SE#PCFG: Semantically Enhanced PCFG for Password Analysis and Cracking

Yangde Wang, Weidong Qiu, Peng Tang, Hao Tian and Shujun Li, *Senior Member, IEEE*

***Abstract*—Much research has been done on user-generated textual passwords. Surprisingly, semantic information in such passwords remain under-investigated, with passwords created by English- and/or Chinese-speaking users being more studied with limited semantics. This paper fills this gap by proposing a *general framework* based on *semantically enhanced* PCFG (probabilistic context-free grammars) named SE#PCFG. It allowed us to consider 43 types of semantic information, the richest set considered so far, for password analysis. Applying SE#PCFG to 17 large leaked password databases of user speaking four languages (English, Chinese, German and French), we demonstrate its usefulness and report a wide range of new insights about password semantics at different levels such as cross-website password correlations. Furthermore, based on SE#PCFG and a new systematic smoothing method, we proposed the Semantically Enhanced Password Cracking Architecture (SEPCA), and compared its performance against three SOTA (state-of-the-art) benchmarks in terms of the password coverage rate: two other PCFG variants and neural network. Our experimental results showed that SEPCA outperformed all the three benchmarks consistently and significantly across 52 test cases, by up to 21.53%, 52.55% and 7.86%, respectively, at the user-level (with duplicate passwords). At the level of unique passwords, SEPCA also beats the three counterparts by up to 43.83%, 94.11% and 11.16%, respectively.**

***Index Terms*—Password security, semantically enhanced PCFG, empirical analysis, password cracking.**

I. Introduction

Textual passwords have dominated user authentication on computer systems and the Internet for decades [1]. Although many new user authentication methods (e.g., fingerprint and face recognition based methods) have been proposed and used widely on smartphones [2], textual passwords remain the most widely used method because none of the new methods can provide a better balance between security and usability. Many people believe that the situation will not change in the foreseeable future [3].

Trade-offs between security and usability have been well known in the cyber security field [4]. For textual passwords, it has been well recognized that users often define easy-to-remember passwords that are not strong enough against password cracking and prefer relying on themselves than using auxiliary tools [5].

This work was supported by the National Key Research and Development Program of China under grant number 2023YFB3106501.

Yangde Wang, Weidong Qiu and Peng Tang are with Shanghai Jiao Tong University, Shanghai, 200240, China.

Shujun Li is with University of Kent, Canterbury, CT2 7NP, UK.

Hao Tian is with Haitong Securities, Shanghai, 200001, China.

Corresponding co-authors: Yangde Wang (softds@163.com) and Shujun Li (hooklee@gmail.com).

The continuous dominance of textual passwords in user authentication means that it remains important to further our understanding of user-generated passwords to improve password security. There has been quite some research looking into semantic patterns of user-generated passwords, but most of which focused on English-speaking users [6], [7], [3], [8], [9] or more recently Chinese-speaking users [10], [11], [12], [13]. However, research covering users speaking more than English and Chinese is still very limited. In addition, past work studied semantic information using stand-alone methods, which means a gap on more reconfigurable frameworks that allow easy incorporation of a rich set of semantic elements. Yet another gap we noticed is that little work has quantitatively analyzed cross-site semantic correlations. Last but not the least, as mentioned in [14], little work has considered applying smoothing techniques to consider unobserved but still plausible passwords to make password cracking methods more generalizable.

This paper fills these gaps via the following main contributions. First, we propose SE#PCFG, semantically enhanced PCFG, a general framework for analyzing semantics of user-generated passwords. We implemented a prototype of SE#PCFG covering 43 types of password semantic information, the richest set considered so far for password analysis (to the best of our knowledge), including semantic information in four different languages (English, Chinese, German and French), entries in Wikipedia, Wiktionary and Urban Dictionary. Second, by applying our implementation of SE#PCFG to 17 large leaked password databases, we demonstrate its usefulness and report a range of new insights about password semantics and the underlying user behaviors such as cross-website password correlations. Third, we propose Semantically Enhanced Password Cracking Architecture (SEPCA), which can leverage training set more effectively, enhanced by a general and systematic smoothing algorithm. Using 52 test cases based on the same 17 password databases (each of four selected databases as the training set and each of the other 13 as the target set), we conducted experiments by comparing the performance of SEPCA against three state-of-the-art password cracking methods in terms of coverage rate: two other variants of the PCFG family – Weir et al.'s latest implementation [15] of the original PCFG-based method [16] and Veras et al.'s method based on their "Semantic PCFG" [9] – and FLA (Fast, Lean, and Accurate) that is n-gram-based and not semantically aware [17]. Our experimental results showed that SEPCA outperformed the two other PCFG variants consistently and significantly at both user- and password-levels. With 5×10^9 guessed passwords, SEPCA performed the best and the aver-

age performance across the 52 test cases was improved by up to 21.53% (user-level), 43.83% (password-level) for one and 52.55% (user-level), 94.11% (password-level) for the other. SEPCA also outperformed FLA by up to 7.86% (user-level) and 11.16% (password-level) averagely.

The rest of the paper is organized as follows. The next section summarizes the content of past related studies and provides a detailed comparison with this paper. The third section introduces SE#PCFG and how our implementation was applied to the 17 leaked password databases for password analysis. Section IV describes SEPCA in detail and reports experimental results. The fifth section provides an in-depth discussion of the experimental results presented in the previous section, summarizing the implications and guidance offered by our findings for both end users and researchers. Section VI includes the ethical statement of this paper, and the last section concludes our work.

II. Related Work and Comparison

A. Related Work

1) Password modeling methods: To better understand the habits of people building passwords, plenty of methods were proposed and evaluated by generating guessing passwords. In 2005, Narayanan and Shmatikov [18] proposed to use Markov models to guess passwords. Their work was further optimized by Dürmuth et al. in 2015 [19] by applying sorting algorithm. In 2014, Ma et al. considered varied orders of n-gram Markov models with additive smoothing for cracking English passwords [20]. In 2016, Melicher et al. [17] proposed to use neural networks to model passwords, and they showed that their work could perform well with less memory requirements. Since then, more machine learning-based methods were proposed. These methods not only based on static models learned from training data, but also can follow a more dynamic training process. In 2019, Hitaj et al. [21] proposed to use GAN (Generative Adversarial Networks) to train a password cracker. In 2021, Pasquini et al. [22] showed how the real distribution of target passwords can be interactively learned to facilitate password cracking. In 2023, Wang et al. [23] proposed re-encoding the password characters, which makes it possible to use traditional machine learning techniques such as random forests for cracking passwords. In the same year, Xu et al. [24] explored and optimized template-based password generation using the bi-transformer technology. In 2024, Li et al. [25] demonstrated the good generalization ability of pre-training and fine-tuning techniques in password analysis through a two-stage learning process based on transformers.

In 2009, a method based on the so-called probabilistic context-free grammars (PCFG) that can learn higher-level structural patterns than these character-level models, was proposed by Weir et al. [16]. Their method segment training passwords based on three different types of characters and generate guesses according to probability orders. In 2014, Ma et al. [20] reported that PCFG-based methods under-performed whole-string Markov models, revealing that simple PCFGs are not as powerful as they looked. Besides, to our surprise, few prior work aim to optimize PCFG by applying smoothing method. In 2015, Houshmand et al. showed the effectiveness of injecting keyboard patterns and using smoothing method limited on them [26]. In 2016, Komanduri designed a smoothing method in an ad hoc way that all types of non-terminals having one pre-defined value [27]. These two literature can be seen as initial attempts to use smoothing algorithm to optimize PCFG-based methods.

2) Password semantic analysis: Some researchers studied password semantics in order to better understand how users define passwords and to overcome limitations of Markov models and PCFG-based methods. In 1989, Riddle et al. [6] reported that names and dates (especially birthday dates) were often used in user-generated passwords. Through a survey of 218 participants and 1,783 passwords, Brown et al. [7] observed similar phenomena in 2004.

In 2014, Veras et al. [9] proposed to use NLP techniques to analyze linguistic semantics in user-defined passwords. In 2021, they reported some extended password semantic analysis work in [28], under the name "Semantic PCFG".

Work introduced above mainly considered passwords of English-speaking users. To fill the gap, a number of recent studies looked at leaked passwords from Chinese websites. In 2014, Li et al. [10] reported that Chinese users preferred using Pinyin and dates in their passwords. In 2016, Han et al. [29] reported some behavioral differences between Chinese and non-Chinese users on password composition, e.g., Chinese users preferred using digits more but non-Chinese users preferred using letters especially lower-case ones more. At the same year, Wang et al. designed a framework named "TarGuess" trying to inject various types of personal information to PCFG model to attack specific person over online-attack scenario. In 2017, Wang et al. [30] reported the observed use of other semantic elements including dates, palindrome, and math. In 2019, Wang et al. [12] re-confirmed some important semantic elements used by Chinese users such as Pinyin and dates. In 2021, Zhang et al. [31] looked at how digits in two groups and 12 types were used by Chinese users for defining passwords.

In addition to work on password semantics in English and Chinese passwords, some researchers also looked at passwords defined by users speaking other languages. For instance, AlSabah et al. [32] studied semantics in less than 66k passwords and demographic information of users leaked from a Middle Eastern bank, representing diverse cultural backgrounds (Arab, Filipino, Indian, and Pakistani) and non-English/Chinese languages the affected users likely spoke. The semantic information they looked at include names, keyboard patterns, phone numbers and birth years.

On the other hand, some literature also focused on the frequently-used non-linguistic semantics. In 2017, Wang et al. [33] studied eight types of transformation rules people usually applied to their passwords. In 2019, Liu et al. [34] systematically studied how to identify, order, and apply mangled-rules to widely used cracking tools. In 2021, Xu et al. [35] trained a Byte-Pair-Encoding algorithm to automatically obtain chunk vocabularies, and leveraged these information to optimize password models. In 2023, Li et al. [36] built an automatic mangling rule generator using density-based clustering to help generating passwords.

Besides research work, there are also many password cracking software tools such as hashcat [37] and John the Ripper (JtR) [38]. These tools typically use one or more password dictionaries and/or mangling rules to form different attacks, and usually do not incorporate more advanced methods discussed in the research literature. Since such tools are less advanced (in modeling passwords), in the rest of the paper, we will focus on the password analysis and cracking methods reported in research papers only.

B. Comparison With Related Work

In this subsection, we explain how our work compare with closest related work. Some terms proposed in our work, especially semantic factors (SFs) and semantic factor types (SFTs), are explained more in Section III.

Methods based on n-grams, such as those proposed by Narayanan & Shmatikov [18], Melicher et al. [17] and Pasquini et al.' [22], treat each sequence of n consecutive characters as an atomic element for password analysis and cracking, which often cannot be mapped to semantic information in any explicit way. While such methods have been proven very powerful in password cracking (more so than PCFG-based methods), they cannot be used to study password semantics.

Weir et al.'s work [16], [15] started to treat passwords as a series of meaningful components based on character types. Obviously, the lack of semantic awareness in their initial design limits their performance. All previous extensions of Weir et al.'s work were aware of this issue and tried to improve by injecting more semantics. Veras et al.'s work [9], [28] introduced NLP tools to develop semantics in English-speaking users. During the same period, [10], [11], [39], [12] tried to better model Chinese behaviors over both online and offline attacking scenario.

Compared with previous work, our work has significant differences in the following key technical aspects. 1) Other work utilized very limited semantic types for password analysis and used ad hoc methods to extract such semantic types, making it difficult to integrate all the different semantic types and extraction methods together into a more comprehensive and expandable framework. For example, Veras et al. [9], [28] focused on linguistic semantics in passwords, while others [10], [11], [39], [12] paid more attention on Chinese names or words in Pinyin. Besides, their choices on the semantic information can be seen as the results of casual observations and appear to be less systematic. In contrast, we followed a more systematic approach to identify different types of semantic information used for password generation by using Google to search for articles about "How to create strong passwords", leading to the most comprehensive coverage of semantic types used so far (see III-A2 for a detailed comparison). 2) Based on the comprehensive semantic information considered in our work, we further propose a new and general smoothing method to address unobserved semantic patterns in passwords, as described in Section IV-A. 3) To validate the effectiveness of our work, we conducted experiments on the largest collection of leaked password database used in the research literature so far (to the best of our knowledge), which includes passwords from 17 datasets, covering four mainstream languages and 310 million passwords.

III. SE#PCFG and Password Semantic Analysis

In this section, we first describe the conceptual model behind SE#PCFG, then introduce a streamlined computational process which can tackle different languages and richer semantics, and finally report some selected experimental results by applying our work to analyze 17 large leaked password databases shown in detail in Table I. All these databases are publicly available and selected according to the following two principles: 1) they should represent a significantly large user population (over 1 million passwords for each) and 2) they should have information about password frequencies to allow richer analysis.

TABLE I: The 17 breached databases used in our work.

No.	Database	Dominating Users	Service	Size	Year
1	CSDN	Chinese	Pro.	6,387,785	2011
2	Tianya	Chinese	Soc.	30,274,001	2011
3	7K7K	Chinese	Ent.	8,460,641	2011
4	17173	Chinese	Ent.	17,942,621	2011
5	178	Chinese	Ent.	9,072,688	2011
6	Dodonew	Chinese	Pro.	14,122,756	2011
7	Twitter	English	Soc.	67,095,263	2016
8	Webhost	English	Pro.	14,436,531	2015
9	RockYou	English	Soc.	28,705,927	2009
10	MyHeritage	English	Life	84,825,745	2017
11	Gmail	English	Life	4,663,677	2014
12	8Fit	Germany	Life	1,121,536	2018
13	Eyeem	Germany	Pro.	4,043,116	2018
14	Ge_Mix1	Germany	Mix	6,761,255	2018
15	Fr_Mix1	French	Mix	1,302,365	2018
16	Fr_Mix2	French	Mix	1,098,418	2018
17	Fr_Mix3	French	Mix	10,284,538	2018

A. Conceptual Model of SE#PCFG

1) Four Structural Levels: First, we define four password structural levels to better guide analysis of password semantics.

1) Characters: At this level, each character bears the lowest-level information about a password.

2) Semantic Factors (**word**-level semantics): This level is about a number of consecutive characters (i.e., a word) that together form a semantically meaningful unit, which we call a semantic factor. To indicate what semantic information a semantic factor carries, we call it a **semantic factor type**. For the sake of brevity, in the following, we use "SF" and "SFT" to refer to "semantic factor" and "semantic factor type" respectively. Furthermore, we denote a tuple (SF, SFT) to make it clear what SFT one SF belongs to.

3) Semantic Patterns (**password**-level semantics): This level looks at how the whole password is semantically composed of one or more semantic factor types. In the rest of this paper, we use an ordered list of SFTs to denote a password's semantic pattern, e.g., [EN_NOUN, NUMBER3] is the semantic pattern of the password "`king123`", and "SP" to refer to "semantic pattern".

4) Semantic Structure (**population**- or **database**-level semantics): This level is about the overall observable semantic structure of passwords generated by a group of users, reflecting their collective behaviors that could map to one or more shared semantic attributes (e.g., language spoken, age, gender, and website type). For our work, we considered language and website type because they are more available with leaked password databases than other attributes.

Based on the four-level password structure, we can more clearly see how our work differs from others. Specifically, [18], [17] work more at the first level to build character-to-character transition probabilities without considering any real semantic information. [9], [10], [12], [33], [34] explore semantic information at the second level with limit SFTs. In contrast, our work provides a more general way to cover a wide range of semantic information, which can also be tailored for specific password databases. An important contribution of our work is the significant expansion of SFTs covered at the second level to enable much more semantically aware password analysis, as explained in greater detail in the following.

2) SFTs and SFs: Understanding the semantic information people use when setting their passwords has never been an easy task because everyone incorporates their life experiences into the password-setting process, resulting in diversity in the semantic components of passwords. In past studies, researches always conducted their analysis through the following steps: manual observation $\rightarrow$ classification $\rightarrow$ semantic categorisation $\rightarrow$ advanced semantic analysis. Unlike past studies, we conducted a systematic search of different types of semantic components considered in previous work and also those mentioned in recommendations for password composition available on the Internet. For the second part, we used "How to create strong passwords" as a search query on the Google search engine, and selected the top 10 relevant returned results to identify relevant password composition recommendations. A detailed summary of our results on what we obtained from the 10 websites and what some selected previous studies considered can be found in Table II. As can be seen from the table, our work has considered the most comprehensive set of semantic factor types. Note that more SFTs can be easily added by password analysts thanks to the general structure of SE#PCFG.

Newly added SFTs: We introduce 14 new SFTs according to some observed gaps (e.g., what were acknowledged in [9]): 1) 5 SFTs for German words and 5 for French words; 2) Chinese acronyms; 3) WKNE and UBE to cover proper nouns and slangs; 4) CONSONANT to cover consecutive consonants.

In addition, we also label any other unknown SFTs as NN. To the best of our knowledge, the 43 SFTs form the richest set of password semantic information considered so far, and serve as a good base line for our implementation and experiments[1]. Table III gives more details about the definitions of these SFTs and sources we used.

[1]We plan to publish our corpus as soon as our work is accepted for publication.

B. A Streamlined Computational Process

Based on the conceptual model, we propose a following streamlined computational process of SE#PCFG to automate password semantic analysis in a more general way which consists of three steps: pre-processing, identifying SFTs in segments and post-processing.

We explain each step with greater details below. Table IV gives five typical examples of how each step works.

1) Step 1 – Pre-processing: Almost all NLP tools consider the change of character type (letter, digit, symbol) as a "split position" of consecutive words in a given text. This means that they cannot identify SFs with mixed character types such as "`1qaz`" (a keyboard pattern) and "`google.com`" (a domain name). Therefore, such SFs have to be identified before NLP tools are applied in the next step. Three SFTs we consider here are borrowed from Weir et al.'s implementation and several previous work [15], [11], [10]: keyboard patterns with n characters (KBn, where $n \geq 4$), domain names (DN) and email addresses (EMAIL). In addition, we also considered three other SFTs with mixed character types: prefixes (PRE), suffixes (SUF) and repeated strings (SR). We defined the above SFTs in relatively simple manner. Others are free to define more complex versions as needed.

For a given password, the pre-processing step tries to search for all possible SFs falling into the six SFTs following a predefined precedence order (KBn > EMAIL > DN > SRn > PRE = SUF). This order is designed following the implementation of original PCFG [15], and adding the three new SFTs in the end for those will not make any ambiguities. After all SFs are labeled, any remaining parts of the password are split into L (Letter), D (Digit) and S (Symbol) segments following the mechanism proposed by Weir et al.'s work [16] for further processing. The first row of Table IV shows how the pre-processing step works for a given password: `qwertpassword` $\rightarrow$ [(`qwert`, KB5), (`password`, L)], where the KB5 indicates the identified SFT of `qwert`. The second row illustrates the identification result of the password "`qazqazqaz`". The remaining parts containing L, D, S segments are for further processing.

2) Step 2a – Identifying SFs in L-Segments: After pre-processing, the remaining L-segments can be seen as a combination of multiple SFs (e.g., "`wonderbread`"), which are highly language-dependent, therefore NLP tools are needed. In this step, we discuss how SE#PCFG leverage a corpus to obtain richer semantics from L-Segments.

In our implementation of SE#PCFG, we followed Veras et al.'s work [9] to choose the widely used NLP library NLTK (https://www.nltk.org/) to identify linguistic SFs and implement a scoring system based on source and reference corpora and n-gram frequencies to disambiguate the results of segmentation. The whole process can be split into two sub-steps: i) further segmenting each input L-segment into smaller linguistic elements (e.g., "`sunnyboy`" into "`sunny boy`") and tagging them, and ii) identifying SFs more than English words.

For sub-step i), we first use several corpora with richer semantics to help NLTK identify SFs. First, we chose to use two widely used English corpora "Brown" and "Web Text"

TABLE II: Semantic factor types claimed to be dangerous and how they are considered by previous work and ours.

Source	Word	PI[a]						SI[a]			Tricks			Twin[b]
		Name	Mobile	Birthday	Address	UN[b]	Email	PN_1[b]	ON[b]	PN_2[b]	Sequence	Keyboard	Sub.[b]	
Microsoft	✓	✓	✗	✗	✗	✗	✗	✓	✓	✗	✗	✗	✗	✓
Norton	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓	✗	✗	✗	✓
GCFglobal	✓	✓	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗	✓
UC Santa	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
Google	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓	✓	✗	✓
CMU	✓	✓	✓	✓	✓	✗	✗	✓	✗	✗	✗	✗	✓	✗
AVAST	✓	✓	✗	✓	✓	✓	✗	✗	✗	✗	✓	✗	✓	✗
CISA	✓	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗
WebRoot	✓	✓	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗
Harvard	✓	✓	✓	✓	✗	✗	✓	✗	✗	✗	✓	✓	✗	✗
Sum	10	9	5	8	5	2	2	3	1	1	3	2	3	4
[9][c]	◐[d]	◐	○	○	●	-	○	◐	◐	◐	○	○	○	-
[10][c]	◐	◐	○	●	○	-	○	○	○	○	●	●	○	-
[12][c]	◐	●	●	●	●	-	○	○	○	○	●	○	○	-
Ours	●	●	●	●	●	-	●	●	●	●	●	●	●	-

[a] PI and SI are short for "Personal Information" and "Social Information" respectively.
[b] UN, PN_1, ON, PN_2 are short for "User Name", "Product Name", "Organization Name" and "Proper Noun" respectively. "Twin" means passwords used in different websites but from the same user. "Sub." is short for "Substitution". Note that password analysis generally clear breached user information to protect privacy, our work exclude "User Name" just as other work did.
[c] We select three past studies mostly related on studying password semantics, and compare them with our work on these mentioned semantic factor types.
[d] ○, ◐, ● mean not, partially and fully considered in each work respectively. As pointed out by [12], [9] and [10] left Chinese Pinyin names unexplored. In terms of words in different languages, the previous three works either focused on English words or added Chinese Pinyin, without considering other languages such as German or French. In addition, the authors of [9] advised to optimize their work by supplementing the corpus containing new terms (e.g. company names, slang or proper nouns) which not appeared in their source corpus.

TABLE III: 43 SFTs used in our implementation of SE#PCFG.

SFT	Description[a]	SFT	Description
EMAIL [11]	Email addresses	DN [11]	Domain namess
PY [10]	Pinyin strings of all Chinese character	**CONSONANTS**	Two or more consecutive consonants can cover many acronyms
SR4, SR5, ... [33]	Kinds of Combination of small strings	YEAR [11]	4-digit years between 1990 and 2100
PRE1, SUF1, ... [33]	prefixes and suffixes	YYMMDD, ... [31]	6- and 8-digit dates in different formats
KB4, KB5, ... [10]	Keyboard patterns with $4, 5, \ldots$ characters	CN_MOBILE [31]	11-digit mobile numbers (used in China)
EN_ [9]	11 POS tags of English: NOUN, VERB, PRON, ADJ, ADV, ADP, CONJ, DET, PRT, X, NUM	**GE_, FR_**	5 most common POS tags in German (GE_) and French (FR_): NOUN, ADJ, ADV, PRON, VERB
NUMBER1, ... [16]	Numbers with $1, 2, \ldots$ digits	SPEC1, ... [16]	Consecutive special characters
LOCATION [12]	English names of places[b]	**WKNE**	Wiki name entity [40]
MONTH [9]	English words for 12 months	**UBE**	Urban Dictionary entity [41]
NAME [12]	Male and female names[c]	LEET [33]	Leet rules described in III-B4
CN_NAME_ABBR	Acronyms of Chinese names[d]	NN	Unknown semantics

[a] 14 newly added SFTs are highlighted in bold, while others were introduced in previous work.
[b] Extracted from the world (non-Chinese) location databases in the Chinese instant messaging software Tencent QQ and the Geonames [42] list of cities.
[c] Extracted from a database released by the US Social Security Administration (SSA) [43], based on a 100% sample of records of Social Security card applications as of March 2019. The database contains information on gender.
[d] 3- and 4-letter only; derived from Chinese names in [44].

TABLE IV: Five typical passwords to show details of each step in the computational process of SE#PCFG. "—" means the output of the former step will stay the same after this step.

Password	Step 1	Step 2a	Step 2b	Step 3	Result
qwertpassword	[(qwert, KB5), (password, L)]	[(qwert, KB5), (password, EN_NOUN)]	—	—	[(qwert, KB5), (password, EN_NOUN)]
qazqazqaz	[(qazqazqaz, SR9)]	—	—	—	[(qazqazqaz, SR9)]
zhangfei1990	[(zhangfei, L), (1990, D)]	[(zhang, PY), (fei, PY), (1990, D)]	[(zhang, PY), (fei, PY), (1990, YEAR)]	—	[(zhang, PY), (fei, PY), (1990, YEAR)]
Pa$$word	[(Pa, L), ($$, SPEC2), (word, L)]	—	—	[(Pa$$word, LEET)]	[(Pa$$word, LEET)]
ahnung	[(ahnung, L)]	[(ahnung, GE_NOUN)]	—	—	[(ahnung, GE_NOUN)]

to cover English words. Then we intersected the German dictionary with word frequencies in WorldLex [45] and Wiktionary of German [46] to get more commonly used German words. The same was done with the French dictionary in WordLex [45] and Fewiktionary [47] to produce a French corpus. To cover slangs and proper nouns/phrases, Wikipedia (WKNE) and Urban Dictionary (UBE) were used to produce two more corpora by concatenating entries they cover. Besides, yet another corpus was produced using a number of ad hoc dictionaries to cover other proposed SFTs such as LOCATION.

After segmentation is done, NLTK's POS module is used to directly identify SFs belonging to SFTs with a clear linguistic meanings in English displayed in Table III, which start with EN_. To recognize non-English words without relying on a POS tagger, we chose to inject non-English words into the English POS tagging process as dummy NP words, which can be mapped to the following SFTs via simple string matching: German and French SFTs, LOCATION, MONTH, MALE_NAME and FEMALE_NAME, etc. For any unrecognized segments, we labeled them as NN. Rows 1, 3 and 5 of Table IV show the results after this step.

3) Step 2b – identifying SFs in D- and S-Segments: Step 2a identifies SFs in L-segments, so other SFs are processed in this step using non-NLP methods. For S-segments (i.e., those with special characters only), we treat them as a single SFT SPECn ($n = 1, 2, \ldots$). For D-segments (i.e., numbers), they are processed in two further sub-steps. First, if the length is 4, 6, 8 or 11, the segment will be checked against one of the four SFTs: i) 4-digit years (YEAR), ii) 6-digit dates in the format of YYMMDD (Chinese style), MMDDYY (American style) and DDMMYY (European style), iii) 8-digit dates in the format of YYYYMMDD, MMDDYYYY, DDMMYYYY, iv) and 11-digit for mobile phone numbers in China (CN_MOBILE). Then, if none of the above SFTs are matched, the number is labeled as NUMBERn ($n = 1, 2, \ldots$). Row 3 of Table IV shows the result after this step.

4) Step 3 – Post-processing: After previous steps, a password will be split into multiple sequential SFs. However, for passwords that went through leet transformations, we will end up with a larger number of wrong SFs, e.g., "`pa$$word`" will lead to three SFs – "`pa`", "`$$`", "`word`". To fix such problems, we introduce a post-processing step to further process NN-SFs and passwords with too many (> 3 for our implementation) SFs. According to [33], the top ten transformations (0 ↔ o, 1 ↔ i, 3 ↔ e, 4 ↔ a, 1 ↔ !, 1 ↔ l, 5 ↔ s, @ ↔ a, 9 ↔ 6, $ ↔ s) can cover 96.6% leet pairs, so we decided to consider these leet transformations only. Once detected, we label the whole leet-transformed SFs as a single SF of type LEET. Note that the main purpose of this step is to refine segmentation results of previous steps, so more optimizations could be applied. Row 4 of Table IV gives a visual example.

C. Experimental Results

Now we report selected results of applying our implementation of SE#PCFG to study password semantics of the 17 leaked password databases listed in Table I.

Attributes of databases: As mentioned before, the 17 databases were selected to cover two main semantic attributes of online services and their users: language (English, Chinese, German, and French), and service type (Social Networks, Entertainment, Profession, and Life). We noticed that users of each database can be from any country all over the world, but we do not have enough information to determine their nationalities and preferences of speaking language(s). So we categorized databases just based on the dominating users the website served. Note that for English databases of large websites, there are likely many users from non-English-speaking countries, so "English" should be treated as "dominated by English".

TABLE V: Segmentation results over all the 17 databases aligned by language.

CN	SR (%)[a]	EN	SR (%)	GE	SR (%)	FR	SR (%)
1	95.31	7	90.26	12	94.84	15	89.72
2	95.71	8	89.00	13	95.04	16	89.46
3	94.80	9	93.33	14	88.80	17	89.29
4	96.88	10	85.11				
5	96.64	11	90.99				
6	97.14						

[a] "SR" is short for "Success Rate", which means the percentage of all segmentation results that not contain the SFT of NN.

Data cleaning: As with [20], [12], [23], we cleaned the databases by removing passwords containing symbols beyond 95 printable ASCII characters or longer than 30 characters. We believe this strategy is reasonable as all these websites only take the 95 printable ASCII characters as legal components of their users' passwords.

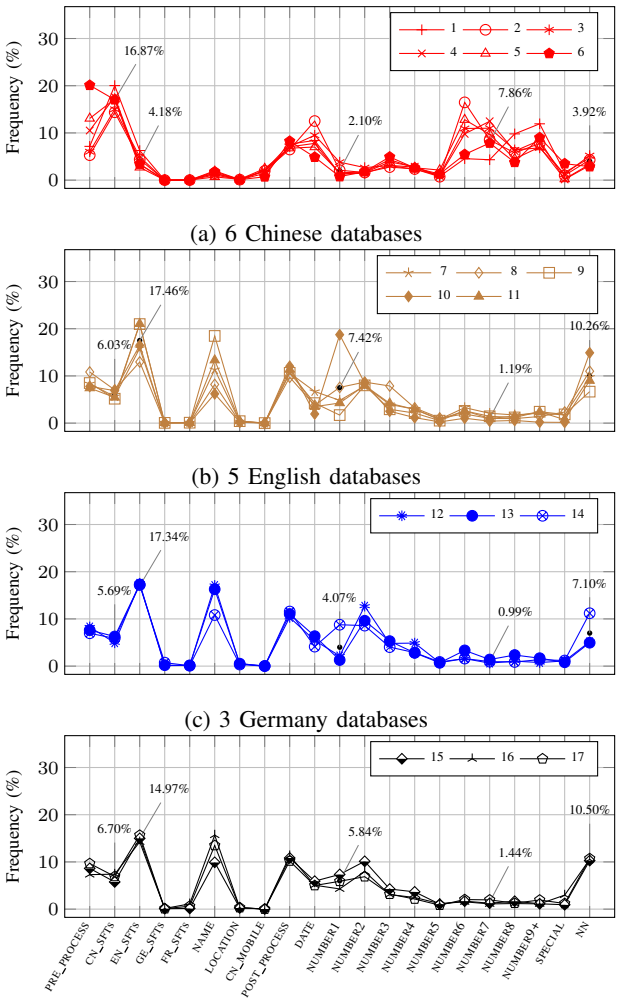
Segmentation results: NN can be seen as a good indicator of how well the framework worked. The less NN remain, the more meaningful SFTs are identified. Our experimental results in Table V showed that our implementation of SE#PCFG can identify 85.11% and 97.14% passwords across all 17 databases. Based on these learned semantic information, we report some selected observations at three semantic levels (SFs/SFTs, SPs, and semantic structures) below.

1) Analysis of SFs and SFTs: Past studies have shown frequent use of some SFTs in user-generated passwords, such as numbers, names, dates, and different linguistic elements [8], [9], [11], [12], [28], but a systematic look at a more diverse set of SFTs (e.g., the 14 new ones in SE#PCFG) and SFs is still lacking. To make it easier to identify useful patterns, we re-grouped all the SFTs into 21 groups with a closer semantic relationship: all special characters-based SFs into one (SPECIAL), all name-related SFs into one (NAME), all date-related SFs into one (DATE), all numeric SFs with at least 9 digits into one (NUMBER9+), all SFs for a specific language into one (EN_SFTs, GE_SFTs, and FR_SFTs), SFs identified during pre-processing and post-processing into PRE_PROCESSING and POST_PROCESSING, respectively.

The results led to a number of interesting observations not reported before. **Preference of languages**: 1) In all databases, Chinese-related SFTs are popular (16.87%, 1st in Chinese databases, 6.03%, 7th in English databases, 5.69%, 6th in German databases and 6.70%, 6th in French databases), which may be explained that non-Chinese databases are all multinational so they have a significant number of Chinese-speaking

users. 2) For all non-Chinese databases, English SFs as a collective SFT group is either the highest or the second highest. This can be explained by the fact that English is the "world" language used widely in all countries. 3) To our surprise, users of German and French databases seemed to prefer English over their native language. Although they have the highest ratio by their own language-related SFTs, but the absolute number is much lower than English-related or even Chinese-related SFTs. This may be explained by non-English-speaking users feeling that using English passwords is more convenient, but more empirical studies involving recruited human participants are needed to understand such a phenomenon more.

Numeric SFs: Past studies [10], [12], [31] have showed the use of numeric segments in user-generated passwords. The richer SFTs used in SE#PCFG still allowed us to observe an interesting new finding: **Chinese and non-Chinese users had different behaviors** – Chinese users tended to use longer numeric SFs (with 6-8 digits) than non-Chinese users (with just 1-3 digits).



(a) 6 Chinese databases

(b) 5 English databases

(c) 3 Germany databases

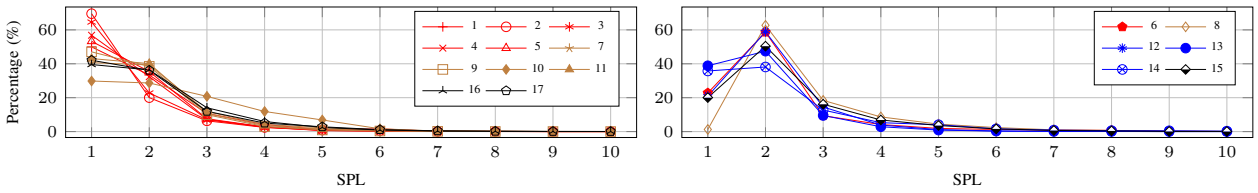
(d) 3 French databases

Fig. 1: Distribution of combined SFTs in the 17 databases. We can see a clear vision that English, German and French databases have similar distribution at SFT-level except for 10 (MyHeritage). Meanwhile, Chinese databases have similar distribution with each other, but quite different from the other databases. All numbers labeled in each figure are on average.

Attributions of databases: No noticeable patterns were observed related to the service type, implying that it may not be a good indicator for analyzing user-generated passwords. In contrast, we can see language plays a key role in the semantic structures at the population/database level: databases sharing the same language have a similar semantic structure, but those labelled with different languages have very different semantic structures. This is a new piece of evidence about users speaking different languages have different password composition behaviors.

New SFTs introduced in SE#PCFG: We had interesting observations about the 14 new SFTs described in Section III-A2. 1) They play an important role in segmentation results. Averagely 10.38%, 13.11%, 12.80% and 13.47% passwords consist of these SFTs in Chinese, English, German and French databases, respectively. Out of all these SFTs, WKNE is in the majority in all databases, which indicates that this SFT works well in enriching our understanding of password semantics. 2) Some past studies [12], [28] reported that in Chinese and English databases, SFs like "`love`" or "`ai`" (the same meaning in Chinese) or "`520x`" (a homophonic number of "`I love you`" in Chinese) appeared very frequently. We observed a similar pattern in German and French databases: "`Ich liebe dich`" and "`Je t'aime`" mean "I love you" in German and French, respectively, and they appear from 73–1,329, 64–5,416 times in the language-aligned databases, respectively. On the other hand, we also noticed frequent use of dirty words. For instance, the English phrase "`fuckyou`" appears between 2,110 and 25,357 times in the four English databases. A similar pattern was also observed in Chinese, German and French databases. 3) Some special types of proper nouns/phrases including names of celebrities, large companies/brands and popular games seem much more popular among non-Chinese users than among Chinese users, e.g., "`samsung`" (non-Chinese 0.19% vs Chinese 0.05%) and "`pokemon`" (non-Chinese 0.17% vs Chinese 0.01%).

2) Analysis of SPs: **SP length**: For a password, we define the length of SP (SPL) as the number of SFTs included in the SP representing the password. SPL can reflect how complicated a user's mental model was when they generated a password. Figure 2 shows the results about SPL. In 11 databases (5 Chinese, 4 English and 2 French databases), the majority of SPL is 1, while passwords having SPL of 2 dominate in the other 6 databases. 91.4% of all passwords have just one to three SFs (39.77% for SPL = 1, 39.95% for SPL = 2 and 11.69 for SPL = 3), and almost all (98.3%) passwords have an SPL no more than five. These results suggest that most users had a relatively simple mental model for generating passwords, which matches the well-reported preference of users for usability over security [48]. Another interesting observation is that the average SPL of all six Chinese databases is 1.702, significantly smaller than that of English (2.136), German (2.037) and French (2.117) databases. Such differences reflect different collective behaviors of Chinese and non-Chinese users.



(a) Databases with peak SPL=1 (b) Databases with peak SPL=2

Fig. 2: Distributions of SPL in the 17 databases

3) Cross-Database Semantic Correlations: **Metric to evaluate password semantics at the database level**: The semantic structure of one database can be represented by a discrete probability density (DPD) of each unique SF, SFT or SP. One useful metric capturing similarities and differences of user behaviors is cosine similarity because it is one of the mostly widely used metrics for such purposes [49]. For the three levels, the one at the SFT level will be more robust and easier to calculate because the dimensionality of the DPD at the SFT level is much smaller than that at the other two levels.

It is also possible to define a correlation metric across two or more semantic levels to make the indicator more informative. For instance, assuming that A and B represent the performance forms of a specific SFT on two databases. A_i, B_i mean the i-th SF in A and B, then we can use the similarity metrics at the SF level for each SFT to adjust the similarity metrics at the SFT level so that the new metric covers both:

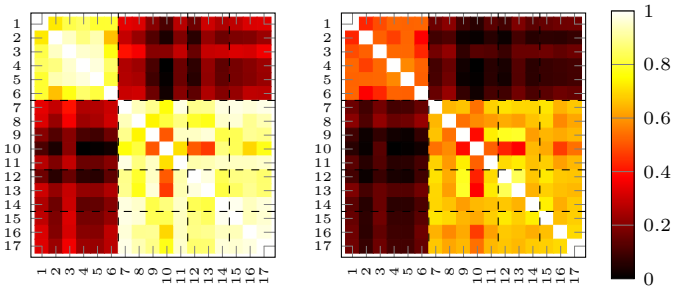
$$\mathrm{Sim}_{\mathbf{A},\mathbf{B}}^{\text{SF-SFT}} = \frac{\sum_{i=1}^{n}(w_{A,B,i}A_iB_i)}{\sqrt{\sum_{i=1}^{n}A_i^2}\sqrt{\sum_{i=1}^{n}B_i^2}}, \tag{1}$$

where $w_{A,B,i}$ is a similarity metric of the SF-level DPDs of the two databases for the i-th SFT, with a range of [0,1], and the base-line DPDs are at the SFT level. Similarly, many correlation metrics can be used to calculate $w_{A,B,i}$. In our experiments, we used a simple metric focusing on the average probability of common SFs shared between two databases for a given SFT:

$$w_{A,B,i} = \sum_{\mathrm{SF}_j \in \mathrm{SFs}_A \cap \mathrm{SFs}_B} \mathrm{Prob}(\overline{\mathrm{SF}_j}), \tag{2}$$

where SFs_A and SFs_B are the sets of all SFs belonging to the i-th SFT in Database A and B, respectively, and $\overline{\mathrm{SF}_j}$ is the average occurrence probability of SF_j in the two databases.

Cross-database semantic correlations at the SFT and SF levels: Following the equations above, we can calculate the overall semantic correlation between any two given databases at different semantic levels. Figure 3 shows the cross-database semantic correlation values between each pair of the 17 databases as a diagonally symmetric heatmap, using [49] and Eqs. (1), respectively. The dashed lines in the heatmaps separate Chinese (1-6), English (7-11), German (12-14) and French (15-17) databases to show the language-dependent patterns more clearly. From the two heatmaps, we can see a number of visual patterns. First, there are two clearly non-overlapping areas – one for Chinese databases, and the other for non-Chinese databases, indicating that Chinese and non-Chinese users have very different collective behaviors. One possible reason of this pattern is that Chinese websites are more dominated by Chinese-speaking users, but Western



(a) SFT level (b) SF-SFT level

Fig. 3: Cross-database semantic correlation values at the SFT level and those at the combined SF-SFT level, according to [49] and Eq. (1), respectively. The x- and y-axis show the indices of the 17 databases shown in Table I.

websites have a more mixed groups of users who speak different languages. Another reason is that Western languages are linguistically more similar to each other than Chinese to any Western languages. Second, users of Myheritage (10) display very different habits from all the other databases. This phenomenon is echoed later by the poorer password cracking performance against Myheritage using other databases as the training database (see Sections IV-C). Finally but equally interestingly, comparing the two heatmaps, the correlation values between Chinese databases drop significantly when SFs are considered to weigh the SFT-level correlations, suggesting that Chinese users share more common behaviors on the selection of SFTs but they behave less similarly on selections of SFs. This phenomenon is much less obvious for non-Chinese databases, suggesting that Western users are more consistent in choosing both SFs and SFTs.

IV. Semantically Enhanced Password Cracking

Thanks to the enhanced semantic awareness, SE#PCFG clearly has the potential to be used for designing more powerful PCFG-based password cracking methods. Combining SE#PCFG with a systematic model smoothing method, we developed Semantically Enhanced Password Cracking Architecture (SEPCA), a new password cracking architecture that was shown to be able to outperform mainstream SOTA password cracking methods under the scene of real-attacking. The main idea of model smoothing is to address SFs that are not present in training sets but appear in the targets. This problem was first mentioned in [16]. Surprisingly, very few researchers have studied how to practically and systematically smooth a

password model, which so far is mainly done by injecting extra information such as new dictionaries with a fixed but ad hoc coefficient.In the following, we first explain the design and implementation of SEPCA, as one of the technical contribution of our work, which allowing us apply a natural way to assign a set of non-zero probabilities to unobserved SFs, then discuss how we conducted our experiments, and finally present the results comparing with three state-of-the-art methods and new insights learned.

A. The Proposed New Architecture SEPCA

A generic probabilistic context-free grammar *G* can be defined by a quintuple, *G*=(*M, T, R, S, P*), where *M* and *T* represent the set of non-terminal and terminal symbols respectively. *S* is the start symbol belonging to *M*. *R* is the set of production rules and *P* contains the probabilities of each rule in *R*.

In SEPCA, *T* is the set of all SFs, *M* is the union of *T* and *S*. The production rules can be categorized into two groups: 1) from *S* to a certain SP, following $\sum_k P(S \rightarrow \text{SP}_k) = 1$. 2) from a SFT to a certain SF, and $\forall i, \sum_j P(\text{SFT}_i \rightarrow \text{SF}_j) = 1$. Under the above grammar, we can calculate the probability of any given password as follows to allow ranking passwords for cracking purposes:

$$P(\text{password}) = P(S \rightarrow \text{SP}_k) \prod_{\text{i, j}} P(\text{SFT}_i \rightarrow \text{SF}_j). \quad (3)$$

Researchers have proposed different ways to assign probabilities to *T*. In [18], probabilities of L-segments are calculated by another Markov model over a natural language, while D- and S-segments share the same probability. In [16], probabilities of D- and S-segments are calculated based on the training set, while L-segments in a given dictionary are assigned the same probability. Different from the above approaches, we design a more general way to deal with SFs not present in the training set. First, we further split SFs into two sub-sets, observed SFs (marked as OSFs, T_{ob}) and unobserved SFs (marked as USFs, T_{uob}). Under these definitions, the probabilities of these two sets are marked as P_{OSFs} and P_{USFs}, respectively. Then we have the following equation:

$$\forall i, P_{\text{OSFs}} + P_{\text{USFs}} = 1, \text{OSFs}, \text{USFs} \in \text{SFT}_i. \quad (4)$$

Our smoothing method tries to assign more meaningful probabilities to USFs. In our experiments, we split the training set into two parts according to the size ratio of the training and target databases, then calculate $w_{A,B,i}$ for every SFT_i following Eq. (2), and set the estimated probabilities of all OSFs and all USFs under SFT_i as $P_{\text{OSFs}} = w_{A,B,i}$ and $P_{\text{USFs}} = 1 - w_{A,B,i}$. Finally, for each individual SF, we do the following:

- For an individual OSF $\in \text{SFT}_i$, its probability is calculated based on its original probability in the training set weighted by $w_{A,B,i}$, i.e., $P(\text{OSF}) = w_{A,B,i} \times P(\text{OSF}|\text{SFT}_i)$.
- For an individual USF $\in \text{SFT}_i$, we assume that each USF appears equally, so $P(\text{USF}) = \frac{1-w_{A,B,i}}{\#(\text{USFs})}$, where $\#(\text{USFs})$ is the number of all USFs in SFT_i.

The smoothing method can be easily generalized to handle more complicated cases, e.g., USFs of a specific SFT and different USFs of the same SFT are handled differently from others. The smoothing method on USFs can in principle be generalized to unobserved SPs, too. These will be left as our future work.

B. Experiment Setups

Performance metrics: To compare the performance of password cracking methods, we need some quantitative metrics. One effective metric widely used in the literature is the "coverage rate" $R(\text{D}, n) = N_c(\text{D}, n)/N(\text{D}) \in [0, 1]$, where $N(D)$ is the total number of passwords in the target (test) database D and $N_c(D, n)$ is the number of successfully cracked passwords in D with n guesses. In fact, this metric can also be split into two different types:

a) $R_{\text{po}}(\text{D}, n)$. If D has duplicate passwords or password frequencies, this metric can be seen as working at the user-level. The higher $R_{\text{po}}(\text{D}, n)$ is, the more users' passwords are cracked.
b) $R_{\text{pa}}(\text{D}, n)$. If D has neither duplicate passwords nor password frequencies, this metric works at the password-level. As reported in [35], [22], this metric is a good indicator to demonstrate a password cracking method's ability to generating new (or unseen) passwords.

There are two main methods for calculating coverage rates: 1) running a real password cracking process to enumerate passwords and calculate the actual coverage rate, i.e., via a simulated "real-attacking", and 2) using a stochastic process like the Monte-Carlo algorithm proposed in [50] to approximately estimate the coverage rate. The "real-attacking" method can give more accurate results, but can be computationally prohibitive if the number of guessed passwords n becomes too large (e.g., above 10^{12}). Therefore, when this method is used, it is common to use a practically large but computationally achievable value of n, e.g., $n = 10^7$ [12], [9] and $n = 10^{10}$ [22]. The Monte-Carlo method can work only with password cracking algorithms based on a clearly defined probability model, but can be used to estimate the coverage rate of a very large n with a much smaller number of randomly sampled passwords (e.g., 10^6 random passwords to estimate the coverage rate with n as large as 10^{16}) [34], [17], [35]. We chose to use "real-attacking" for all our experiments for the following reasons: 1) We hoped to compare our work with as many different models as possible, but [22] was clearly claimed that it was not suitable for Monte-Carlo estimation. 2) In [50], [17], it was mentioned that the exact error rates of the Monte-Carlo estimation method depend heavily on the attack methods, so we conducted a small experiment to see whether we could use Monte-Carlo estimation for SEPCA. Our results in Figure 4 showed that the coverage rates calculated from real-attacking and Monte-Carlo experiments can have a gap as high as 17.79% for SEPCA, which we considered too high for a fair and reliable comparison with other SOTA methods. Therefore, to better understand how SEPCA performs, we chose to use "real-attacking" metrics for all our experiments.

The SOTA benchmarks: To investigate how SEPCA's performance compares against other mainstream SOTA password cracking methods, we used the latest implementation of [16],

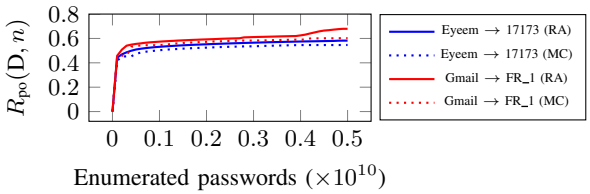


Fig. 4: Performance using Monte-Carlo (MC) estimation and real-attacks (RA).

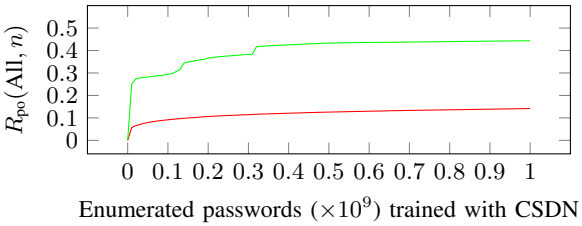


Fig. 5: Performance comparison between SEPCA and DPG over all testing sets. SEPCA, DPG [22].

i.e., PCFG ver. 4.3 [15] (denoted by "PCFG$_w$"), Veras et al.'s Semantic PCFG [9], [51] (denoted by "PCFG$_{\text{Se}}$"), and Melicher et al.'s neural network [17], [52] (denoted by "FLA"), as the benchmarks. We noticed that there are also some other password modeling methods, such as CPG and DPG [22], PassGAN [21], RFGuess [23], PassBERT [24], PassTSL [25], OMEN [19], and one based on an n-gram Markov model [20]. However, based on the results of [22], [17], we observed that FLA can always outperform OMEN and 6-gram Markov models. As to RFGuess and PassTSL, we noticed that they evaluated their performance by Monte Carlo estimation. We excluded PassBERT and CPG model as they both require a bunch of preset templates as additional input. The DPG model can be considered an enhanced version of PassGAN, so we conducted some experiments to see how DPG performs without the feedback of target sets using the open-sourced code. We trained DPG using the training sets described in Section IV-B and generated $\times 10^9$ passwords in about 3 days. Our results over all testing sets showed that, without the feedback of the testing set, e.g., $\alpha = 0$, DPG can outperform PassGAN, but has a much lower performance compared with SEPCA (see Fig. 5). Considering the above experimental results, as well as the fact that DPG can only obtain character-level semantic information in reality, we finally decided to not consider CPG, DPG, PassGAN, RFGuess, PassBERT, PassTSL or OMEN as part of our benchmarks.

Training and test sets: For our experiments, we used CSDN, Gmail, Eyeem, Fr_Mix1 as training sets, for they have similar sizes and one for each of the four languages studied. The other 13 databases are treated as testing sets, and it ends up to $4 \times 13 = 52$ test cases. More precisely, we used the output of SE#PCFG dealing with each of the four databases as SEPCA's input, then enumerated a set of passwords to attack each of the other 13 databases under the "real-attacking" scene. All these 17 databases have duplicate passwords shared by different users, so that we can investigate the attack performance at two levels: user-level (having duplicated passwords)and password-level (having unique passwords). All the three benchmarking methods and SEPCA are training-based, and we used exactly the same training set to ensure the comparison is fair.

Parameter selections: For all the three benchmarks, we used their default configurations recommended by their authors/developers to generate passwords and calculate guess numbers. Note that the FLA implementation [52] does not provide a direct interface to generate a specified number of passwords, but can output passwords with their probabilities higher than a given threshold. Therefore, to align with the scale of guessed passwords that previous work used [12], [9], [22], we set a threshold of 10^{-12} for FLA, which led to maximum 5×10^9 guessed passwords for each training set. This number of guessed passwords is large enough to compare password cracking performance, and to make the computational costs of the experiments manageable in a few weeks[2].

C. Experimental Results

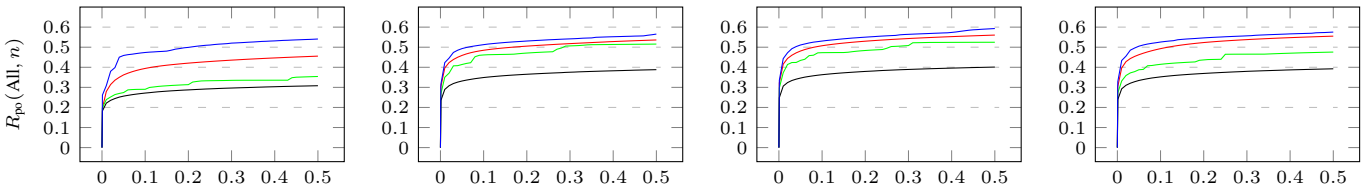
In this section, we report results of a series of experiments we conducted to show how much our SEPCA benefits from the richer semantic information enabled by SE#PCFG. We ran all experiments on a machine with an Intel Xeon E5-2640 CPU and two Nvidia Tesla M40 GPUs.

Performance Comparison at the User-Level: Figure 6 shows average results of all testing sets at the user-level. There are several clear observations as follows.

In terms of the average performance across 52 test cases, SEPCA performed significantly better than all the benchmarks: it outperformed PCFG$_w$ by 21.53%, PCFG$_{\text{Se}}$ by 52.55% and FLA by 7.86%. If we look at all the 52 test cases individually, the results are also overwhelmingly positive: SEPCA outperformed PCFG$_w$ and PCFG$_{\text{Se}}$ for all 52 cases, and FLA for all but one case (for the only one the performance drop is negligible at -0.3%). The only slight performance drop when compared with FLA happened when attacking MyHeritage. This exceptional case is not surprising: as mentioned in Section III-C3, users of MyHeritage tended to choose very unique SFTs and SFs, therefore the alignment between the training set and MyHeritage will be poorer. Detailed information are displayed in Table VI. These results indicate that SEPCA can be seen as the most practical and effective method for attacking a given database, as long as the number of guessed password is not prohibitively large (up to the level of 10^{10}).

Performance at (Unique) Password-Level: As shown in Table VII, SEPCA outperformed the benchmarks significantly on all 52 test cases: PCFG$_w$ by 43.83%, PCFG$_{\text{Se}}$ by 94.11%, and FLA by 11.16%. In terms of individual test cases, SEPCA performed the best in 50 out of all 52 cases (96.15%), except for using Gmail, Eyeem and Fr_Mix1 to attack MyHeritage. Again, as mentioned before, the poorer results on MyHeritage is not surprising given the database-correlation results in Section III-C3.

[2]According to the run-time performance results reported in Section IV-C, FLA is the least efficient password generating method. As reported in [22], FLA would need more than two weeks to generate 10^{10} passwords.



(a) Trained with CSDN ($\times 10^{10}$) (b) Trained with Gmail ($\times 10^{10}$) (c) Trained with Eyeem ($\times 10^{10}$) (d) Trained with Fr_Mix1 ($\times 10^{10}$)

Fig. 6: Performance comparison at the user-level between SEPCA and three SOTA password cracking methods over all testing sets on average using real-attacking. SEPCA, PCFG_w [15], PCFG_{Se} [51], FLA [52].

TABLE VI: Performance comparison between SEPCA and three state-of-the-art password cracking methods at user-level.

	Metrics[b]	2	3	4	5	6	7	8	9	10	12	14	16	17	Average
1	CR(S_1[a])	74.19%	74.06%	76.36%	74.06%	54.33%	35.37%	17.32%	40.27%	22.74%	31.32%	29.46%	30.06%	32.67%	45.56%
	CR(S_2)	57.10%	54.21%	55.42%	58.22%	39.73%	27.68%	16.58%	32.81%	18.33%	28.71%	23.56%	22.32%	25.74%	35.42%
	CR(S_3)	42.59%	40.88%	43.10%	47.49%	31.67%	23.79%	17.00%	34.11%	18.70%	30.92%	23.84%	22.52%	24.02%	30.82%
	CR(S_4)	**74.62%**[c]	**74.80%**	**76.78%**	**76.59%**	**55.84%**	**48.90%**	**23.43%**	**60.00%**	**33.08%**	**43.69%**	**42.53%**	**45.14%**	**47.35%**	**54.06%**
	RIR(S_1)	0.58%	1.00%	0.54%	3.42%	2.79%	38.25%	35.28%	48.99%	45.48%	39.48%	44.35%	50.16%	44.92%	27.33%
	RIR(S_2)	30.67%	37.97%	38.53%	31.56%	40.56%	76.68%	41.36%	82.87%	80.50%	52.15%	80.50%	102.2%	83.94%	59.96%
	RIR(S_3)	75.18%	82.99%	78.15%	61.29%	76.31%	105.5%	37.89%	75.90%	76.97%	41.30%	78.42%	100.4%	97.14%	75.96%
11	CR(S_1)	65.99%	62.86%	58.53%	58.98%	32.67%	56.72%	31.34%	70.18%	**41.36%**	54.79%	50.51%	55.55%	57.41%	53.61%
	CR(S_2)	63.58%	59.20%	55.31%	56.63%	29.99%	55.70%	30.74%	69.23%	40.93%	52.52%	47.93%	52.56%	55.94%	51.56%
	CR(S_3)	27.76%	25.74%	27.53%	32.88%	17.83%	45.91%	28.98%	63.04%	35.67%	58.70%	45.69%	46.95%	47.88%	38.81%
	CR(S_4)	**67.18%**	**64.96%**	**61.69%**	**61.96%**	**35.77%**	**59.91%**	**33.72%**	**73.52%**	41.23%	**61.99%**	**54.40%**	**58.22%**	**60.41%**	**56.54%**
	RIR(S_1)	1.82%	3.34%	5.39%	5.04%	9.47%	5.63%	7.60%	4.76%	-0.3%	13.14%	7.70%	4.81%	5.22%	5.66%
	RIR(S_2)	5.67%	9.72%	11.52%	9.42%	19.26%	7.55%	9.68%	6.20%	0.75%	18.03%	13.49%	10.77%	7.98%	10.00%
	RIR(S_3)	141.9%	152.3%	124.1%	88.46%	100.5%	30.48%	16.35%	16.62%	15.59%	5.60%	19.07%	24.01%	26.16%	58.56%
13	CR(S_1)	67.30%	65.08%	62.78%	63.02%	37.99%	57.37%	33.13%	70.02%	40.13%	64.59%	53.04%	56.73%	57.28%	56.04%
	CR(S_2)	60.66%	55.20%	52.73%	54.39%	30.93%	56.24%	33.09%	69.51%	39.81%	65.18%	52.06%	55.33%	56.86%	52.46%
	CR(S_3)	30.29%	28.52%	31.18%	35.31%	20.70%	45.31%	29.81%	62.89%	34.52%	60.34%	46.41%	48.13%	47.96%	40.10%
	CR(S_4)	**69.95%**	**68.65%**	**67.96%**	**67.85%**	**42.48%**	**60.92%**	**34.65%**	**74.47%**	**41.00%**	**66.97%**	**55.82%**	**58.84%**	**60.67%**	**59.25%**
	RIR(S_1)	3.94%	5.48%	8.27%	7.66%	11.80%	6.18%	4.58%	6.35%	2.17%	3.69%	5.25%	3.73%	5.92%	5.77%
	RIR(S_2)	15.32%	24.36%	28.88%	24.74%	37.32%	8.32%	4.73%	7.13%	3.00%	2.73%	7.23%	6.35%	6.69%	13.60%
	RIR(S_3)	130.9%	140.7%	117.9%	92.17%	105.2%	34.45%	16.23%	18.40%	18.77%	10.98%	20.30%	22.26%	26.51%	58.07%
15	CR(S_1)	65.34%	62.65%	59.46%	60.03%	36.01%	59.79%	33.61%	68.99%	40.86%	63.77%	54.03%	58.55%	58.48%	55.51%
	CR(S_2)	49.50%	43.58%	41.95%	45.51%	25.74%	54.58%	32.49%	64.38%	38.61%	61.29%	50.67%	55.00%	55.14%	47.57%
	CR(S_3)	27.41%	25.45%	27.57%	33.09%	18.50%	46.86%	29.67%	60.74%	36.76%	58.86%	46.84%	49.50%	48.97%	39.25%
	CR(S_4)	**66.33%**	**63.56%**	**60.76%**	**61.69%**	**36.92%**	**62.32%**	**35.20%**	**73.26%**	**41.65%**	**65.95%**	**56.65%**	**60.52%**	**61.81%**	**57.43%**
	RIR(S_1)	1.51%	1.45%	2.18%	2.77%	2.53%	4.24%	4.76%	6.18%	1.94%	3.42%	4.85%	3.37%	5.70%	3.45%
	RIR(S_2)	33.98%	45.85%	44.84%	35.55%	43.47%	14.18%	8.36%	13.78%	7.89%	7.59%	11.81%	10.04%	12.10%	22.26%
	RIR(S_3)	141.9%	149.7%	120.4%	86.46%	99.58%	33.00%	18.65%	20.61%	13.29%	12.04%	20.94%	22.28%	26.22%	58.86%

[a] Denotations of cracking methods: S_1 – FLA [52], S_2 – PCFG_w [15], S_3 – PCFG_{Se} [51], S_4 – SEPCA. All experiments are conducted across $4 \times 13 = 52$ different test cases (4 training databases in the first column and 13 target databases in Columns 3 to 15) at user-level.
[b] Performance metrics in Column 2: CR = Coverage Rate, RIR = Relative Improvement Rate defined as $\text{RIR}(x) = (S_4 - x)/x$.
[c] The items in bold indicate that the method has the best cracking performance on the corresponding target database.

TABLE VII: Comparison between SEPCA and three state-of-the-art methods on password-level over all targets.

Methods	**Training Sets**				**AIR (%)**[a]
	1	11	13	15	
FLA [52]	20182435	26690266	27731271	28383124	11.16%
PCFG_w [15]	12584740	26172639	23947390	21048687	43.83%
PCFG_{Se} [51]	9430673	16872607	16713065	18518927	94.11%
SEPCA	**25148844**[b]	**28404466**	**30711293**	**29198645**	-

[a] AIR = Average Improvement Rate. Improvement Rate (IR) is defined as IR(x) = (SEPCA-x)/x.
[b] The items in bold indicate that the method has the most unique passwords cracked over all target databases .

TABLE VIII: Average coverage rate on user-level aligned to the language.

Training Sets	1	11	13	15
CN	71.73%	58.31%	63.38%	57.85%
EN	41.36%	52.10%	52.76%	53.10%
GE	43.10%	58.19%	61.39%	61.30%
FR	46.24%	59.32%	59.76%	61.17%

Performance on different language settings: Table VIII shows a number of interesting observations, which also echo some visual patterns in Figure 3 in Section III-C. 1) For Chinese, German and French targets, SEPCA performed better when being trained using language-aligned settings. 2) For English targets, training using an English database does not always produce the best results (52.10% in Gmail attacking English databases, lower than 53.10% in Fr_Mix1), which indicates that English databases likely include users with more diverse backgrounds. Actually this phenomenon is not surprising since Fr_Mix1 has a higher correlation with English

databases than Gmail (Fr_Mix1, 93.29% vs. Gmail, 91.53% on SFT-level, while 74.93% vs. 72.18% on SF-level). 3) The performances of SEPCA are more robust compared to other benchmarks: no matter which database was used for training, SEPCA always have a similar performance to attack all test sets (CSDN: 54.06%, Gmail: 56.54%, Eyeem: 59.25%, Fr_Mix1: 57.43%), while PCFG$_w$ fluctuates in the range from 35.42% to 52.46%, FLA moves between 45.56% and 56.04%, and PCFG$_{\text{Se}}$ from 30.81% to 40.10%.

Run-time performance: Table IX shows the run-time performance of the password generation process of SEPCA and each of the three benchmarks. One can see that SEPCA is much faster (around 5.6 times) on password generation than FLA, although is slower than the other two benchmarks – around 4.4 times slower than PCFG$_w$ and around 2.6 times slower than PCFG$_{\text{Se}}$ (likely due to its utilization of richer semantics). Considering SEPCA outperformed other benchmarks in its password cracking performance, we consider the effectiveness-efficiency balance of SEPCA reasonable.

TABLE IX: The average speed of generating passwords (p/s = passwords per second).

SEPCA	PCFG$_w$ [15]	PCFG$_{\text{Se}}$ [51]	FLA [52]
32,258 p/s	140,880 p/s	82,595 p/s	5,787 p/s

V. Further Discussions

The enhanced semantic analysis power of SE#PCFG and the improved password cracking capabilities of SEPCA have many profound implications in real-world applications. In addition to providing researchers with new tools for studying password security and usability, end users of password systems can also benefit from our work, e.g., they have better insights on how to define stronger but still usable passwords, and cyber security professionals have more evidence on how to define password policies to enforce or nudge securer password creation behaviors.

To demonstrate how our experimental results can help inform end users about weak passwords, in Table X we list top 10 weakest (i.e., the easiest to crack) password semantic patterns (SPs) for each of the four subsets of password databases grouped by language, leading to in total 20 SPs representing different types of weak passwords. As can be seen from this table, users speaking different languages have different weak password behaviors, but there are also shared patterns such as the use of numbers, dates, names of different types, nouns, and Pinyin for Chinese names. The behavioral patterns have clear psychological reasons since people tend to use things they can remember to define passwords. Such behaviors can be changed by introducing stricter password policies and adopting more intelligent password checkers, and the methods and tools reported in this paper can be used to continuously monitor leaked passwords and to support pentesting exercises simulating password cracking activities of adversarial actors.

Based on observed weak passwords, we can derive tips that can help non-expert users to define stronger passwords. For instance, if we use the weak passwords shown in Table X as examples, we can provide the following suggestions to end users: 1) avoid using personally identifiable information (e.g., one's own or family members' names and birthdays), commonly used names and other nouns in different languages including Pinyin names in Chinese; 2) rather than using any semantic factors directly, transforming or obfuscating them using different methods to make them harder to guess; 3) constructing passwords using three or more different semantic factors to increase the semantic complexity; and 4) using random passwords with a password manager whenever possible. Note that simply adding prefixes or suffixes before or after a single semantic factor does not effectively increase password security, as two such password patterns are among the weak patterns shown in Table X. The user-facing suggestions should not be taken rigidly and statically, since human users' password composition behaviors and password cracking techniques are both constantly evolving.

VI. Ethical Considerations

We considered ethical issues in our research following the common practice followed by other researchers. We used only password databases that were already leaked publicly, many of which are widely used standard databases in password-related research in the literature. We removed all non-password personal information from the databases and kept only passwords themselves for our research. We did not and will not redistribute the password databases we used to avoid potential misuse. Instead, reproducibility is supported by providing sufficient details of the password databases used and how we processed them.

VII. Conclusion

This paper presents SE#PCFG, a new framework and an associated computational process for analyzing password semantics in four different levels. By applying it to 17 leaked databases, we demonstrated how the framework can be used to produce useful new insights about password semantics and the underlying user behaviors. Then, we further proposed SEPCA, a semantic-aware password cracking architecture equipped by a general smoothing method. Our experiments with the 17 leaked databases showed that SEPCA could outperform other SOTA password cracking methods and it also performed very robustly across 52 test cases with different pairs of training and testing databases.

References

[1] B. Joseph, H. Cormac, P. C. Van Oorschot, and S. Frank, "The quest to replace passwords: A framework for comparative evaluation of web authentication schemes," in *Proceedings of the 2012 IEEE Symposium on Security and Privacy*. IEEE, 2012, pp. 553–567. [Online]. Available: https://doi.org/10.1109/SP.2012.44

[2] G. Guo and H. Wechsler, *Mobile Biometrics*. IET, 2017. [Online]. Available: https://doi.org/10.1049/PBSE003E

[3] C. Herley and P. C. van Oorschot, "A research agenda acknowledging the persistence of passwords," *IEEE Security and Privacy*, vol. 10, no. 1, pp. 28–36, 2012. [Online]. Available: https://doi.org/10.1109/MSP.2011.150

[4] S. Garfinkel and H. R. Lipford, *Usable Security: History, Themes, and Challenges*. Springer Nature, 2014. [Online]. Available: https://doi.org/10.1007/978-3-031-02343-9

TABLE X: Top 10 cracked SPs for each of the four subsets of password databases grouped by language, leading to in total 20 SPs representing different types of weak passwords.

SP	CN (%)	EN (%)	GE (%)	FR (%)
[NUMBER6]	**12.0**	**3.0**	**2.4**	**3.2**
[NUMBER7]	**10.1**	**1.4**	0.9	**1.5**
[NUMBER8]	**6.3**	**1.5**	**1.6**	**2.0**
[NUMBER9]	**3.8**	0.5	0.6	0.7
[YYMMDD]	**3.4**	0.4	0.2	0.3
[YYYYMMDD]	**3.0**	< 0.1	< 0.1	< 0.1
[WKNE]	**1.3**	**3.8**	**4.6**	**5.2**
[PY, PY, PY]	**1.1**	0.2	0.2	< 0.1
[MMDDYY]	**1.0**	0.8	0.4	1.0
[PRE1, NUMBER7]	**0.9**	< 0.1	< 0.1	< 0.1
[FEMALE_NAME]	< 0.1	**3.4**	**2.8**	**4.8**
[EN_NOUN]	0.2	**2.8**	**3.0**	**2.9**
[FEMALE_NAME, NUMBER2]	< 0.1	**2.1**	**3.9**	**2.0**
[EN_NOUN, NUMBER2]	< 0.1	**1.3**	**2.0**	0.9
[WKNE, NUMBER2]	0.3	**1.3**	**1.8**	1.4
[FEMALE_NAME, SUF1]	< 0.1	**1.3**	< 0.1	**1.5**
[FEMALE_NAME, NUMBER3]	0.3	0.9	**1.5**	0.7
[FEMALE_NAME, YEAR]	< 0.1	0.6	**1.4**	0.5
[MALE_NAME]	< 0.1	1.0	1.0	**1.6**
[DDMMYY]	0.2	0.7	0.5	**1.5**

[5] S. G. Lyastani, M. Schilling, S. Fahl, and S. Bugiel, "Better managed than memorized? studying the impact of managers on password strength and reuse," in *Proceedings of the 27th USENIX Security Symposium*. USENIX Association, 2018, pp. 203–220. [Online]. Available: https://www.usenix.org/conference/usenixsecurity18/presentation/lyastani

[6] B. L. Riddle, M. S. Miron, and J. A. Semo, "Passwords in use in a university timesharing environment," *Computers & Security*, vol. 8, no. 7, pp. 569–579, 1989. [Online]. Available: https://doi.org/10.1016/0167-4048(89)90049-7

[7] A. S. Brown, E. Bracken, S. Zoccoli, and K. Douglas, "Generating and remembering passwords," *Applied Cognitive Psychology*, vol. 18, no. 6, pp. 641–651, 2004. [Online]. Available: https://doi.org/10.1002/acp.1014

[8] R. Veras, J. Thorpe, and C. Collins, "Visualizing semantics in passwords: The role of dates," in *Proceedings of the 9th International Symposium on Visualization for Cyber Security*. ACM, 2012, pp. 88–95. [Online]. Available: https://doi.org/10.1145/2379690.2379702

[9] V. Rafael, C. Christopher, and T. Julie, "On the semantic patterns of passwords and their security impact," in *Proceedings of the 2014 Network and Distributed System Security Symposium*, 2014. [Online]. Available: https://doi.org/10.14722/ndss.2014.23103

[10] Z. Li, W. Han, and W. Xu, "A large-scale empirical analysis of Chinese web passwords," in *Proceedings of the 23rd USENIX Security Symposium*. USENIX Association, 2014, pp. 559–574. [Online]. Available: https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/li_zhigong

[11] W. Ding, Z. Zijian, W. Ping, Y. Jeff, and H. Xinyi, "Targeted online password guessing: An underestimated threat," in *Proceedings of the 2016 ACM Conference on Computer and Communications Security*. ACM, 2016, pp. 1242–1254. [Online]. Available: https://doi.org/10.1145/2976749.2978339

[12] 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 Security Symposium*. USENIX Association, 2019, pp. 1537–1555. [Online]. Available: https://www.usenix.org/conference/usenixsecurity19/presentation/wang-ding

[13] W. Han, Z. Li, M. Ni, G. Gu, and W. Xu, "Shadow attacks based on password reuses: A quantitative empirical analysis," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 2, pp. 309–320, 2018. [Online]. Available: https://doi.org/10.1109/TDSC.2016.2568187

[14] W. Han, M. Xu, J. Zhang, C. Wang, K. Zhang, and X. S. Wang, "TransPCFG: Transferring the grammars from short passwords to guess long passwords effectively," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 451–465, 2021. [Online]. Available: https://doi.org/10.1109/TIFS.2020.3003696

[15] C. M. Weir, "Probabilistic Context Free Grammar (PCFG) password guess generator," Online source code repository. [Online]. Available: https://github.com/lakiw/pcfg_cracker

[16] M. Weir, S. Aggarwal, B. de Medeiros, and B. Glodek, "Password cracking using probabilistic context-free grammars," in *Proceedings of the 2009 IEEE Symposium on Security and Privacy*. IEEE, 2009, pp. 391–405. [Online]. Available: https://doi.org/10.1109/SP.2009.8

[17] 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 *Proceedings of the 25th USENIX Security Symposium*. USENIX Association, 2016, pp. 175–191. [Online]. Available: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/melicher

[18] A. Narayanan and V. Shmatikov, "Fast dictionary attacks on passwords using time-space tradeoff," in *Proceedings of the 2005 ACM Conference on Computer and Communications Security*. ACM, 2005, pp. 364–372. [Online]. Available: https://doi.org/10.1145/1102120.1102168

[19] M. Dürmuth, F. Angelstorf, C. Castelluccia, D. Perito, and A. Chaabane, "OMEN: Faster password guessing using an ordered markov enumerator," in *Engineering Secure Software and Systems: 7th International Symposium, ESSoS 2015, Milan, Italy, March 4-6, 2015, Proceedings*. Springer, 2015, pp. 119–132. [Online]. Available: https://doi.org/10.1007/978-3-319-15618-7_10

[20] J. Ma, W. Yang, M. Luo, and N. Li, "A study of probabilistic password models," in *Proceedings of the 2014 IEEE Symposium on Security and Privacy*. IEEE, 2014, pp. 689–704. [Online]. Available: https://doi.org/10.1109/SP.2014.50

[21] B. Hitaj, P. Gasti, G. Ateniese, and F. Perez-Cruz, "PassGAN: A deep learning approach for password guessing," arXiv:1709.00440 [cs.CR], 2019. [Online]. Available: https://doi.org/10.48550/arXiv.1709.00440

[22] D. Pasquini, A. Gangwal, G. Ateniese, M. Bernaschi, and M. Conti, "Improving password guessing via representation learning," in *Proceedings of the 2021 IEEE Symposium on Security and Privacy*. IEEE, 2021, pp. 265–282. [Online]. Available: https://doi.org/10.1109/SP40001.2021.00016

[23] D. Wang, Y. Zou, Z. Zhang, and K. Xiu, "Password guessing using random forest," in *Proceedings of the 32nd USENIX Security Symposium (USENIX Security 23)*. USENIX Association, 2023, pp. 965–982. [Online]. Available: https://www.usenix.org/conference/usenixsecurity23/presentation/wang-ding-password-guessing

[24] M. Xu, J. Yu, X. Zhang, C. Wang, S. Zhang, H. Wu, and W. Han, "Improving real-world password guessing attacks via bi-directional transformers," in *Proceedings of the 32nd USENIX Security Symposium (USENIX Security 23)*. USENIX Association, 2023, pp. 1001–1018. [Online]. Available: https://www.usenix.org/conference/usenixsecurity23/presentation/xu-ming

[25] H. Li, Y. Wang, W. Qiu, S. Li, and P. Tang, "PassTSL: Modeling human-created passwords through two-stage learning," in *Information Security and Privacy: 29th Australasian Conference, ACISP 2024, Sydney, NSW, Australia, July 15-17, 2024, Proceedings, Part III*. Springer Nature, 2024, pp. 404–423. [Online]. Available: https://doi.org/10.1007/978-981-97-5101-3_22

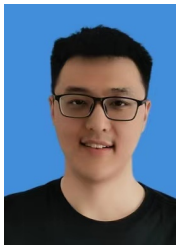
[26] S. Houshmand, S. Aggarwal, and R. Flood, "Next gen PCFG password cracking," *IEEE Transactions on Information Forensics and*

Security, vol. 10, no. 8, pp. 1776–1791, 2015. [Online]. Available: https://doi.org/10.1109/TIFS.2015.2428671
[27] S. Komanduri, "Modeling the adversary to evaluate password strength with limited samples," Ph.D. dissertation, Carnegie Mellon University, USA, 2016. [Online]. Available: https://doi.org/10.1184/R1/6720701.v1
[28] 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, no. 3, 2021. [Online]. Available: https://doi.org/10.1145/3448608
[29] W. Han, Z. Li, L. Yuan, and W. Xu, "Regional patterns and vulnerability analysis of Chinese web passwords," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 2, pp. 258–272, 2016. [Online]. Available: https://doi.org/10.1109/TIFS.2015.2490620
[30] D. Wang, Q. Gu, X. Huang, and P. Wang, "Understanding human-chosen PINs: Characteristics, distribution and security," in *Proceedings of the 2017 ACM Asia Conference on Computer and Communications Security*. ACM, 2017, pp. 372–385. [Online]. Available: https://doi.org/10.1145/3052973.3053031
[31] H. Zhang, C. Wang, W. Ruan, J. Zhang, M. Xu, and W. Han, "Digit semantics based optimization for practical password cracking tools," in *Proceedings of the 2021 Annual Computer Security Applications Conference*. ACM, 2021, pp. 513–527. [Online]. Available: https://doi.org/10.1145/3485832.3488025
[32] M. AlSabah, G. Oligeri, and R. Riley, "Your culture is in your password: An analysis of a demographically-diverse password dataset," *Computers & Security*, vol. 77, pp. 427–441, 2018. [Online]. Available: https://doi.org/10.1016/j.cose.2018.03.014
[33] C. Wang, S. Jan, H. Hu, and G. Wang, "Empirical analysis of password reuse and modification across online service," in *Proceedings of the 8th ACM Conference on Data and Application Security and Privacy*. ACM, 2017, pp. 196–203. [Online]. Available: https://doi.org/10.1145/3176258.3176332
[34] E. Liu, A. Nakanishi, M. Golla, D. Cash, and B. Ur, "Reasoning analytically about password-cracking software," in *Proceedings of the 2019 IEEE Symposium on Security and Privacy*. IEEE, 2019, pp. 380–397. [Online]. Available: https://doi.org/10.1109/SP.2019.00070
[35] M. Xu, C. Wang, J. Yu, J. Zhang, K. Zhang, and W. Han, "Chunk-level password guessing: Towards modeling refined password composition representations," *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pp. 5–20, 2021. [Online]. Available: https://doi.org/10.1145/3460120.3484743
[36] S. Li, Z. Wang, R. Zhang, C. Wu, and H. Luo, "Mangling rules generation with density-based clustering for password guessing," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 5, pp. 3588–3600, 2023. [Online]. Available: https://doi.org/10.1109/TDSC.2022.3217002
[37] Jens, Steube et al., "hashcat - advanced password recovery," Web page. [Online]. Available: https://hashcat.net/hashcat/
[38] OpenWall, "John the Ripper password cracker," Web page. [Online]. Available: http://www.openwall.com/john/
[39] D. Wang, H. Cheng, P. Wang, X. Huang, and G. Jian, "Zipf's law in passwords," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2776–2791, 2017. [Online]. Available: https://doi.org/10.1109/TIFS.2017.2721359
[40] Wikimedia Foundation, "WikiNameEntity," Web page. [Online]. Available: https://dumps.wikimedia.org/enwiki/latest/
[41] Urban Dictionary, "Urban Dictionary," Website. [Online]. Available: https://www.urbandictionary.com/
[42] GeoNames, "GeoNames data," Web page. [Online]. Available: http://download.geonames.org/export/dump/
[43] U.S. Social Security Administration, "Popular baby names," Web page. [Online]. Available: https://www.ssa.gov/oact/babynames/limits.html
[44] SunnyFresh, "Common name dictionary collection TOP1000 (Pinyin)," Web page. [Online]. Available: https://download.csdn.net/download/u011827798/9999625
[45] Lexique, "WorldLex: Blog, twitter and newspapers word frequencies for 66 languages," Website, 2011. [Online]. Available: http://www.lexique.org/
[46] Wikimedia Foundation, "DeWikitionary," Web page. [Online]. Available: https://dumps.wikimedia.org/dewiktionary/
[47] ——, "FrWikitionary," Web page. [Online]. Available: https://dumps.wikimedia.org/frwiktionary/
[48] M. Dell'Amico, P. Michiardi, and Y. Roudier, "Password strength: An empirical analysis," in *Proceedings of the 2010 29th IEEE Conference on Computer Communications*. IEEE, 2010. [Online]. Available: https://doi.org/10.1109/INFCOM.2010.5461951
[49] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd ed. Morgan Kaufmann, 2012. [Online]. Available: https://doi.org/10.1016/C2009-0-61819-5
[50] M. Dell'Amico and M. Filippone, "Monte Carlo strength evaluation: Fast and reliable password checking," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015, pp. 158–169. [Online]. Available: https://doi.org/10.1145/2810103.2813631
[51] R. Veras, "Semantic password guesser," Online repository. [Online]. Available: https://github.com/vialab/semantic-guesser
[52] W. Melicher, "Nerual network cracking," Online repository. [Online]. Available: https://github.com/cupslab/neural_network_cracking

Yangde Wang received the M.S. degree in Information Security from Shanghai Jiao Tong University, Shanghai, China, in 2013. He is currently a Ph.D. candidate in the School of Cyber Science and Engineering, Shanghai Jiao Tong University. His main research areas include computer forensics and password security.

Weidong Qiu received the Ph.D. degree in Computer Software Theory from Shanghai Jiao Tong University, Shanghai, China, in 2001 and received the M.S. degree in Cryptography from Xidian University, Xi'an, China, in 1998. He is currently a professor and doctoral supervisor in the School of Cyber Science and Engineering, Shanghai Jiao Tong University. His main research areas include data science, privacy computing, cryptography and computer forensics.

Peng Tang is a postdoctoral in the School of Cyber Science and Engineering, at Shanghai Jiao Tong University. He received his M.S. degree in Computer Science from Beijing University of Posts and Telecommunications in 2017 and his Ph.D. degree in Cyber Security from Shanghai Jiao Tong University in 2022. His research focuses on privacy protection, data science, and AI security.



Hao Tian received the M.S. degree in Electronics and Communications Engineering from Shanghai Jiao Tong University in 2021. He is currently working at Haitong Securities, with research interests that include AI security and data circulation security.

Shujun Li (M'2008, SM'2012) received his B.E. degree in Information Science and Engineering and Ph.D. degree in Information and Communication Engineering, both from Xi'an Jiaotong University, China, in 1997 and 2003, respectively. He is Professor of Cyber Security at the School of Computing and Director of the Institute of Cyber Security for Society (iCSS), University of Kent, U.K. His research interests are mostly about inter-disciplinary topics related to cyber security and privacy, human factors, digital forensics and cybercrime, multimedia computing, AI and data science. His work covers multiple application domains, including but not limited to cybercrime, social media analytics, digital health, smart cities, smart homes, and e-tourism. He received multiple awards and honors including the 2022 IEEE Transactions on Circuits and Systems Guillemin-Cauer Best Paper Award.