The Past and Possible Future Development of Password Guessing

Ze-Long Li
University Of Jinan
UJN
Shandong Province, China
Email: 202121200928@stu.ujn.edu.cn

Teng Liu
University Of Jinan
UJN
Shandong Province, China
Email: 202121200914@stu.ujn.edu.cn

Lei Li
Jinan Blue Sword Junxin Information Technology Co., Ltd
Shandong Province, China
Email: leilimoon@hotmail.com

***Abstract*—According to the China Internet Network Information Center (CNNIC), by December 2022, the number of Internet users in China has reached 1.067 billion. Recently, consulting firm Kepios pointed out that nearly 5 billion people worldwide are currently active on social networks. Nowadays although there are many methods of identity authentication: fingerprint recognition, facial recognition and static password, static password is still the most widely used identity authentication method. Most people usually set passwords too simple and easily cracked. This allows attackers to crack their passwords with less cost. Password guessing technology can generate large-scale password dictionaries, which can be used to evaluate the strength of passwords and encourage users to change their own passwords. With the development of deep learning, password guessing technology is also constantly breaking through. But few people provide systematic surveys, which allow us to systematically review the most advanced methods and avoid repetitive research. Firstly, we will conduct a comprehensive analysis of the development of password guessing technology to this day. Secondly, we will propose future feasibility research methods based on the latest technology to address the shortcomings of password guessing models.**

Keywords-deep learning; generative models; neural networks; normalization methods; network and information security.

I. INTRODUCTION

In the real world, there are three basic methods for authenticating users: 1) proving your identity based on what you know; 2) proving your identity based on what you have; 3) directly proving your identity based on unique physical characteristics. Currently, identity authentication based on passwords, especially static passwords, is still one of the most widely used authentication methods. The password set by users is always related to personal information, as it is easy to remember and can be set repeatedly for different accounts [30]. For example, according to Dojo's 2023 cracked password list, many people prefer to use pet names, lover birthday, and other information as passwords. Attackers can directly access this information through social media and public personal information, thereby stealing passwords. Therefore, although password settings are simple, their security is still an important issue.

Password security issues have long been a concern, and various websites have adopted different password setting rules to prevent users from using weak passwords. Forcing users to follow password rules has little impact on improving password strength, as evidenced by the three points (in the first paragraph) mentioned earlier. Therefore, research on password guessing attacks is necessary. Our goal is not to crack user passwords, but rather to provide password strength detection, allowing users to understand the strength of their passwords and prompting them to modify them. Password guessing can be divided into offline password guessing and online password guessing [38]: offline mode requires stealing password files in advance, conducting unrestricted attack attempts, and does not require cracking speed; Online mode must use the same login portal as the user, with a limit on the number of times. This article conducts a systematic investigation of password guessing technology and provides a detailed explanation of most models.

The main contributions of this article are as follows:

- A systematic review was conducted on the password guessing methods mentioned in the references, with some models providing method details.
- Introduce improvement methods based on the original model by class, and each method improves the original method.
- Discuss the limitations of password guessing and propose feasible future research directions based on new technologies.
- Mention three methods for optimizing password guessing.

The rest of this paper is organized as follows. Section II is the background and related work. Section III describes the models. Section IV proposes several future research directions. The conclusion closes the article.

II. BACKGROUND AND RELATED WORK

As early as 1979, Robert Morris and Ken Thompson mentioned two attacks that are very familiar in the field of information security: violent cracking (a method of cracking passwords by calculating them one by one until the true password is found.) and Dictionary attack in their paper on UNIX password security [6]. The disadvantage of the former is that it is very time-consuming, while the latter requires a large amount of memory. According to the shortcomings of the two methods, in 1980, Hellman proposed a time-memory trade-off (TMTO) method, which allows people to balance

 ISBN: 978-1-68558-104-6

time and memory costs [12]. In 2003, in order to reduce the number of calculations in the cryptanalysis process, Oechslin proposed a precomputation method - Rainbow table [16]. The Rainbow table is an improvement on the TMTO method.

The above is the most original method for password guessing. In 2005, Narayanan and Shmatikov proposed to apply the Markov model to password guessing [1], which is better than the Rainbow table method. Since then, password guessing has entered a "new era". The Markov model is a statistical model, and its most widespread application is speech recognition. Since 2005, with the continuous development of artificial intelligence, more and more experts have begun to pay attention to how Markov models can be optimized for better password guessing [3][14][18][19][47].

Probabilistic Context Free Grammar (PCFG) was originally used for syntactic analysis and is another traditional password guessing method after the Markov model. It was proposed by Weir et al. [4] in 2009. PCFG checks grammar structures (combinations of special characters, numbers, and alphanumeric sequences) and generates distribution probabilities, which are then used to generate candidate passwords.

When using PCFG, we need to consider the password structure, that is, the password setting rules. Therefore, we need to understand people's setting habits and website requirements, which are aimed at domestic and foreign users [31][32][44]. Most of the literature is about English-speaking users, and only a few studies have examined how non-English users choose passwords. In 2019, Wang et al. [32] conducted a comparative analysis of 73.1 million domestic passwords and 33.2 million English websites in real life, emphasizing the structural and semantic features of domestic password settings. Compared with foreign users, the passwords set by domestic users are less resistant to online guessing attacks, but better resistant to offline guessing attacks. Wang et al. [32] systematically discussed several basic attributes of passwords, such as the relationship between passwords and language, and found that there are great differences in letter distribution, structure and semantic patterns between domestic and foreign. PCFG and Markov models are used to attack, with the main purpose of enabling users to protect personal accounts more deeply. In the same year, Kaevrestad et al. [44] conducted research on the classification of password creation strategies. The main purpose was to better understand the password setting rules and better understand passwords. According to the survey summary provided by 21 experts, the password categories are divided into 7 categories: phrases; biographical passwords; leetspeak; dates; words; combination of words and numbers; random passwords. More specifically, it can be divided into four categories: only numbers; alpha numeric characters (numbers, small and large letters); special characters.

After 2011, the field of artificial intelligence has entered a booming period. Recursive Neural Network (RNN) model has been widely used in the field of Natural language processing. It can model based on time series data to process data, such as text prediction. In 2016, considering the inaccuracy of modeling password guessing at the time, Melicher's team [5] proposed using neural networks to simulate the resistance of text passwords to guessing attacks. This is another major progress after applying the Markov model to the field of password guessing in 2005. Traditional RNN may cause gradient explosion, so Melicher's team [5] chose to use Long Short Memory Network (LSTM) to solve the gradient explosion problem. Two years later, Zhang et al. [8] proposed a password cracking method based on structural partitioning and BiLSTM recurrent neural network. It is also the use of neural networks. In 2022, Ye et al. [11] applied time domain Convolutional neural network (TCN) to password guessing and added tag learning method. After a year, in 2023, Wu et al. [15] once again used TCN for password guessing and named it PGTCN, which can automatically study the structure and characteristics of passwords and generate new passwords based on the knowledge learned.

In 2014, Ian Goodfellow et al. [37] proposed Generative Adversarial Networks (GANs), which have powerful functions and have been studied and applied since their introduction [40][41][42][43]. Considering that the GANs model is a Generative model, it can be used for Natural Language processing. Therefore, in 2019, Hitaj et al. [9] applied the GANs model to password guessing, and its performance was superior. The GANs model has obvious drawbacks: it is difficult to train due to unstable training, vanishing gradients and pattern collapse when processing text data. Although the Hitaj team used wasserstein distance to slightly improve, its shortcomings are still evident.

III. MODEL EXPLANATION

This section provides a detailed explanation of password guessing models related to Markov, PCFG, and deep learning.

A. *Markov model family*

Markov chain can be traced back to 1906-1912, which was proposed by Markov and is an important concept in machine learning. Markov chain is a Stochastic process in the state space through the transition from one state to another. The probability distribution of the next state can only be determined by the current state, and the events before it in the time series are independent of it. This specific type of "memoryless" is called Markov property.

The following are the basic elements of a Markov chain:

1) State space: Let $X_n=i$ indicate that the state at time n is i, and the set of values of all states is called the "state space".

2)Transition probability: the Conditional probability from the state at the current time to a state at the next time is called "transition probability".

$$P_{ij}=P(X_n=j|X_{n-1}=i) \quad (1)$$

The above equation represents the probability of transitioning from state i to state j.

3) State-transition matrix: there may be more than one state at each time, so the transition probability between all states is formed into a matrix, which is called "State-transition matrix", and the size of the matrix is set to |I| * |I|. It should be noted that this matrix does not change over time.

4) Initial state: p_0.

In the Markov hypothesis, we need to use the N-Gram algorithm, which assumes that the occurrence of the nth word is only related to the first N-1 word and not to any other word. The probability of the entire sentence is the product of the probabilities of each word's occurrence. These probabilities can be obtained by directly counting the number of times N words appear simultaneously from the corpus.

In the zero order Markov model, the generation of the current character is independent of the previously generated character. In the first-order Markov model, each diagram (ordered pair) of characters is assigned a probability, and the current character is generated by looking at the previous character. Mathematically, in the zero-order model [1]:

$$P(\alpha)=\pi_{x\in\alpha}v(x) \quad (2)$$

In the first-order model:

$$P(x_1x_2...x_n)=v(x_1)\pi_{i=1}^{n-1}v(x_{i+1}|x_i) \quad (3)$$

The reason why Markov model can be used for password guessing is that a Markov model defines a probability distribution on a symbol sequence. In other words, it allows for sampling of character sequences with certain attributes.

The drawbacks of the Markov model are also evident, as it generates a large amount of duplicate data when cracking passwords, resulting in high repetition rates and low coverage, resulting in resource waste (as shown in Table I). A new method based on Markov model has been proposed. In 2015, Dürmuth et al. [3] proposed a method using an ordered Markov enumerator (OMEN) based on the idea proposed by Narayanan, considering orderliness. Simply put, they generate candidate passwords based on the probability of their occurrence, and the first output is the one with the highest probability. OMEN has improved the speed of password guessing, and it is worth noting that it only approximates the likelihood of passwords.

The main parameters of OMEN include n-gram size, alphabet size, and Number of levels.

The main algorithm enumPwd (): At a high level, enumPwd () will discretize all probabilities into multiple bins, iterate each bin in descending order of probability, and output the password that matches the probability of the bin in each bin. For specific password lengths ℓ and level η. EnumPwd (η, ℓ) executes as follows:

Firstly, we need to calculate a vector **a**= (a_3,..., a_l) with a length of ℓ-1. Each a_i represents an integer within [0, nbLevel-1], and the sum of all elements is η. Because there are ℓ-1 elements, when using 3-grams, it is necessary to have ℓ-2 transition probabilities and 1 initial probability to determine the probability of a string of length ℓ. For example, the probability of a password with a length ℓ=7 is calculated as follows:

$$P(loveyou)=P(lo)P(v|lo)P(e|ov)P(y|ve)P(o|ey)P(u|yo) \quad (4)$$

For each such vector a, select 2-grams x_1x_2(all) and iterate through all x_3, with the aim of 3-grams $x_1x_2x_3$ to obtain the level value a_3. Next, iterate x_4 for each 3-gram to obtain the level value a_4, with the aim of 3-gram $x_2x_3x_4$ Continue this process until the expected length is reached, and the final output is a set of candidate passwords (length of ℓ, level of η).

When setting parameters ℓ and η, the setting of ℓ is quite difficult, as the problem arises from the password length during training and people's guesses about a specific length. Therefore, Dürmuth et al. [3] added an adaptive algorithm to track the success rate of different password lengths.

Although this method effectively improves the speed of guessing and the coverage rate, its results will not change no matter how many training parameters are determined, and it always generates the same password in the same order, which is a deterministic algorithm. Fully considering the advantages and disadvantages of ordinary Markov and OMEN, Guo et al. [18] proposed a dynamic mechanism called the dynamic Markov model in 2021. Compared with ordinary Markov and OMEN, this model reduced the repetition rate from 75.88% to 66.50% and increased the coverage rate from 37.65% to 43.49%.

The purpose of the dynamic mechanism is to reduce the repetition rate and to improve coverage.

1)This method is only suitable for random sampling.

2)Every time a password is generated, a dynamic mechanism is used to reduce the probability of its subsequent occurrence.

3)For any string C: m≤m_{MAX}, set the string space to S_S . Form a set of strings with probability values greater than 0 and define this set as a support set S_M , which is a subset of S_S.

4)For strings outside of S_M, we believe they have no cracking significance.

Dynamic mechanism principle: For any original distribution, we set it to $D_{original}$ and represent the number of passwords in $S_M$with N. We randomly select a password P from S_M, which needs to meet the following requirements:

$$p_i^{original}\times N\geq 1 \quad (5)$$

By a small parameter α reduces its probability, based on α/ (N-1) increasing the probability of other passwords, and the probability distribution D_{new} of the new password is closer to a uniform distribution D_{uni}.

Although the S_M size is assumed to be N, N is unknown. In practical operations, we cannot directly handle the entire password probability. The authors provide a simplified method to replace it, which is to only consider n-gram fragment. Figure 1 shows the process of dynamic Markov generating passwords.

There is also an application method of the Markov model, which combines the GAN model and will be explained in

TABLE I. THE NUMBER AND RATE OF REPETITIONS IN PASSWORD GENERATION BY MARKOV MODELS

password generation	10^6	10^7	10^8
Number of duplicate passwords	4.79×10^5	5.94×10^6	6.86×10^7
Repetition rate	47.93%	59.37%	68.61%

section D. Table II compares three Markov models for the coverage number of different probability ciphers.

B. Probabilistic Context Free Grammar family

Note that rule processing in dictionary-based password guessing is a difficult task. Therefore, Weir et al. [4] proposed a method based on PCFG to generate password structures in the highest probability order, which fundamentally considers the structure of passwords, i.e., the rules for password settings.

PCFG is an extension of Context Free Grammar (CFG), which is a method of Rule Based Natural Language Processing (NLP). The main function of CFG is to verify whether the input string conforms to a certain grammar G, which is similar to regular expressions, but CFG can express more complex grammars. CFG is a set of replacement rules, for example: 0->O indicates that variable 0 can be replaced by variable O.

PCFG only adds the probability associated with each generation, and all associated productions add up to 1. Weir et al. [4] used only L_n, D_n *and* S_n (L represents letters, D represents numbers, and S represents special characters.) for the specified n-value in grammar, except for the starting symbol. They call these variables alpha variables, digit variables and special variables respectively. Table III is an

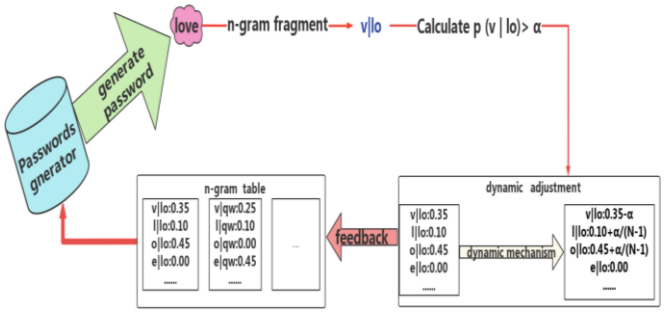

Figure 1. Dynamic Markov model generating password process.

TABLE II. THE COVERAGE OF THREE MARKOV MODELS

Probability	Total	Markov Model	OMEN	Dynamic Markov Model
$<10^{-12}$	310024	161	817	68
$[10^{-12},10^{-11})$	231826	1168	26988	980
$[10^{-11},10^{-10})$	334797	13507	1409272	13275
$[10^{-10},10^{-9})$	410065	95250	287433	122463
$[10^{-9},10^{-8})$	390731	272115	340801	350988
$[10^{-8},10^{-7})$	284911	271945	268945	284897
$[10^{-7},10^{-6})$	116095	115973	112954	116094
$[10^{-6},10^{-5})$	19827	19826	19456	19827
$[10^{-5},10^{-4})$	2701	2701	2671	2701
$[10^{-4},10^{-3})$	243	243	243	243
$[10^{-3},10^{-2})$	4	4	4	4

example of PCFG, based on which the pre-terminal structure can be further derived:

$$S \rightarrow L3D1S1 \rightarrow L34S1 \rightarrow L34! \quad (6)$$

Keyboard order (keyboard mode) and multi word strategy can greatly improve the PCFG based password guessing method proposed by Weir et al. [4], which is an important supplement to PCFG [45]. This method was proposed by Houshmand et al. [45] in 2015. By learning the new model, it can achieve 22% improvement over PCFG, and the authors also defined metrics to help analyze and improve the dictionary. Increase the coverage of standard attack dictionary, achieving an additional increase of~33%. The keyboard mode does not consider the characters actually entered but is a shape that is easy to remember.

For example, “asdfg” is a series of keys from “a” to the next four letters on the right. Therefore, the keyboard mode is considered as a series of keys on the keyboard, and passwords can be created according to various combinations. The keyboard mode uses the symbol K as a new nonterminal character to be introduced into PCFG. Table IV is a comparison between Weir et al.'s [4] method and the other author's method. There are two considerations on how to determine whether it is keyboard mode or original (L, D, S) structure:

1)Pure numbers or special characters are classified as original structures and as components of D/S.

2)When it does not belong to the first item and the substring contains at least 3 characters, the keyboard mode requires the maximum length. For example, “asdfghui12” is classified as keyboard mode K8D2, not L6D4/K6D4.

Assuming a set of words is W $\{w_1...w_n\}$, considering it as a dictionary, and R is the cipher set $\{p_1...p_m\}$. If w is an L-structure in R, the password in R must have at least one w. If w is found in R, I (w, R) =1, otherwise I (w, R) =0. The accuracy definition of W for R is as follows:

TABLE III. PCFG EXAMPLE

Left-Hand Side	Right-Hand Side	Probability
S→	$D_1L_3S_2D_1$	0.75
S→	$L_3D_1S_1$	0.25
D_1→	5	0.60
D_1→	2	0.20
D_1→	1	0.20
S_1→	!	0.65
S_1→	%	0.30
S_1→	*	0.05
S_2→	&&	0.70
S_2→	$$	0.30

 ISBN: 978-1-68558-104-6

$$P(W,R)=\frac{1}{|W|}\sum_{i=1}^{n} I(w_i, R) \quad (7)$$

Assuming a password has k different L-structures, letting the count be c (w, p), where p is the number of L-structures, and the value is w. The coverage of word w is:

$$C(w, p)=\frac{c(w, p)}{k} \& C(w, R)=\sum_{i=1}^{m} C(w, p_i) \quad (8)$$

Set to a subset of passwords with at least one L-structure in R. The coverage of dictionaries W and R is as follows:

$$C(W,R)=\frac{1}{|R_L|}\sum_{i=1}^{n} C(w_i, R) \quad (9)$$

Through its development, PCFG not only allows for guessing passwords in probabilistic order, but also fully considers keyboard mode, resulting in higher cracking coverage. However, in practical applications, low probability passwords are still difficult to crack because they often lack semantic structure. Although lacking semantic structure, low probability ciphers also have a large search space and certain semantic information [13]. Therefore, considering the importance of improving the low probability password hit rate for offline attack efficiency, Guo et al. [13] proposed a degenerate distribution collection method in 2022 and designed a corresponding Low Probability Generator Probabilistic Context Free Grammar (LPG-PCFG) model based on PCFG. Compared with PCFG, LPG-PCFG aims to increase the distribution of low probability passwords, and when generating 10^7 and 10^8 passwords respectively, the number of hits increases by 50.4% and 42.0%.

Assuming the degenerate distribution as D_{deg} is the intermediate state between modeling and D_{uni}. The closer the degenerate distribution is to a uniform distribution, the better the distribution for generating low probability ciphers. However, passwords generated very close to each other may lack learning features, and measuring quality and low probability passwords is a challenge. There will be an optimal degenerate distribution D^{*}_{deg}, which can achieve a balance between modeling and uniform distribution, thus effectively generating low probability ciphers. Table V shows several probability correction rules aimed at obtaining a degenerate distribution, mainly through the following mechanisms:

Sampling to obtain password x^+, by sampling the generated model, modification probability:

TABLE IV. KEYBOARD BASE STRUCTURES VS PCFG

Passwords	PCFG	Keyboard
1234	D_4	K_4
w2w2	LDLD	K_4
ASD1234QW	$L_3D_4L_2$	$K_3D_4L_2$
Q1!2	LDSD	K_4

$$p(x^+) - \alpha \quad (10)$$

Support the probability of other passwords (x^-) in the set, where N_S represents the number of passwords supported in the set:

$$p(x^-)+\alpha(N_S-1) \quad (11)$$

Guo et al.'s [13] method enables low probability ciphers to also have good guessing performance, greatly improving the hit rate. The article mentions the semantic structure and low probability password semantic information but has not conducted in-depth research on them. It still uses passwords created by English or Chinese users for research. In June 2023, due to insufficient investigation of cryptographic semantic information, Wang et al. [46] proposed a general framework for PCFG based on semantic enhancement, named SE # PCFG. 43 types of semantic information are allowed to be considered for password analysis, which is by far the most abundant set. In addition, a Semantically Enhanced Password Cracking Architecture (SEPCA) was proposed by combining SE # PCFG with a smoothing method.

For better semantic analysis of passwords, the authors define four levels of password structures:

1)Character: The lowest level information about a password.

2)Semantic factor (SF): Some consecutive characters together form a semantic unit, which can be a word and carry semantic information called semantic factor type (SFT).

3)Semantic Pattern (SP): Consisting of one or more semantic factor types semantically, considering the entire password.

4)Semantic Structure (SS): Reflects the collective behavior of users, mapping shared and semantic attributes (language, website type).

Three step calculation process: preprocessing, identifying SFTs fragments, and post-processing.

The authors define a general PCFG as:

$$G=(M,T,R,S,P) \quad (12)$$

M and T represent non-terminal and terminal symbols. S is the beginning. R is the set of rules, and P is the probability contained in each rule R.

TABLE V. MODIFICATION RULES OF DEGENERATION DISTRIBUTION

Rule	Adjust $p(x^+)$	Adjust $p(x^-)$
Rule1	$p(x^+) - \alpha$	$p(x^-) + \alpha/(N_s - 1)$
Rule2	$p(x^+) - \alpha$	$p(x^-) + \alpha/(1 - p(x^+))p(x^-)$
Rule3	$\beta p(x^+)$	$p(x^-) + (1 - \beta)p(x^+)(1 - p(x^+))p(x^-)$
Rule4	$\beta p(x^+)$	$p(x^-)(1 - \beta)p(x^+)(1 - p(x^+))p(x^-)$
Rule5	$1-\gamma(1 - p(x^+))$	$\gamma p(x^-)$

In SEPCA, T is the set of all semantic factors, and M is the union of T and S. The rules are divided into two groups: from S to a certain SP; From one SFT to a certain SF.

At this point, the traditional password guessing methods, namely the Markov and PCFG models, as well as the improvements based on the two models, have been explained. Starting from section C, the application of neural networks in the field of password guessing is discussed. Table VI in section E summarizes the password guessing model.

C. Neural Network model family

Using Neural Networks to simulate the resistance of passwords to guessing attacks can be more effective than Markov models and PCFG. Neural network modeling uses less space than Markov models, and neural networks can transfer knowledge from a task to related tasks. Elements used in neural network models [5]:

1)Model architecture: Using recursive neural networks, character level text can be generated.

2)Alphabet size.

3)Password Context: Predicts characters related to the context (similar to Markov models).

4)Model size: Implementing with LSTM requires determining how many parameters are present in the model.

5)Transfer learning: Train a model about all passwords but adjust and guess longer passwords.

6)Data during training.

Melicher et al. [5] used an LSTM network, which could solve the gradient explosion problem caused by long text, including memory gates, forgetting gates, and output gates. Determine which information is discarded and which information is left through three "gates". The article proved that neural networks could guess passwords faster and more accurately. Since Melicher et al. [5] applied Neural Networks to password guessing, various Neural Network models, including various variants of LSTM, have been used to improve the hit rate and efficiency of password guessing [8][11][15][33].

In 2018, Zhang et al. [8] combined the advantages of PCFG and neural networks to propose a password guessing method based on structural partitioning and Bidirectional Long Short-Term Memory Recursive Neural Networks, named the SPRNN model. Firstly, divide the password into substructures, and then use the BiLSTM model to generate substrings based on the substructures, considering the accuracy and generalization ability of the model. The article points out that the SPRNN model performs well across datasets, with a hit rate of 25% to 30% higher than the general Markov model and 10% higher than the Weir et al. [4] method. In 2019, Li et al. [49] also applied BiLSTM to password guessing. In 2022, Chang et al. [33] addressed the difficulty of selecting sequence length in traditional LSTM models for password generation, and it is unclear whether there is a relationship between sequences of different lengths. Chang et al. [33] considered user personal information and proposed a multi sequence length LSTM password guessing model. Compared with traditional PCFG models, the hit rate has increased by 68.2%, and there is also an improvement of 7.6%~42.1% compared to traditional LSTM models. MLSTM consists of two stages: training stage and generation stage.

The method proposed by Chang et al. [33] addresses the length limitation of LSTM sequences, but the datasets used are all analyses of Chinese ciphers, and other datasets should also be considered in order to be more representative.

Ye et al. [11] proposed a password guessing model based on Time Convolutional Neural Network (TCN) (PassTCN).

Figure 2 introduces some Recursive Neural Networks and Convolutional Neural Networks.

TCN is an algorithm used to solve time series prediction. In order to further improve the performance of password generation, a new password probability label learning method is also proposed. Figure 3 shows the password guessing structure based on the TCN model.

The password probability label learning proposed by Ye et al. [11] is based on the probability distribution of the password and constructs a unique password label based on the probability distribution in the training set. Firstly, it is necessary to calculate the probability of different characters with known password prefixes in the training set, and construct password labels based on the probability values. Assuming it is any possible password prefix, set the ground truth label of the next character to y_i and thus obtain the probability of the next arbitrary character c:

$$P(c|prefix)=\frac{Count(prefix+c)}{Count(prefix)} \quad (13)$$

The PassTCN-PPLL method proposed by Ye et al. [11] effectively improves password coverage by combining time convolutional neural networks and password probability distribution labels. In 2023, Wu et al. [15] also proposed a password guessing model PGTCN improved by feature

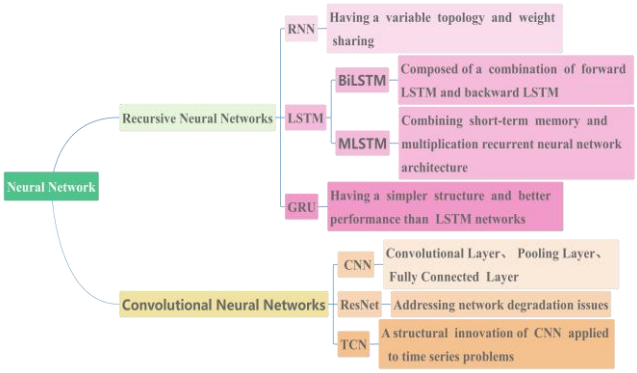

Figure 2. Partial Recursive Neural Networks and Convolutional Neural Networks.

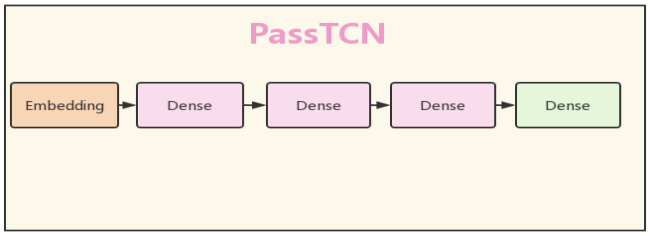

Figure 3. Structure Based on TCN.

fusion technology based on TCN combined with residual learning, with the aim of improving model performance and thus improving guessing efficiency. Relatively speaking, PGTCN is more stable. In order to reduce the probability of repetition, PGTCN also adopts a label like approach. Wu et al. [15] randomly introduce some labels when sampling the next label, rather than always selecting the most likely one. It is worth noting that PGTCN can thoroughly extract high level structures and low-level characteristics, which helps in password generation.

Neural networks are widely used in password guessing, with the main network models being LSTM and TCN. This only introduces the past two years and the original methods.

D. GAN model family

In 2014, Goodfellow et al. [37] proposed Generative Adversarial Nets (GANs). GANs are inspired by the zero-sum game theory, which consists of two parts: a generative model (G) and a discriminant model (D). The generative model captures the data distribution of samples, which is used in password guessing to generate passwords that can deceive the discriminant model. The discriminant model is actually a binary classifier used to distinguish whether the input data is true or false, whether it is real data or generated samples by the generative model. The emergence of GAN has enabled networks to learn more precise losses through adversarial learning, prompting generators to generate higher quality results, greatly promoting the development of this field and entering the vision of more popular.

The optimization objective function of GAN is as follows:

$$\min_{G} \max_{D} V(D,G) \tag{14}$$

$$V(D,G)=E_{x\sim P_{data}(x)}[\log D(x)]+E_{z\sim P_{z}(z)}[\log(1-D(G(z)))] \tag{15}$$

Equation (15) represents the loss function of GAN. Train network G to minimize log(1-D(G(z))), i.e., to maximize the loss of D. Training network D to maximize log D(x) and log(1-D(G(z))). In the G network, log(1-D(G(z))) represents loss. Under the D network, - log D(x) + log(1-D(G(z))) represents loss.

In 2019, Hitaj et al. [9] proposed applying GAN to password guessing and named it PassGAN. PassGAN does not rely on password analysis like Markov models, PCFG, and neural networks, but instead uses GAN to automatically learn the true password distribution from publicly leaked passwords. In other words, we do not need any professional knowledge related to cryptography, and applying GAN can generate high-quality passwords for guessing. Hitaj et al. [9] used Improved training of Wasserstein GANs (IWGAN) [42][43], with the optimizer using ADAM, which IWGAN relies on to minimize training errors.

The PassGAN generator G structure consists of 5 residual blocks, a one-dimensional convolutional layer, and activation functions using Linear and SoftMax; The discriminator D structure includes 5 residual blocks, a one-dimensional convolutional layer, and an activation function using Linear. The positions of G and D convolutional layers and linear activation functions are different. Figure 4 shows the residual module structure, and Figure 5 shows the PassGAN structure.

GAN does not require complex Markov chains to perform well in password guessing, but it has problems with unstable training, vanishing gradients, and mode collapse. Although Hitaj et al. [9] applied IWGAN, the problem still exists. Nam et al. [27] proposed a candidate password for optimizing guessing, named REDPACK using a relativistic GAN method. REDPACK effectively combines multiple generation models to generate passwords. Generator G can effectively optimize candidate password selection by selecting different models, such as OMEN, PCFG, etc. Nam et al. [27] improved the performance of cracking through custom rules, and there is still room for further improvement in the future.

In 2022, Jiang et al. [14] and Yu et al. [10] proposed a password generation model based on ordered Markov enumeration and discriminant networks (OMECDN) for PassGAN and added gradient normalization to PassGAN [10][21][22]. Jiang et al. [14] changed generator G to OMEN and discriminator D to critical discriminant network. OMECDN can sort based on the probability of password combinations, match the true password distribution, and reduce repetition rates. Yu et al. [10] found that the combination of IWGAN and gradient penalty is not an ideal method to solve the shortcomings of GAN, so they added gradient normalization counting to discriminator D and named it GNPassGAN. GNPassGAN guessed 88.03% more passwords than PassGAN, reducing repetition by 31.69%. Zhou et al. [17] proposed a new structure based on PassGAN, which uses LSTM network for generator G and multiple convolutional layers in discriminator D, based on the non-differentiability of discrete data sampling process and the impact on backpropagation. In addition, the biggest contribution is the addition of Gumbel SoftMax, named G-Pass. Dynamically adjust parameters during the training process, balancing sample diversity and sample quality.

Gumbel SoftMax assumes that the vocabulary size is v and $h \in R^{v}$ is the output of the last layer of the generator; P specifies the distribution of categories, Y

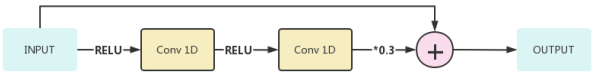

Figure 4. Residual Block's Architecture.

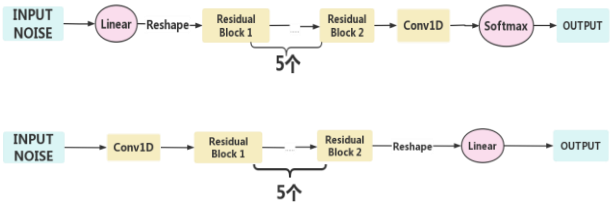

Figure 5. PassGAN's Architecture.Upper generator G, lower discriminator D.

follows a distribution $P(p_1, p_2, \ldots, p_v)$, and p_i is the probability calculated by SoftMax; Using the reparameterization technique to reconstruct random sampling into a deterministic (h_i) and a random element combination(g_i), let F (x) represent a random distribution; We can obtain the inverse function of F (x), from which we can calculate g_i when u~U [0,1]; At this point, one_Hot (argmax (.))is still non-differentiable, using SoftMax as an approximation.

E. Others

This section mainly summarizes the applications of other neural networks and deep learning models proposed in the field of password guessing in recent 2018 and beyond. Table VI at the end of this section displays all the models mentioned in this article.

The various models described in sections A to D rarely consider cross site nature, and the characteristics of different datasets are different because different websites require different password setting rules and target different user groups. In 2018, Liu et al. [7] proposed a universal password guessing model GENPass for cross site nature. Its generator is PCFG+LSTM, and under10^{12}guessing times, the cross-site performance of this model is 20% higher than that of a simple mixed dataset hit rate. In the final analysis, this model is also an application of adversarial thinking, and Xia et al. [48] proposed a similar idea in 2020. In 2022, He et al. [28] proposed a password reuse model called PassTrans based on transformer. The attention mechanism of transformer can be calculated according to the following equation:

$$\text{Attention}(Q,K,V)=\text{softmax}\left(\frac{QK^T}{\sqrt{4iQ_k}}\right) \qquad (16)$$

Q, K and V represent query, key, and value, respectively. Q. K and V calculate the similarity between the current query and all keys and obtain a set of weights by passing this similarity value through the Softmax layer. Q, K and V are both weight matrices.

Sanjay et al. [29] proposed a password generation technique based on a bidirectional generative adversarial network algorithm (BiGAN) using classification and guessing strategy methods, with the aim of generating passwords that improve convergence speed, named PassMon. BiGAN structure: generator, encoder, and discriminator. Pagotta et al. [34] proposed a stream-based generation model for password guessing. The stream-based method was first proposed, providing a representation of the latent space, making it possible to explore specific subspaces and interpolation operations of the latent space, named PassFlow. The PassFlow application has a smaller training set and performs better than PassGAN. The generated password quality is good, and even if it does not match, its rules are very similar to people's password habits. The PassFlow training set is small, in other words, even a subset of the training set, PassFlow, can be effectively used, so it is less affected by the limited number of datasets due to domain specificity. In 2023, Rando et al. [36] proposed a password guessing method based on large language models (LLMs) that can successfully model natural language from a large amount of text without explicit supervision, named PassGPT.

TABLE VI. VARIOUS PASSWORD GUESSING MODELS

Model Name	Basic Generation Model Types	Publication Year
Markov	Markov	2005
PCFG	PCFG	2009
OMEN	Markov	2015
Next Gen PCFG	PCFG	2015
FLA	RNN,LSTM	2016
PassGAN	GAN,IWGAN	2017,2019
GENPass	PCFG,LSTM	2018,2020
SPRNN	BiLSTM	2018
BiLSTM	BiLSTM	2019
REDPACK	PCFG,GAN,etc.	2020
Dynamic Markov	Dynamic Markov	2021
GNPassGAN	GAN	2022
PassTCN-PPLL	TCN	2022
LPG-PCFG	PCFG	2022
G-Pass	GAN	2022
Passtrans	Transformer	2022
OMECDN	Markov,GAN	2022
PassMon	BiGAN	2022
MLSTM	MLSTM	2022
PassFlow	Flow	2021,2022
WordMarkov	Markov	2022
SE#PCFG	PCFG	2023
PassGPT	GPT-2	2023
PassTCN	TCN	2023

IV. DISCUSSION AND FUTURE RESEARCH

Table VI shows that password guessing technology has developed rapidly in the past two years. Although various models have performed well, there are also various defects and deficiencies. Based on the newly proposed optimization methods and model structures in recent years, this chapter proposes several future research directions in the field of password guessing to address limitations and unresolved work:

- Public password data is not easy to find. We can consider data augmentation technology (DA) to obtain more data. Note that the application of DA technology should not aimlessly expand the data, as

obtaining poor data can lead to worse results. We need to clean the obtained data and eliminate bad data. According to research, some users will set their passwords based on the topic of the website [24]. Password guessing often requires a dictionary, and we can use the DA method to obtain as many websites with the same topic as possible. Based on the special password generation strategy of website themes, a dictionary is generated for password guessing.

- Spectral Normalization (SN) can improve the stability of discriminator D in GAN [35], which is also a variant of GAN. The work we are doing not only applies SN to discriminator D, but also adds SN to generator G. Multiple variants of GAN for password guessing may achieve better results. Of course, model training requires the use of optimizers [20][21][22].
- There are already models in the literature other than Markov models, PCFG, GAN, etc. applied to password guessing. We hope that more types of neural networks and deep learning models can be applied to password guessing [2][23][25][26].
- Password rules cannot be ignored, as most literature does not consider password rules, and the rules required by different websites may vary. Considering the combination of password setting rules and the topic dictionary mentioned above, we hope to apply them simultaneously to password guessing.
- We need to detect password leaks, and Honeywords is a type of bait password used to provide feedback on password leaks [39]. As an effective method for detecting whether passwords have been cracked, how to generate Honeywords better has become a research direction. We can consider using the basic models of various password guessing models mentioned in Table 6, such as PCFG, GAN, etc., to generate Honeywords, with the main goal of making it difficult to distinguish between Honeywords and real passwords.

V. CONCLUSION

With the development of technology, other authentication methods have emerged, but passwords were still a widely used authentication method. In this article, we introduced various methods of password guessing, most of which were based on Markov models, PCFG, NN, and GAN. In other words, we could divide the models mentioned in the article into two categories: probability-based models and deep learning-based models. Markov and PCFG were both related to probability, with the difference being that Markov predicted the next character based on the previous character in the password, while PCFG predicted the next character based on the structure of the password (numbers, letters, special characters). PCFG could be regarded as an optimization of Markov methods, but both had the problem of high computational complexity. RNN, GAN and other related models belonged to deep learning models. There were many types of models in this part. For optimization under the same model, basically, the later model performed better than the previous model.

Some experts have also proposed different types of models for application in password guessing. The field of password guessing, as a relatively new research field, has also benefited from the rapid growth of neural networks and deep learning in the past two years. In this article, we mentioned several feasible future research directions and hoped that researchers could pay attention to and find feasible solutions.

REFERENCES

[1] A. Narayanan and V. Shmatikov, “Fast dictionary attacks on passwords using time-space tradeoff,” Computer and Communications Security, pp. 364–372, 2005.

[2] D. Suleiman, A. Awajan, and W. Al Etaiwi, “The Use of Hidden Markov Model in Natural ARABIC Language Processing: a survey,” Procedia Computer Science, vol. 113, pp. 240-247, 2017.

[3] M. Dürmuth, F. Angelstorf, C. Castelluccia, D. Perito, and A. Chaabane, “OMEN: Faster password guessing using an ordered markov enumerator,” Lecture Notes in Computer Science, vol.8978, pp. 119–132, 2015.

[4] M. Weir, S. Aggarwal, B. De Medeiros, and B. Glodek, “Password cracking using probabilistic context-free grammars,” In Proceedings of the 2009 30th IEEE Symposium on Security and Privacy, pp. 391–405, 2009.

[5] W. Melicher, et al., “Fast, lean, and accurate: Modeling password guessability using neural networks,” In Proceedings of the 25th USENIX Security Symposium, pp. 175–191, 2016.

[6] R. Morris and K. Thomson, “Password security: A case history,” In Communications of the ACM, vol. 22, pp. 594–597, 1979.

[7] Y. Liu, et al., “GENPass: A general deep learning model for password guessing with PCFG rules and adversarial generation,” In Proceedings of the 2018 IEEE International Conference on Communications (ICC), pp. 1–6, 2018.

[8] M. Zhang, Q. Zhang, X. Hu, and W. Liu, “A Password Cracking Method Based On Structure Partition and BiLSTM Recurrent Neural Network,” In Proceedings of the Eighth International Conference on Communication and Network Security, pp. 79–83, 2018.

[9] B. Hitaj, P. Gasti, G. Ateniese, and F. Perez-Cruz, “PassGAN: A Deep Learning Approach for Password Guessing,” Applied Cryptography and Network Security, Lecture Notes in Computer Science, vol.11464, pp. 217-237, 2019.

[10] F. Yu and M. Vargas Martin, “GNPassGAN: Improved Generative Adversarial Networks For Trawling Offline Password Guessing,” 2022 IEEE European Symposium on Security and Privacy Workshops, pp. 10-18, 2022.

[11] J. Ye, M. Jin, G. Gong, R. Shen, and H. Lu, “PassTCN-PPLL: A Password Guessing Model Based on Probability Label Learning and Temporal Convolutional Neural Network,” Sensors 2022, vol.22(17), Article Number: 6484, 2022.

[12] M. Hellman, “A cryptanalytic time-memory trade-off,” IEEE Trans, vol.26, pp. 401–406, 1980.

[13] X. Guo, K. Tan, Y. Liu, M. Jin, and H. Lu, “LPG–PCFG: An Improved Probabilistic Context- Free Grammar to Hit Low-Probability Passwords,” Sensors 2022, vol.22(12), Article Number: 4604, 2022.

[14] J. Jiang, A. Zhou, L. Liu, and L. Zhang, “OMECDN: A Password-Generation Model Based on an Ordered Markov Enumerator and Critic Discriminant Network,” Applied Sciences, vol.12(23), Article Number: 12379, 2022.

[15] Y. Wu, X. Wan, X. Guan, T.Ji, and F.Ye, “PGTCN: A Novel Password-Guessing Model Based on Temporal Convolution

Network," Journal of Network and Computer Applications, vol.213, pp. 103592, 2023.

[16] P. Oechslin, "Making a Faster Cryptanalytic Time-Memory Trade-Off," Lecture Notes in Computer Science, vol. 2729, pp. 617-630, 2003.

[17] T. Zhou, H. Wu, H. Lu, P. Xu, and Y. Cheung, "Password Guessing Based on GAN with Gumbel-Softmax," Security and Communication Networks, vol. 2022, Article Number: 5670629, 2022.

[18] X. Guo, Y. Liu, K. Tan, W. Mao, M. Jin, and H. Lu, "Dynamic Markov Model: Password Guessing Using Probability Adjustment Method," Applied Sciences, vol. 11(10), Article Number: 4607, 2021.

[19] J.Chen and J. S. Rosenthal, "Decrypting classical cipher text using Markov chain Monte Carlo," Statistics and Computing, vol. 22, pp. 397-413, 2012.

[20] L. Liu, et al., "On the Variance of the Adaptive Learning Rate and Beyond," International Conference on Learning Representations, arxiv. 1908.03265, 2019.

[21] Z. Chen, V. Badrinarayanan, C. Y. Lee, and A. Rabinovich "GradNorm: Gradient Normalization for Adaptive Loss Balancing in Deep Multitask Networks," Proceedings of Machine Learning Research, vol.80, pp. 794-803, 2018.

[22] Y. L. Wu, H. H. Shuai, Z. R. Tam, and H. Y. Chiu, "Gradient Normalization for Generative Adversarial Networks," International Conference on Computer Vision, pp. 6353-6362, 2021.

[23] L. Rabiner and B. Juang, "An introduction to Hidden Markov Models," IEEE ASSP Magazine, vol.3, pp. 4-16, 1986.

[24] A. Kanta, I. Coisel, and M. Scanlon, "A Novel Dictionary Generation Methodology for Contextual-Based Password Cracking," IEEE Access, vol. 10, pp. 59178-59188, 2022.

[25] X. Li, J. Thickstun, I. Gulrajani, P. Liang, and T. Hashimoto, "Diffusion-LM Improves Controllable Text Generation," Advances in Neural Information Processing Systems, vol. 35, pp. 4328-4343, 2022.

[26] J. Lovelace, V. Kishore, C. Wan, E. Shekhtman, and K. Q. Weinberger, "Latent Diffusion for Language Generation," arxiv. 2212.09462, 2022.

[27] S. Nam, S. Jeon, and J. Moon, "Generating Optimized Guessing Candidates toward Better Password Cracking from Multi-Dictionaries Using Relativistic GAN," Applied Sciences, vol. 10(20), pp. 1-19, 2020.

[28] X. He, H. Cheng, J. Xie, P. Wang, and K. Liang, "Passtrans: An Improved Password Reuse Model Based on Transformer," 2022 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 3044-3048, 2022.

[29] S. Murmu, H. Kasyap, and S. Tripathy, "PassMon: A Technique for Password Generation and Strength Estimation," Journal of Network and Systems Management, vol. 30(1), Article number: 13, 2022.

[30] M. Siponen, P. Puhakainen, and A. Vance, "Can individuals' neutralization techniques be overcome? A field experiment on password policy," Computers and Security, vol. 88(C), 2020.

[31] R. Veras, C. Collins, and J. Thorpe, "A Large-Scale Analysis of the Semantic Password Model and Linguistic Patterns in Passwords," ACM Transactions on Privacy and Security, vol. 24(3), pp. 1-21, 2021.

[32] D. Wang, P. Wang, D. He, and Y. Tian, "Birthday, name and bifacial-security: understanding passwords of Chinese web users," In Proceedings of the 28th USENIX Conference on Security Symposium, pp. 1537–1554, 2019.

[33] G. Chang, L. Zhao, and W. Chen, "MLSTM:A Password Guessing Method Based on Multiple Sequence Length LSTM," Computer Science, vol. 49(4), pp. 354-361, 2022.

[34] G. Pagnotta, D. Hitaj, F. D. Gaspari, and L. V. Mancini, "PassFlow: Guessing Passwords with Generative Flows," International Conference on Dependable Systems and Networks, pp. 251-262, 2022.

[35] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral Normalization for Generative Adversarial Networks," International Conference on Learning Representations, arxiv. 1802.05957, 2018.

[36] J. Rando, F. Pérez-Cruz, and B. Hitaj, "PassGPT: Password Modeling and (Guided) Generation with Large Language Models," arxiv. 2306.01545, 2023.

[37] I. J. Goodfellow, et al., "Generative adversarial nets," International Conference on Neural Information Processing Systems, vol. 2, pp. 2672‑2680, 2014.

[38] X. Zhang, X. Zhang, J. Hu, and Y. Zhu, "A New Targeted Online Password Guessing Algorithm Based on Old Password," International Conference on Computer Supported Cooperative Work in Design, pp. 1470-1475, 2023.

[39] D. Wang, Y. Zou, Q. Dong, Y. Song, and X. Huang, "How to Attack and Generate Honeywords," 2022 IEEE Symposium on Security and Privacy (SP), pp. 966-983, 2022.

[40] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets," Computer Science, arxiv. 1411.1784, 2014:.

[41] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," International Conference on Learning Representations, arXiv. 1511.06434, 2015.

[42] Y. Chen, Q. Gao, and X. Wang, "Inferential Wasserstein Generative Adversarial Networks," Journal of the Royal Statistical Society Series, vol. 84(1), pp. 83-113, 2022.

[43] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," International Conference on Machine Learning, vol. 70, pp. 214-223, 2017.

[44] J. Kaevrestad, F. Eriksson, and M. Nohlberg, "Understanding passwords ‑ a taxonomy of password creation strategies," Information and Computer Security, vol. 27(3), pp. 453-467, 2019.

[45] S. Houshmand, S. Aggarwal, and R. Flood, "Next Gen PCFG Password Cracking," In IEEE Transactions on Information Forensics and Security, vol. 10(8), pp. 1776-1791, 2015.

[46] Y. Wang, W. Qiu, W. Zhang, H. Tian, and S. Li, "SE#PCFG: Semantically Enhanced PCFG for Password Analysis and Cracking," arxiv. 2306.06824, 2023.

[47] J. Xie, H. Cheng, R. Zhu, P. Wang, and K. Liang, "WordMarkov: A New Password Probability Model of Semantics," IEEE International Conference on Acoustics, pp. 3034-3038, 2022.

[48] Z. Xia, P. Yi, Y. Liu, B. Jiang, W. Wang, and T. Zhu, "GENPass: A Multi-Source Deep Learning Model for Password Guessing," In IEEE Transactions on Multimedia, vol. 22(5), pp. 1323-1332, 2020.

[49] H. Li, M. Chen, S. Yan, C. Jia, and Z. Li, "Password Guessing via Neural Language Modeling," In Machine Learning for Cyber Security, Lecture Notes in Computer Science, vol. 11806, pp. 78‑93, 2019.

 ISBN: 978-1-68558-104-6