC2 algorithm including orthogonal constraint. After decomposition, we obtain component matrices **A**, **B** and **C** containing, where **A** consists of three password group-to-factor sub matrices ((109,2276,875) × r) showing the factor dependency on each password across the strength matrices. And **B** represents the feature-to-factor matrix (31 × r) indicating how much each feature (syntactic and semantic features) is affected by each factor. Final **C** captures strength-to-factor matrix (3 × r) accounting for how much each factor affect passwords in each strength group.

Selecting an Optimal Rank r: When decomposing a tensor, it is critical to choose the optimal factor (rank) r, which is the number of factors to decompose a tensor. However, unlike 2-way factorization, there is no general straightforward algorithm to select the optimal r. Instead, a reconstruction error R_{error} is often used to find r, which is the difference between original tensor $\boldsymbol{X}$ and approximate tensor $\hat{\boldsymbol{X}}$ and it is calculated using the outer product of all matrices. It is defined as the Frobenius norm (Euclidean distance) $||\boldsymbol{X} - \hat{\boldsymbol{X}}||_{\mathbf{F}}$. In our case, from the 3rd-order tensor $\boldsymbol{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, our goal is to find the following: $R_{error} = \min_{\hat{\mathbf{X}}} ||\boldsymbol{X} - \hat{\mathbf{X}}||_{\mathbf{F}}$, where $\hat{\mathbf{X}}$ is generated in the same way in Eq. 5 and $||\boldsymbol{X}||$ is calculated as root-mean-square as follow:

$$||\boldsymbol{X}|| = \sqrt{\sum_{i_1=1}^{I_1}\sum_{i_2=1}^{I_2}\sum_{i_3=1}^{I_3} X^2_{i_1 i_2 i_3}}. \tag{7}$$

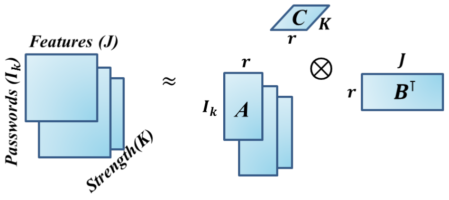


Figure 2: PARAFAC2 decomposition with password dataset, where the $\dim(I[K] \times J \times K) = (I[3] \times 31 \times 3)$. $I[1] \times J$, $I[2] \times J$, and $I[3] \times J$ are the password-to-feature matrix in weak, medium, and strong password group.

In addition, we use the alternating least squares (ALS) algorithm to find an optimal value, and we must set the condition to stop the iteration for finding optimal solution. Otherwise, there is a possibility of falling into an infinite loop. We set 500 as the maximum iteration and 10^{-4} as the error tolerance, which is the gap of residuals between the present and previous iteration. Since the optimal r makes the reconstruction error the smallest, we trace Frobenius norm residuals and observe the percentage of variance to find an optimum.

A factor r and the residuals are mostly in inversely proportional to each other. It means that we can have a smaller residual as r increases. However, unfortunately, the ALS algorithm can perform poorly, if the number of factors r is too large. Therefore, we should choose a small r. And the Elbow method is one of the popular methods to achieve this goal, where the value at the sharp bending part of norm residual graph can be the optimal r. We present our optimum r in the following section.

6 RESULTS

Optimal Rank (r): We analyzed at the variance of the reconstruction ratio (R^2_{ratio}=$||\hat{\boldsymbol{X}}||^2/||\boldsymbol{X}||^2$) of the estimated tensor $||\hat{\boldsymbol{X}}||^2$ against the original $||\boldsymbol{X}||^2$ by changing the number of r from 2 to 10, as shown in Fig. 3. The Y-axis in Fig. 3 shows the percentage of variance between R_{error} as a red line and R^2_{ratio} values as a blue line, where R^2_{ratio} has an inverse relation from R_{error}. The sharp bending point shown in a dotted red circle is an elbow, which is the optimal r = 2. When r = 2, we further analyzed the reconstruction error R_{error} and found that R_{error} is less than 1% and the dropping rate is 73.80% showing drastic decrease compared to other neighboring points. The average dropping rate of other points between 3 and 10 were 18.48%. However, PARAFAC2 does not guarantee the uniqueness of decomposition solution. Hence, instead of relaxing the factor invariance constraint across the parallel set of matrices, we calculated the average R_{error} and R^2_{ratio} of each r by repeating 10 times.

With the factor value r = 2, our goal is to discover specifically which features have an dominating effect on the password memorability and strength from component matrices **B** and **C**. When we decomposed matrices, we applied the non-negativity constraints

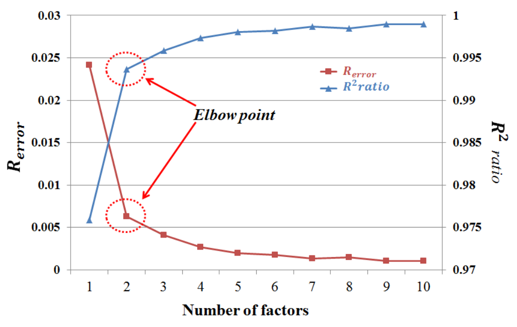


Figure 3: R_{error} (red line with the left Y-axis) and R^2_{ratio} by the number of factor (blue line with the right Y-axis). Each point is calculated as the average of 10 experiments. Red dotted circle indicates the elbow point.

to matrices **B** and **C**. Since non-negativity constraints not only mitigate the uniqueness problem but also provide positive values to component matrix after decomposition, it is more interpretable and practically used. Even though we mitigate the uniqueness issue by imposing non-negativity constraints, it does not eliminate the issue completely. Hence, we repeated the experiments 10 times and verified that all of them showed consistent patterns. Therefore, all results in this section are obtained from the one out of 10 representative experimental results.

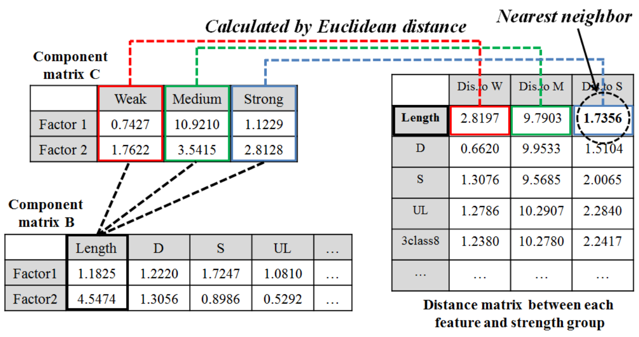
Result 1. Syntactic + Semantic Features: We obtained the component matrices **B** and **C** from PARAFAC2 decomposition. To compute the decomposition, we use the alternating least square (ALS) method to select an optimal rank *r*. Specifically, a component matrix **C** captures the factors that directly influence password strength, as shown in Fig. 2. The detailed results of **C** are presented in Table 2, where passwords in *weak* strength group is affected by *factor1* with 0.7427 and *factor2* with 1.7622. On the other hand, passwords in *strong* strength group is impacted by *factor1* with 1.1229 and *factor2* with 2.8128. In addition, a component matrix **B** captures the relationship between two factors and each password feature. The detailed results of matrix **B** are presented in Table. 3. Note that, in tensor decomposition, it is not important to understand and interpret the actual value of factors. However, the relative effect and relationships among features are more important as shown in other research [2].

Table 2: A component matrix C, which reflects the relationship between two factors and strength group.

	Weak	Medium	Strong
factor 1	0.7427	10.9210	1.1229
factor 2	1.7622	3.5415	2.8128

Table 3: A component matrix B, which reflects the relationship between two factors and each feature.

	length	**D**	**S**	**UL**	**3class8**
factor 1	1.1825	1.2220	1.7247	1.0810	1.0816
factor 2	4.5474	1.3056	0.8986	0.5292	0.5715
	flipLS	**flipLD**	**flipDS**	**flipDL**	**flipSL**
factor 1	0.8386	0.7471	0.8063	0.6525	1.8137
factor 2	0.4487	0.5724	0.3768	0.3674	0.7957
	flipSD	**tFlip**	**article**	**number4**	**np1**
factor 1	0.5066	0.9256	1.0885	1.4400	0.2527
factor 2	0.2423	1.0480	0.7758	0.6400	0.2567
	char1	**special1**	**nn1**	**number3**	**number1**
factor 1	0.8178	1.2084	1.5271	0.7259	1.0695
factor 2	0.4551	0.6389	0.7424	0.3235	0.5625
	number2	**special2**	**verb**	**adjective**	**nn2**
factor 1	0.8536	0.3212	0.4912	0.1142	0.6428
factor 2	0.4225	0.1440	0.2336	0.0784	0.2797
	pronoun	**nn**	**char2**	**char3**	**num**
factor 1	0.0774	1.5661	0.6900	0.9926	1.0850
factor 2	0.0638	0.8208	0.3338	0.4134	0.7775
	np				
factor 1	0.0678				
factor 2	0.1956				



Calculated by Euclidean distance
Nearest neighbor
Component matrix C
Component matrix B
Distance matrix between each feature and strength group

	Weak	Medium	Strong
Factor 1	0.7427	10.9210	1.1229
Factor 2	1.7622	3.5415	2.8128

	Length	D	S	UL	…
Factor1	1.1825	1.2220	1.7247	1.0810	…
Factor2	4.5474	1.3056	0.8986	0.5292	…

	Dis. to W	Dis. to M	Dis. to S
Length	2.8197	9.7903	**1.7356**
D	0.6620	9.9533	1.5104
S	1.3076	9.5685	2.0065
UL	1.2786	10.2907	2.2840
3class8	1.2380	10.2780	2.2417
…	…	…	…

Figure 4: Illustration of finding the nearest neighbor of each feature from the component matrices B and C

Next, we need to identify features that have strong influence to each password strength group. We used the Euclidean distance (ED) algorithm to find the distance from each feature (i.e. length, D) to three password strength groups, where ED is computed as follows:

$$ED_{str} = \sqrt{(B^{str}_{f1} - C^{ft}_{f1})^2 + (B^{str}_{fr2} - C^{ft}_{f2})^2}, \tag{8}$$

where $str\ is\ strength \in \{weak, medium, strong\}$, $f\ is\ factor$, and $ft\ is\ feature$. B^{str}_{f1} is the $factor1$ value in Table 3 and C^{ft}_{f1} is the $factor1$ value from each feature in Table 2. The actual example of computing *ED* for length is illustrated in Fig. 4.

Each row in Table 4 is the Euclidean distance calculated using Eq. 8 and gray colored value in each row is the nearest neighbor (min.) distance. The smaller distance means the stronger impact to each password. As shown in Table. 4, only *length* is consistently computed as *strong*. This means that password length is the consistent and dominating feature that is related to strong passwords. In other words, a longer password triumphs over a more complex but shorter password, where similar results are found in other research [36].

Table 4: Distance matrix between 31 features and password strength. Length is the dominant contributor in making stronger passwords.

	Dist. to Weak	Dist. to Medium	Dist. to Strong
length	2.8197	9.7903	1.7356
D	0.6620	9.9533	1.5104
S	1.3076	9.5685	2.0065
UL	1.2786	10.2907	2.2840
3class8	1.2380	10.2780	2.2417
filpLS	1.3170	10.5461	2.3811
filpLD	1.1898	10.5982	2.2717
filpDS	1.3868	10.5982	2.4564
filpDL	1.3977	10.7479	2.4903
flipSL	1.4426	9.5121	2.1321
flipSD	1.5381	10.9244	2.6434
tFlip	0.7373	10.3017	1.7758
article	1.0453	10.2141	2.0373
number4	1.3212	9.9150	2.1958
np1	1.5832	11.1626	2.7002
char1	1.3092	10.5641	2.3774
special1	1.2160	10.1370	2.1756
nn1	1.2866	9.8021	2.1095
number3	1.4388	10.6909	2.5208
number1	1.2434	10.2921	2.2509
number2	1.3442	10.5394	2.4054
special2	1.6721	11.1310	2.7866
verb	1.5492	10.9418	2.6555
adjective	1.7972	11.3481	2.9145
nn2	1.4859	10.7834	2.5782
pronoun	1.8241	11.3876	2.9411
nn	1.2507	9.7425	2.0407
char2	1.4294	10.7220	2.5165
char3	1.3717	10.4094	2.4029
num	1.0425	10.2169	2.0356
np	1.7057	11.3572	2.8218

Result 2. Semantic features only: We found that the dominant contributor to strength is *length*. However, we still need to uncover what other semantic features can influence on both memorable and strong password creations. As we strongly hypothesize that semantic aspects would contribute in memorability, we conducted an additional experiment to examine semantic features other than length and syntactic features. In this experiment, we excluded syntactic features and only used 19 semantic features for a tensor decomposition.

Among 19 semantic features, *char1* was classified as *strong* most of the times. And *number4*, *nn1* (singular common noun, e.g. book), and *number2* (two consecutive digits: e.g. 37) was classified as *strong* 8 times. These four features can be considered as dominant contributors to generating strong passwords. Whereas *verb*, *adjective*, *nn2* (plural common noun, e.g. girls), *np* (proper noun and neutral for number, e.g. IBM), and *num* (every number, not digit, e.g. three) never appeared for *strong* strength group. Also, *np1* (singular proper noun, e.g. London) and *pronoun* were classified as *strong* only once.

From 19 semantic features, 10 features were based on dictionary word and 9 features non-dictionary word such as *char1*. Only one dictionary word feature (*nn1*) was shown to influence to strong password, as shown in Table. 5. It indicates dictionary words are rarely observed in strong passwords. It indirectly captures that users are making an effort to use non dictionary words to create strong passwords. If dictionary based words occur, they tended to be mostly *nn1* in strong passwords, which can be a crucial factor to improve the memorability of their passwords.

Table 5: Distance matrix between semantic features and password strength: dominant contributors are *number3, char1, nn1,* and *number2*. Most dictionary words features are not close to *strong* strength group. Only *nn1* is classified as *strong* among all dictionary word features.

	Dist. to Weak	Dist. to Medium	Dist. to Strong
article	2.6494	1.5463	2.4830
number4	0.2515	0.8782	0.1755
np1	1.4964	0.9686	1.4219
char1	0.4515	0.6565	0.2878
special1	1.3039	0.2056	1.1374
nn1	0.8761	1.0084	0.7335
number3	0.0634	1.1711	0.2501
number1	0.7042	0.4169	0.5257
number2	0.3244	0.7847	0.1781
special2	0.3199	1.4258	0. 4919
verb	0.2562	1.3524	0.4155
adjective	0.2632	1.3565	0.4193
nn2	0.3561	1.4520	0.5147
pronoun	0.2326	1.3387	0.4060
nn	1.1882	0.9352	1.0200
char2	0.1801	1.2826	0.3484
char3	0.3326	1.4390	0.5054
num	2.7266	1.6237	2.5604
np	1.6317	1.0366	1.5519

7 DISCUSSION AND LIMITATION

In summary, the purpose of our research is to identify the features to make a password strong and memorable. Using syntactic and semantic features, we analyze password dataset and provide the insight to creating memorable and strong password. With 2-dim. password dataset, we constructed a 3rd-order tensor consisting of $(passwords \times features \times strength) = ((109, 2276, 875) \times 31 \times 3)$ and decomposed them using PARAFAC2. We discovered that password length is the dominant contributor in making a password stronger, compared to all other features. Among semantic features, most of them did not influence a strong password, except common nouns. Hence, we can conclude that creating a long password is the most powerful and practical way to strengthen memorable passwords. Although PARAFAC2 tensor decomposition provides a high degree of flexibility, it suffers from the uniqueness issue. Future research is needed in this area to further improve the performance of PARAFAC2. Also, we used more than 3,200 memorable passwords but more dataset would be helpful in more accurately identifying features. We also plan to perform our analysis with additional password datasets to verify our findings.

8 CONCLUSION

A tensor decomposition is the powerful method to quantitatively capture the importance and relationships among of underlying features in a dataset, providing the higher level of explainability. In this work, we applied the tensor decomposition for password data analysis and validate the features that strongly influence the creation of memorable and strong passwords. Our work complements the existing qualitative research and confirm the importance of creating a longer password using non dictionary words. Our work can be applied for better understanding the password structures for an unlabelled password dataset. Further, it can be integrated in generating password suggestions and advice for users to create memorable and strong passwords.

9 ACKNOWLEDGMENTS

We thank anonymous reviewers for their helpful feedback on drafts of this paper. This research was supported by the KEIT (Korea Institute for Industrial Economics & Trade), Korea, under the "Global Advanced Technology Center" (10053204), and by the MSIT (Ministry of Science and ICT), Korea, under the ICT Consilience Creative Program (IITP-2017-R0346-16-1007) supervised by the IITP (Institute for Information & Communications Technology Promotion), and by NRF of Korea by the MSIT (NRF-2017R1C1B5076474).

REFERENCES

[1] J. B. Kruskal. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and Its Applications* 18, 2 (J. B. Kruskal), 95–138. https://doi.org/10.1016/0024-3795(77)90069-6

[2] Evrim Acar, Seyit A Çamtepe, Mukkai S Krishnamoorthy, and Bülent Yener. 2005. Modeling and multiway analysis of chatroom tensors. In *International Conference on Intelligence and Security Informatics*. Springer, 256–268.

[3] Evrim Acar, Seyit A Camtepe, and Bülent Yener. 2006. Collective sampling and analysis of high order tensors for chatroom communications. In *International Conference on Intelligence and Security Informatics*. Springer, 213–224.

[4] Evrim Acar and Bülent Yener. 2009. Unsupervised multiway data analysis: A literature survey. *IEEE transactions on knowledge and data engineering* 21, 1 (2009), 6–20.

[5] Charlotte Møller Andersen and R Bro. 2003. Practical aspects of PARAFAC modeling of fluorescence excitation-emission data. *Journal of Chemometrics: A Journal of the Chemometrics Society* 17, 4 (2003), 200–215.

[6] Carl J Appellof and Ernest R Davidson. 1981. Strategies for analyzing data from video fluorometric monitoring of liquid chromatographic effluents. *Analytical Chemistry* 53, 13 (1981), 2053–2056.

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

[8] J Douglas Carroll and Jih-Jie Chang. 1970. Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika* 35, 3 (1970), 283–319.

[9] Claude Castelluccia, Markus Dürmuth, and Daniele Perito. 2012. Adaptive Password-Strength Meters from Markov Models.. In *NDSS*.

[10] Hsien-Cheng Chou, Hung-Chang Lee, Hwan-Jeu Yu, Fei-Pei Lai, Kuo-Hsuan Huang, Chih-Wen Hsueh, et al. 2013. Password cracking based on learned patterns from disclosed passwords. *IJICIC* 9, 2 (2013), 821–839.

[11] Andrzej Cichocki, Danilo Mandic, Lieven De Lathauwer, Guoxu Zhou, Qibin Zhao, Cesar Caiafa, and Huy Anh Phan. 2015. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Processing Magazine* 32, 2 (2015), 145–163.

[12] Lieven De Lathauwer and Joos Vandewalle. 2004. Dimensionality reduction in higher-order signal processing and rank-(R1, R2,..., RN) reduction in multilinear algebra. *Linear Algebra Appl.* 391 (2004), 31–55.

[13] Matteo Dell'Amico and Maurizio Filippone. 2015. Monte Carlo Strength Evaluation: Fast and Reliable Password Checking. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 158–169.

[14] Dinei Florêncio, Cormac Herley, and Paul C Van Oorschot. 2016. Pushing on string: The'don't care'region of password strength. *Commun. ACM* 59, 11 (2016), 66–74.

[15] Wolfgang Hackbusch. 2012. *Tensor spaces and numerical tensor calculus*. Vol. 42. Springer Science & Business Media.

[16] Richard A Harshman. 1970. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis. (1970).

[17] Frank L Hitchcock. 1927. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics* 6, 1-4 (1927), 164–189.

[18] Frank L Hitchcock. 1928. Multiple invariants and generalized rank of a p-way matrix or tensor. *Journal of Mathematics and Physics* 7, 1-4 (1928), 39–79.

[19] Markus Jakobsson and Mayank Dhiman. 2013. The benefits of understanding passwords. In *Mobile Authentication*. Springer, 5–24.

[20] Henk AL Kiers. 2000. Towards a standardized notation and terminology in multiway analysis. *Journal of Chemometrics: A Journal of the Chemometrics Society* 14, 3 (2000), 105–122.

[21] Tamara G Kolda and Brett W Bader. 2009. Tensor decompositions and applications. *SIAM review* 51, 3 (2009), 455–500.

[22] J. B. Kruskal. 1989. Multiway Data Analysis. North-Holland Publishing Co., Amsterdam, The Netherlands, The Netherlands, Chapter Rank, Decomposition, and Uniqueness for 3-way and N-way Arrays, 7–18. http://dl.acm.org/citation.cfm?id=120565.120567

[23] Arvind Narayanan and Vitaly Shmatikov. 2005. Fast dictionary attacks on passwords using time-space tradeoff. In *Proceedings of the 12th ACM conference on Computer and communications security*. ACM, 364–372.

[24] Evangelos E Papalexakis, Christos Faloutsos, and Nicholas D Sidiropoulos. 2017. Tensors for data mining and data fusion: Models, applications, and scalable algorithms. *ACM Transactions on Intelligent Systems and Technology (TIST)* 8, 2 (2017), 16.

[25] Ashwini Rao, Birendra Jha, and Gananand Kini. 2013. Effect of grammar on security of long passwords. In *Proceedings of the third ACM conference on Data and application security and privacy*. ACM, 317–324.

[26] Paul Rayson, Dawn Archer, Scott Piao, and Anthony M McEnery. 2004. The UCREL semantic analysis system. (2004).

[27] Ledyard R Tucker. 1963. Implications of factor analysis of three-way matrices for measurement of change. *Problems in measuring change* 15 (1963), 122–137.

[28] Ledyard R Tucker. 1964. The extension of factor analysis to three-dimensional matrices. *Contributions to mathematical psychology* 110119 (1964).

[29] Ledyard R Tucker. 1966. Some mathematical notes on three-mode factor analysis. *Psychometrika* 31, 3 (1966), 279–311.

[30] Blase Ur, Fumiko Noma, Jonathan Bees, Sean M Segreti, Richard Shay, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. 2015. "I added '!' at the end to make it secure": Observing password creation in the lab. In *Proc. SOUPS*.

[31] M Alex O Vasilescu and Demetri Terzopoulos. 2002. Multilinear analysis of image ensembles: Tensorfaces. In *European Conference on Computer Vision*. Springer, 447–460.

[32] M Alex O Vasilescu and Demetri Terzopoulos. 2002. Multilinear image analysis for facial recognition. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, Vol. 2. IEEE, 511–514.

[33] Rafael Veras, Christopher Collins, and Julie Thorpe. 2014. On the semantic patterns of passwords and their security impact. In *Network and Distributed System Security Symposium (NDSS'14)*.

[34] Matt Weir, Sudhir Aggarwal, Breno De Medeiros, and Bill Glodek. 2009. Password cracking using probabilistic context-free grammars. In *Security and Privacy, 2009 30th IEEE Symposium on*. IEEE, 391–405.

[35] Simon Woo, Elsi Kaiser, Ron Artstein, and Jelena Mirkovic. 2016. Life-experience passwords (leps). In *Proceedings of the 32nd Annual Conference on Computer Security Applications*. ACM, 113–126.

[36] Simon S Woo and Jelena Mirkovic. 2018. GuidedPass: Helping Users to Create Strong and Memorable Passwords. In *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 250–270.