2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD) | 978-1-6654-0527-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/CSCWD54268.2022.9776149

PG-Pass:Targeted Online Password Guessing Model based on Pointer Generator Network

Yang Li, Yong Li, Xi Chen, Ruixin Shi, Jizhong Han
Institute of Information Engineering, Chinese Academy of Sciences
School of Cyber Security, University of Chinese Academy of Sciences
Beijing,China
[liyang8119, liyong, chenxi, shiruixin, hanjizhong]@iie.ac.cn

***Abstract*—Existing targeted online password guessing models were based on Probabilistic Context-Free Grammars (PCFG), having inherent disadvantages that guessing structures were always the same for different users. This problem would lead to poor guessing efficiency. In order to solve this problem, we propose a targeted online password guessing model (PG-Pass) composed of the pointer generator network. It could automatically learn the impact of personal information on passwords and guess the target user's password more accurately. Through extensive experiments, we obtain the optimal parameters. The results show that with only Personally Identifiable Information (PII), the guessing success rate of the PG-Pass model could reach 19.49% in guessing once, which is ten times higher than TarGuess-I. When guessing 100 times, the guessing success rate could be 41.07%, proving the effectiveness of the proposed model in this paper.**

***Keywords*—targeted online password guessing, deep learning, pointer generator network**

I. Introduction

Text password is one of the most popular authentication technologies. It has the advantages of easy operation and familiarity with users, which will still be used in the foreseeable future [1]. As too many website passwords need users to remember, users always choose passwords far from random [2] [3]. They pick easy-to-remember strings as part of the password, such as personal information, which gives the attacker a chance. Most websites enforce strict password policies and arrange website password strength meters to help users build safe passwords. However, these password strength meters are proven to be ad-hoc and inconsistent [4]. Moreover, several password guessing models have been proposed to measure password strength.

According to the attack mode and personal information used, password guessing models can be divided into targeted online guessing and trawling offline guessing. More recent researches [5] [6] [7] conducted on trawling offline guessing. Owing to personally identifiable information (PII) becoming available [8], concerns have arisen about targeted online attacks. As far as we know, in targeted online guessing, Das et al. [9] studied passwords reusing and proposed a cross-site password guessing model for the first time. Then, Li et al. [10] analyzed how PII influences passwords. By adding PII to the probabilistic context-free grammars (PCFG), they have put forward the Personal-PCFG model. Afterward, Wang et al. [8] systematically characterized typical targeted guessing scenarios and creatively proposed the TarGuess-I model. However, these models were all based on the PCFG, which has limits due to the fixed order of generating password structures. This is inconsistent with user's diverse habits of composing passwords.

This paper focuses on targeted online password guessing. We propose a targeted online password guessing model (PG-Pass) based on the pointer generator network [11], with the aim of guessing target users' passwords more accurately. The pointer generator network was chosen because it is one of the most practical methods in automatic text summarization. Based on the copying mechanism, the pointer generator network increases the ability to deal with out of vocabulary (OOV) words while keeping the capacity to generate new words. We believe that generating a password based PII is similar to generating a summary from the text. The experimental results are promising and show that the PG-Pass model can outperform all previous targeted online password guessing models with fewer guessing times.

We summarize our main contributions as follows:

- We propose a new targeted online password guessing model (PG-Pass), which can learn to compose passwords well based on PII and generate high-quality guesses. To the best of our knowledge, it is the first time the pointer generator network has applied to targeted online guessing.
- We designed the overall PG-Pass model framework, including three steps for data processing, model training, and password generation. Through extensive experiments, we obtain the optimal parameters.
- Experiment results demonstrate the PG-Pass model attaining state-of-the-art performance in targeted online password guessing. The guessing success rate of the PG-Pass model could reach 19.49% when guessing once.

This paper is organized as follows: Sect.II gives a brief overview of related work. Sect.III presents the model framework. Sect.IV conducts the experiments and analyzes the effect of our model. Our conclusions are drawn in the final section.

II. Related Work

The research time of targeted online password guessing is shorter compared with trawling offline guessing. Some preliminary work has been carried out over the past five years.

It is believable that two factors may influence the guessing success rate obtained. One is how to characterize the PII in the password and the other is to find a suitable model to generate passwords based on PII. As far as we know, previous work has mainly focused on characterizing the PII in the password.

In 2016, li et al. [10] explored the extent to which a user's PII resides in a password. The analysis results extended the PCFG to be semantics-rich, and they proposed the Personal-PCFG model. Apart from "L, " D, " and "S" symbols in PCFG, Personal-PCFG added six types of PII, including birthday, name, email address, account name, mobile phone number, and ID number. The model adopted a length-based PII matching method. They matched the entire password with PII, recorded the matching string's length, and replaced it with a PII symbol. For unmatched strings in the password, Personal-PCFG replaced them with LDS symbols. In the generation phase, Personal-PCFG replaced LDS and PII symbols with segments in turn based on the sorted structures.

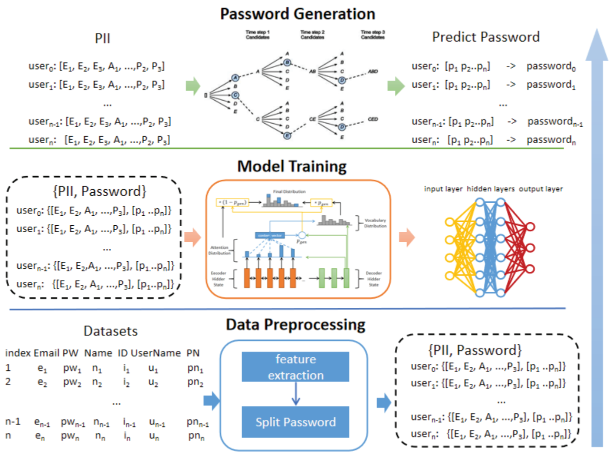
In 2016, Wang et al. [8] systematically analyzed typical targeted guessing scenarios and proposed the targeted online guessing framework TarGuess. They discovered weaknesses in the length-based PII matching method, which may underestimate or overestimate PII use. Also, the length-based tags are insensitive to their subtypes. Therefore, they put forward a type-based PII matching method. For a type-based PII tag, its subscript number stands for a particular subtype of one PII but not the length. Based on the type-based PII tags and PCFG model, they proposed the TarGuess-I model for scenario one and improved the guessing success rate. Wang et al.' work [8] has led to the revision of NIST SP800-63-2.

In 2020, Xie [12] [13] found that some effective semantic tags have not been testified and employed in the TarGuess-I model. After adding three types of semantic tags into TarGuess-I, they proposed $TarGuess - I^{+KPX}$. "P" stands for the popular password list created from a dataset similar to the target website; "K" stands for identifying password segments with physical location sequence on the QWERTY keyboard; "X" stands for continuous characters from the user-generated PII. They further improved the effect of the TarGuess-I model.

The main weakness in their studies is that they made no attempt to try new models. All these works are largely based on PCFG. In the password generation phase, PCFG obtains candidate passwords by instantiating the password structure frequency table. The guessing order of password structure is the same for different users, and the difference lies in the filled segments. In fact, the password structures of different users are not arranged in the same order. Therefore, there is certainly room for improvement. Then, we introduce the PG-Pass model proposed by this paper in the latter sections.

III. MODEL FRAMEWORK

In this section, we will discuss the PG-Pass model framework in detail. As shown in Fig.1, the PG-Pass model framework consists of three steps: data processing, model training, and password generation. Firstly, data processing characterizes and labels the PII, slices the password based on these PII tags and organizes these data into a format supported by the pointer generator network. Secondly, in the training phase, the pointer generator network learns significant relationships between PII and passwords and simulates password distribution. Thirdly, using beam search, the pointer generation network generates guessing passwords based on the target user's PII.



Password Generation
PII
Predict Password
user0: [E1, E2, E3, A1, ...,P2, P3]
user1: [E1, E2, E3, A1, ...,P2, P3]
usern-1: [E1, E2, E3, A1, ...,P2, P3]
usern: [E1, E2, E3, A1, ...,P2, P3]
user0: [p1 p2..pn] -> password0
user1: [p1 p2..pn] -> password1
usern-1: [p1 p2..pn] -> passwordn-1
usern: [p1 p2..pn] -> passwordn
Model Training
{PII, Password}
input layer hidden layers output layer
Data Preprocessing
Datasets
index Email PW Name ID UserName PN
feature extraction
Split Password
{PII, Password}

Fig. 1. The framework of the PG-Pass model.

A. Data Preprocessing

Passwords can be regarded as short text, but they are somewhat different from natural languages. For example, passwords are not made up of Chinese characters but Pinyin letters or single characters compared to Chinese text. Compared with English text, there is no obvious space separator. Therefore, natural language processing models cannot be used in password text directly.

Taking the 12306 (a train ticketing websites) dataset as an example, the original PII includes the five attributes: name, email address, mobile phone number, account name, and ID number. The name is composed of Chinese characters. The birthday is hidden in the ID number. So we need to extract PII and characterize them with tags. We split the user's password using the string matching method based on PII tags for training data. Then we split the unmatched strings into single characters in the password.

B. Model Training

The input dataset D contains two parts in the training phase, PII and password, which can be expressed as follows. For example, U_1 represents the first user's PII, P_1 represents the first user's password. At the same time, $U_1 = [u_{11}, ., u_{1k}]$ and $P_1 = [p_{11}, ., p_{1l}]$ contain several valus respectively, among them, u_{11} represents the first PII tag of the first user's PII, p_{11} represents the first segment of the first user's password.

$$D = \{[U_1, P_1], [U_2, P_2], ..., [U_n, P_n]\}$$

The problem is analogy to generating passwords under the guidance of PII. In this paper, we make use of pointer generator network. The framework of pointer generator network can be seen in Fig.2. Here, we take PII as the original article and password segments as the text summary.

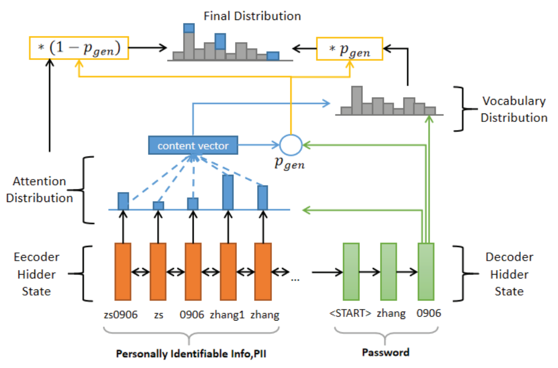


Fig. 2. The Framework of Pointer Generator Network.

1) Seq2seq with attention: The pointer generator network is based on the sequence-to-sequence (Seq2Seq) with attention [14]. The PII is encoded into a hidden state in the middle layer and then decoded into another representation. During the encoding, the model uses a single-layer bidirectional Long-Short-Term-Memory (LSTM) network to calculate the hidden status h of the PII and the hidden status h_i of the t-th tags in the PII. During the decoding, the model adds an attention mechanism and uses a single-layer unidirectional LSTM. In the training phase, each step t the model input the previous tags of the PII to get the decoded status s_t. Use h_i and s_t to get attention weight a^t of the tags in PII this moment.

$$e_i^t = v^T tanh(W_h h_i + W_s s_t + b_{attn})$$

$$a^t = softmax(e^t)$$

Next, the attention weight a^t and the hidden state h_i calculate a context vector h_t^*.

$$h_t^* = \sum_i a_i^t h_i$$

The context vector h_t^* is concatenated with the decoder state s_t and fed through two linear layers to produce the vocabulary distribution P_{vocab}. P_{vocab} is a probability distribution over all words in the vocabulary.

$$P_{vocab} = softmax(V'(V[s_t, h_t^*] + b) + b')$$

2) Pointer Generator Network: Pointer generator network introduced a weight p_{gen}. According to the context vector h_t^*, the decoder status s_t, and the decoder input x_t in the seq2seq model, calculate the generation probability p_{gen} of step t.

$$p_{gen} = \sigma(w_{h^*}^T) h_t^* + w_s^T s_t + w_x^T x_t + b_{ptr}$$

$p_{gen} \in [0, 1]$ is a soft switch that decides whether to generate a word in the vocabulary or copy a word from the input PII through the attention distribution. There is an OOV extended vocabulary for each input PII, which stores all unregistered words in the given vocabulary. After introducing the OOV vocabulary, the calculation formula for the probability

TABLE I
BASIC INFORMATION OF THE PASSWORD DATASET USED IN THIS PAPER

Dataset	Web service	Total Passwords	Leaked Time	With PII
CSDN	Programmer	6,428,632	2011	
dodonew	E-commerce	16,283,140	2011	
12306	Train ticketing	130,347	2014	$\surd$

of generating the target word w at the decoding time step t is as follows:

$$P_{OOV} = \sum_{i:w_i=w} a_i^t$$

$$P(w) = p_{gen} P_{vocab}(w) + (1 - p_{gen}) P_{OOV}(w)$$

C. Password Generation

The PG-Pass model inputs the content of the type-based PII tags to sequentially generate password segments. During the training phase, the number of password segments is fixed. We pad the remaining part with spaces or truncate portions beyond a fixed length for passwords that do not meet the length requirement. Therefore, the number of password fragments generated by the model is also fixed. After generation, we remove the spaces before the terminator and join password segments to form the final guessed password. When the model predicts the next password segment, we use the beam search method with the parameter set to K. At each step to predict the next password segment, we take the top K password segments in order of predicted probability. Finally, we obtain K-predicted passwords.

IV. Experiment and Evaluation

A. Experiment Setup

1) Datasets: We use three public real-world datasets, i.e., CSDN, Dodonew and 12306 in Table I. Because the comparison experiment results used in this paper are taken from [8], to ensure the fairness of the comparison, the data set and processing method we use are the same as [8] as possible. However, the number of passwords in our final dataset is still slightly different. The maximum difference rate of passwords number among the three datasets is 0.8%. The comparison results with the TarGuess-I model and our model PG-Pass based on dataset 12306 are shown in Sect. IV-D1.

We set the two datasets CSDN and Dodonew as test sets to simulate real online attacks in the experiment Sect. IV-D2. PII-CSDN and PII-Dodonew are get by matching CSDN and Dodonew with 12306 using email addresses. The passwords numbers in matching datasets PII-CSDN and PII-Dodonew are shown in Table II. Compared with [8], the maximum difference rate of passwords number in the two datasets is 1.76%, which is receivable.

2) Pointer generator network parameters: This section introduces the setting of some parameters. The hidden state dimension in the seq2seq model is set to 256. The word embedding dimension is set to 128. The pointer generation network uses only the pointer network, not the coverage mechanism.

TABLE II
BASIC INFORMATION OF PASSWORD PAIR SET BY MATCHING CSDN AND DODONEW WITH 12306 USING EMAIL ADDRESS

Dataset	Total Passwords	With PII
PII-Dodonew	49,567	√
PII-CSDN	12,412	√

In the training phase, we use Adagrad with a learning rate of 0.15. The batch size is set to 8 during training and testing. We truncate or fill the PII into 400 tokens and limit the password segment to 15 tokens.

3) Guessing number: Each website has a specific limit on the password guesses numbers to defend against attacks. According to the U.S. National Identity Certification Guide NIST sp800-63-2, for level 1 or 2 systems, the number of false logins allowed for each account should not exceed 100 within 30 days [8]. So in this paper, we set the upper limit of guess number to 100. Namely, we only guess 100 times for each user account.

B. Data Preprocessing

Several different methods of characterizing the PII in the password are used in the experiments.

Method 1: This method is at a character level. We extract email prefix, user name, name (pinyin format), birthday, and mobile phone number from PII, dividing them into single characters. Similarly, the password is also split into characters.

Method 2: We adopt the 29 type-based PII tags as defined in [8]. The 29 type-based PII tags include N1,..., N7, B1,..., B10, A1, A2, A3, E1, E2, E3, P1, P2, I1, I2, I3, where: (1) N stands for name usage, N1 for the usage of full name, N2 for the abbr. of full name, N3 for family name, N4 for given name, N5 for the 1st letter of the given name + family name, N6 for last name+the 1st letter of the given name, N7 for family name with its 1st letter capitalized; (2) B stands for birthday usages, B1 for full birthday in YMD format, B2 for full birthday in MDY, B2 have been specified earlier, B3 stands for full birthday in DMY, B4 for the date in birthday, B5 for the year in birthday, B6 for Year+Month, B7 for Month+Year, B8 for the last two digits of year + date in MD format, B9 for date in MD format + the last two digits of year, B10 for date in DM format + the last two digits of year; (3) A stands for account name usages, A1 for full account name, A2 for the (first) letter-segment of account name, A3 for the (first) digital segment of account name; (4) E stands for email prefix usages, E1 for the full email prefix, E2 for the first letter segment of email prefix, E3 for the first digital segment of account name; (5) P stands for mobile phone number usages, P1 for the full number, P2 for the first three digits, P3 for the last four digits; (6) I stands for the Chinese Notional Identification number, I1 for the last 4 digits, I2 for the first 3 digits, I3 for the first 6 digits. We cut the password using the type-based matching method, and the unmatched string in the password is split into single characters.

Method 3: This method is similar to method 2, but some of the 29 type-based PII tags are different, including email prefix, account name, name, and birthday. We divide email prefixes and account names into segments according to character types separately. We use pinyin and abbr. corresponding to each Chinese character as a separate segment in names respectively, and family name with its first letter capitalized. We adopt year in YYYY and YY format, month in MM format, and date in DD format for birthdays.

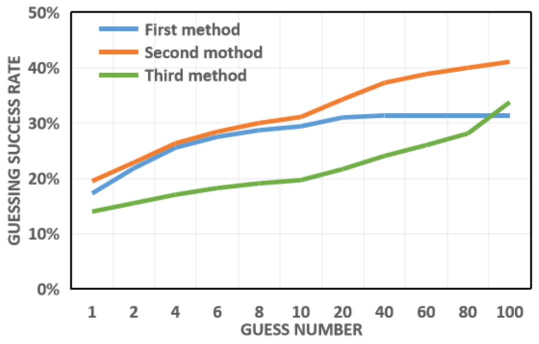


Fig. 3. The results under different PII tags characterize method.

TABLE III
THREE DIFFERENT TYPE-BASED PII TAGS ORDER

index	Tags Order
1	E1-E3, A1-A3, N1-N7, B1-B10, I1-I3, P1-P3
2	N1-N7, B1-B10, A1-A3, E1-E3, P1-P3, I1-I3
3	I1-I3, P1-P3, A1-A3, N1-N7, B1-B10, E1-E3

The experimental results are shown in Fig.3. The performance of method 2 is the best. Our experiments also confirm type-based PII tags are highly adaptive.

C. Parameter Optimization

This section analyzes the effect of type-based PII tags order and determines the value of vocabulary size.

1) PII order: The order of words in a sentence is important for natural language processing. We tested whether PII order affects the experimental result, following three different PII orders shown in Table III. The representation of the structure introduced in Sect. IV-B. From Fig.4 (A), we note that the PII order nearly does not affect guessing success rates.

2) Vocabulary size: We calculate guessing success rates using different vocabulary sizes. In Fig.4 (B), the results show that when the vocabulary size is between 30000 and 80000, guessing success rates are similar. In this paper, we set the vocabulary size as 50000.

D. Experimental Results

1) Comparison result on the dataset 12306: With the optimal model parameters, we compared the guessing success rate of the PG-Pass with TarGuess-I, Personal-PCFG, and PCFG models based on the 12306 dataset. Guess success rate is a standard indicator in password guessing. The higher it is, the better the model works. We report the comparison results in Fig.5. We use the same setting as [8] in experiments for the sake of fairness. 50% data in the dataset 12306 is used

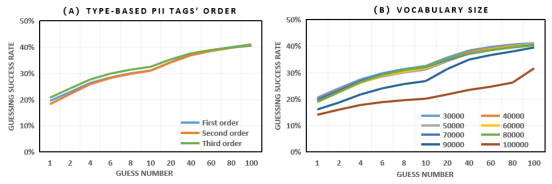


Fig. 4. The guessing success rate under different Type-based PII tags order and vocabulary size.

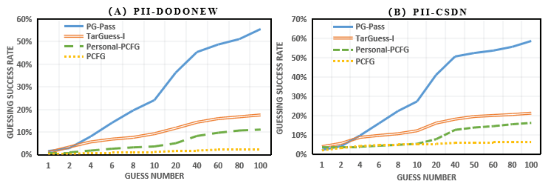


Fig. 6. Comparison between PG-Pass and TarGuess-I, Personal-PCFG, PCFG based on Dodonew and CSDN dataset.

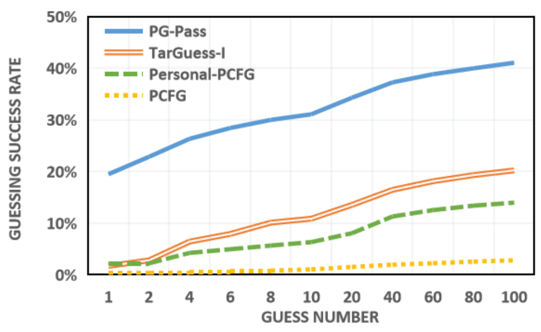


Fig. 5. Comparison between PG-Pass and TarGuess-I, Personal-PCFG, PCFG based on 12306 dataset.

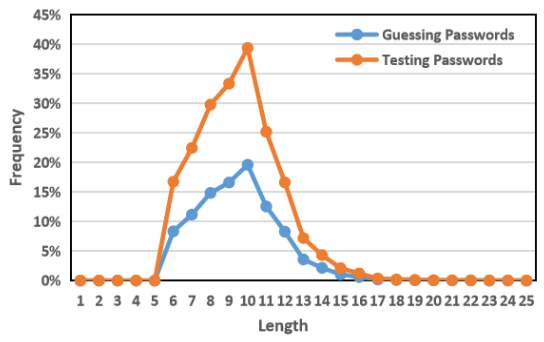


Fig. 7. The length distribution of the guessing password and real password based on 12306 dataset.

for training, and the remaining 50% is used for testing. The guessing success rates produced by the models TarGuess-I, Personal-PCFG, and PCFG are taken from [8]. As shown in Fig.5, the guessing success rate of the PG-Pass model is about 100%˜1000% higher than TarGuess-I. The PG-Pass model reached 19.49% when guessing only once, about ten times higher than that of TarGuess-I. Furthermore, the guessing success rate of the PG-Pass model reached up to 41.07% when guessing 100 times, about twice that of TarGuess-I.

The above results of the PG-Pass model suggest that the model can well learn: (I) how PII can influence generating passwords; (II) how to construct passwords.

2) Real-world Practice: Usually, the data source of training data and target users varies from different websites in practical applications. We simulate online attacks in order to gain a better verification effect. We train on the dataset 12306 and test on the matching datasets PII-CSDN and PII-Dodonew respectively. In Fig.6, the guessing success rate of our model PG-Pass is 42%˜216% higher than that of the TarGuess-I model on the Dodonew dataset, and 11%˜179% higher than that of the TarGuess-I model on the CSDN dataset. The guessing success rates on the Dodonew and CSDN datasets can reach 55.59% and 58.73% when guessing 100 times respectively. The experiment results indicate that the PG-Pass model has good adaptability and applicability.

E. Results Analysis

In this section, we analyze the successfully guessed passwords from the length and structure distribution.

1) Length analysis: We count the guessing passwords' length distribution and the passwords' length distribution in the test set based on the 12306 dataset separately, as shown in Fig.7. We find that the guessing password length distribution is similar to the actual passwords. Passwords with 10-character length account for the most prominent part, followed by 9-character and 8-character. It also reflects that the PG-Pass model can learn the composition of passwords well and generate passwords with a certain proportion.

2) Structural Analysis: Firstly we analyzed the password structures of guessing successful in the 12306 dataset. Fig.8 shows a clear trend in the structure number generated under different guessing times. When guessing once, the PG-Pass model tried 22 password structures for all user accounts. When guessing 100 times, the PG-Pass model tried 364 password structures. The diversity of generated structures might be one reason for the good performance of the PG-Pass model. Moreover, we compare the high-frequency structures of the guessing success passwords with real passwords. Based on the 12306 dataset, the top ten password structures when guessed 100 times and the top ten password structures in the test set are shown in Table IV. The representation of the structure introduced in Sect. IV-B. In the first four structures, they are exactly the same. In the first ten structures, only two structures' order are different. It has proved that the PG-Pass model can learn the structural characteristics of passwords well.

Finally, we conducted a more careful analysis of different datasets. Fig.9 shows the proportions of the ten most frequently occurring structures in the 12306, PII-Dodonew, PII-CSDN test set and the proportion of each structure guessing correct passwords at 100 guessing times. Compared to the PII-Dodonew and PII-CSDN datasets, the model achieved a relatively higher guessing success rate in each of the top ten

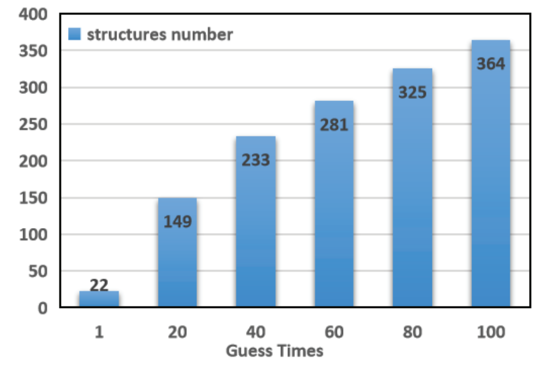


Fig. 8. The structures number of the guessing password under different guess times.

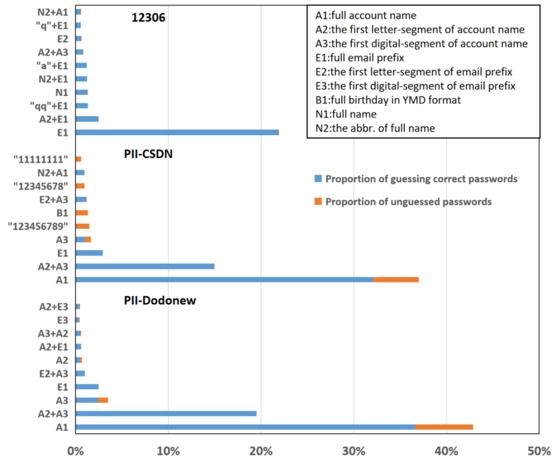


Fig. 9. The proportion of the ten most frequently occurring structures in the 12306, PII-Dodonew, PII-CSDN test sets and the proportion of each structure guessing correct at 100 guessing times.

TABLE IV
TOP TEN PASSWORD STRUCTURES COMPARISON

index	Guessing 100 times	Test set
1	E1	E1
2	A2+E1	A2+E1
3	”qq”+E1	“qq”+E1
4	N1	N1
5	“a”+E1	N2+E1
6	N2+E1	“a”+E1
7	A2+A3	A2+A3
8	E2	E2
9	“q”+E1	“q”+E1
10	N2+A1	N2+A1

structures of the 12306 dataset. However, because the 12306 dataset structure distribution is relatively dispersed, the PII-Dodonew and PII-CSDN dataset structure distribution is more concentrated, and the number of model generated structures is limited, the guessing success rate based on the 12306 dataset is lower than that of PII-Dodonew and PII-CSDN when guessing 100 times. The guessing success rates of the 12306, PII-Dodonew and PII-CSDN datasets are 41.07%,55.59% and 58.73%.

V. Conclusion

This paper proposes the PG-Pass model, a targeted online password guessing model PG-Pass based on pointer generator network for the first time. It is also the first study that the pointer generator network applied in targeted online password guessing. We designed the model framework and obtained the optimal parameters through detailed experiments. The results show PG-Pass model has the ability to outperform previous target online password guessing models. Guessing passwords is close to the actual passwords in length distribution and structure composition. Simulating actual online attacks, the model shows the effectiveness. It reminds users that the targeted online password guessing should be paid more attention to, and website passwords should be set more complicated.

References

[1] C. Herley and P. Van Oorschot, “A research agenda acknowledging the persistence of passwords,” *IEEE Security & privacy*, vol. 10, no. 1, pp. 28–36, 2011.

[2] J. Bonneau, “The science of guessing: analyzing an anonymized corpus of 70 million passwords,” in *2012 IEEE Symposium on Security and Privacy*. IEEE, 2012, pp. 538–552.

[3] D. Wang, P. Wang, D. He, and Y. Tian, “Birthday, name and bifacial-security: understanding passwords of chinese web users,” in *28th USENIX Security Symposium (USENIX Security 19)*, 2019, pp. 1537–1555.

[4] X. de Carné de Carnavalet and M. Mannan, “From very weak to very strong: Analyzing password-strength meters,” in *Network and Distributed System Security Symposium (NDSS 2014)*. Internet Society, 2014.

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

[6] W. Melicher, B. Ur, S. M. Segreti, S. Komanduri, L. Bauer, N. Christin, and L. F. Cranor, “Fast, lean, and accurate: Modeling password guessability using neural networks,” in *25th {USENIX} Security Symposium ({USENIX} Security 16)*, 2016, pp. 175–191.

[7] B. Hitaj, P. Gasti, G. Ateniese, and F. Perez-Cruz, “Passgan: A deep learning approach for password guessing,” in *International Conference on Applied Cryptography and Network Security*. Springer, 2019, pp. 217–237.

[8] D. Wang, Z. Zhang, P. Wang, J. Yan, and X. Huang, “Targeted online password guessing: An underestimated threat,” in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 1242–1254.

[9] A. Das, J. Bonneau, M. Caesar, N. Borisov, and X. Wang, “The tangled web of password reuse.” in *NDSS*, vol. 14, no. 2014, 2014, pp. 23–26.

[10] Y. Li, H. Wang, and K. Sun, “A study of personal information in human-chosen passwords and its security implications,” in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 2016, pp. 1–9.

[11] A. See, P. J. Liu, and C. D. Manning, “Get to the point: Summarization with pointer-generator networks,” *arXiv preprint arXiv:1704.04368*, 2017.

[12] Z. Xie, M. Zhang, Y. Guo, Z. Li, and H. Wang, “Modified password guessing methods based on targuess-i,” *Wireless Communications and Mobile Computing*, vol. 2020, pp. 1–22, 2020.

[13] Z. Xie, M. Zhang, A. Yin, and Z. Li, “A new targeted password guessing model,” in *Australasian Conference on Information Security and Privacy*. Springer, 2020, pp. 350–368.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in neural information processing systems*, 2017, pp. 5998–6008.