

MAYA: Addressing Inconsistencies in Generative Password Guessing through a Unified Benchmark

William Corrias

Sapienza University of Rome
Rome, Italy
corrias@di.uniroma1.it

Fabio De Gaspari

Sapienza University of Rome
Rome, Italy
degaspari@di.uniroma1.it

Dorjan Hitaj

Sapienza University of Rome
Rome, Italy
hitaj.d@di.uniroma1.it

Luigi V. Mancini

Sapienza University of Rome
Rome, Italy
mancini@di.uniroma1.it

Abstract—Recent advances in generative models have led to their application in password guessing, with the aim of replicating the complexity, structure, and patterns of human-created passwords. Despite their potential, inconsistencies and inadequate evaluation methodologies in prior research have hindered meaningful comparisons and a comprehensive, unbiased understanding of their capabilities. This paper introduces MAYA, a unified, customizable, plug-and-play benchmarking framework designed to facilitate the systematic characterization and benchmarking of generative password-guessing models in the context of trawling attacks. Using MAYA, we conduct a comprehensive assessment of six state-of-the-art approaches, which we re-implemented and adapted to ensure standardization. Our evaluation spans eight real-world password datasets and covers an exhaustive set of advanced testing scenarios, totaling over 15,000 compute hours. Our findings indicate that these models effectively capture different aspects of human password distribution and exhibit strong generalization capabilities. However, their effectiveness varies significantly with long and complex passwords. Through our evaluation, sequential models consistently outperform other generative architectures and traditional password-guessing tools, demonstrating unique capabilities in generating accurate and complex guesses. Moreover, the diverse password distributions learned by the models enable a multi-model attack that outperforms the best individual model. By releasing MAYA, we aim to foster further research, providing the community with a new tool to consistently and reliably benchmark generative password-guessing models. Our framework is publicly available at <https://github.com/williamcorrias/MAYA-Password-Benchmarking>.

1. Introduction

Despite the rise of increasingly secure authentication methods, traditional passwords remain the most widely used mechanism due to their usability and familiarity [9], [37]. However, users often utilize and reuse simplistic passwords across services, making them vulnerable to a variety of attacks [14], [8], [46], [19], [13]. As a result, password security has long been a focus of research. Brute-force and dictionary attacks remain the most prevalent password-cracking techniques, primarily exploiting weaknesses such as the use of weak or reused passwords. Tools like John

The Ripper (JTR) [49] and Hashcat [25] enhance these techniques by applying transformation rules to dictionaries of passwords, generating variations based on common patterns. Yet, effective transformation rules are complex to design and require extensive contextual knowledge and manual labor.

Beyond these limitations, traditional Machine Learning password guessing techniques, such as Probabilistic Context-Free Grammars (PCFGs) [31], [12], [63], [24] and Markov Models [38], [15], [22], [62], generate passwords based on statistical likelihood derived from real-world observed data. While these methods have demonstrated considerable effectiveness, they are fundamentally constrained by structural assumptions. Specifically, Markov Models, which rely on fixed-length n-grams, are inherently limited when modeling long-range dependencies and complex patterns. Likewise, PCFGs impose rigid template structures that constrain the diversity and flexibility of the generated password space.

In recent years, research has increasingly focused on harnessing advancements in generative models to enhance password guessing techniques [29], [45], [42], [48], [40], [65], [61], [55]. Unlike traditional approaches, these models aim to learn and replicate the complexity, structure, and patterns of human-created passwords without relying on prior assumptions. Nevertheless, the current literature suffers from inconsistencies, inadequate evaluation practices, and a lack of rigorous, standardized characterization of generative models' behaviors. Proposed approaches are often evaluated using inconsistent methodologies across studies and are tested in non-uniform, often simplistic settings, hindering meaningful comparisons and limiting a full understanding of their capabilities. As a result, critical aspects remain underexplored: how models behave across different settings, what types of passwords they tend to generate, how human-like their guesses are, whether they genuinely capture the underlying complexities of human password behavior, and where potential blind spots may lie. These limitations underscore a broader absence of systematic model characterization.

To address these gaps, we introduce MAYA¹, a unified and customizable plug-and-play benchmarking framework.

1. The name "MAYA" is inspired by A. Schopenhauer's philosophical concept of "the veil of Maya", representing an illusion that obscures true reality. Similarly, our framework aims to unveil the potential of password-guessing generative models, which has remained obscured until now.

While MAYA is specifically designed to support fair comparison and in-depth characterization and evaluation of generative password-guessing models in trawling attack scenarios, it builds upon a general, standardized, and rigorous evaluation methodology that can be applied across a wide range of guessing attacks, regardless of the underlying guessing approach. Using MAYA, we systematically evaluate six state-of-the-art generative models, each re-implemented with standardized data preprocessing, dependencies, and configurations. We focus on trawling password attacks, as they represent the most general and widely applicable setting. Our experimental setup spans eight real-world password datasets for training and testing, and covers a wide range of evaluation scenarios, totaling over 15,000 hours of computation.

Our key findings include: (a) Increasing password length yields diminishing returns in reducing guessability; similarly, increasing the number of generated passwords offers reducing improvements in successful guesses, as models tend to exhaust easily guessable passwords early in the generation process. (b) Rule-based tools are competitive on smaller datasets and traditional machine learning models excel on challenging ones. However, the best-performing generative models, on average, outperform both. (c) Generative models do not always require large datasets to effectively model data distribution. While some architectures benefit from larger datasets, others exhibit minimal improvement or even a decrease in performance as the training data increases. (d) Models can successfully guess passwords even when trained on datasets from different communities and/or cultures compared to the test set, highlighting strong generalization capabilities and suggesting the existence of common structures underlying human-created passwords across disparate groups. (e) While all models demonstrate strong capabilities in guessing common and simple passwords and a decrease in performance for rarer passwords, sequential models are the only ones that remain effective in guessing longer passwords and more complex patterns. (f) Different architectures learn generation functions with different probability distributions over the codomain, enabling their combination into a multi-model attack that outperforms individual models. (g) Generative models effectively capture various aspects of human-created passwords, generating high-quality and diverse passwords while minimizing mode failures. However, some models struggle to accurately replicate the length distribution.

This paper makes the following contributions:

- We analyze eight real-world leaked password datasets, providing a detailed characterization of each and examining the impact of factors such as dataset size, geographic origin, linguistic and cultural background, and temporal span on the resulting password distributions.
- We propose a rigorous and standardized evaluation methodology for trawling password-guessing attacks, addressing key methodological gaps in prior work. Our proposal enables fair comparisons, a rigorous empirical evaluation, and a comprehensive characterization of

each approach.

- We develop MAYA, a fully customizable, plug-and-play framework for evaluating generative password-guessing models in trawling attack scenarios. To support reproducibility and encourage further research, we publicly release our code and data at <https://github.com/williamcorrias/MAYA-Password-Benchmarking>.
- Leveraging MAYA, we characterize and benchmark six state-of-the-art generative models across eight datasets and a diverse set of testing scenarios, addressing seven key research questions. Our evaluation spans over 15,000 compute hours.
- We provide insights to guide future research in enhancing model password-guessing capabilities and integrating them into other password-related domains.

2. Motivation

The current body of research on generative password-guessing models suffers from several notable limitations. First, existing evaluation methodologies are inconsistent across studies, hindering meaningful comparisons. Second, evaluations often lack methodological rigor, typically relying on overly simplified scenarios that fail to provide a comprehensive or unbiased assessment of model performance. Third, there is a lack of systematic characterization, which restricts our understanding of what these models learn and how they behave under different conditions. These limitations motivate the need for a unified and comprehensive benchmarking framework that enables fair comparisons, subjects models to advanced and realistic scenarios, and offers a deeper insight into their underlying capabilities.

2.1. Lack of Consistency

Each existing approach adopts its own evaluation methodology, as models are neither trained nor assessed on the same data and under identical settings. These methodological inconsistencies encompass various factors, including data preprocessing algorithms, file encoding, vocabulary, maximum password length, the number of generated passwords, and the size of the training and testing datasets. As a result, fair comparisons across studies are challenging. For instance, differing data preprocessing methods lead to distinct training and testing distributions, making direct comparison challenging. Likewise, evaluating models with different maximum password lengths introduces inherent bias, as shorter passwords are generally easier to guess. Analogous considerations apply to other settings as well. Such inconsistencies highlight the necessity for a standardized evaluation methodology to serve as a foundation for future research.

2.2. Lack of Rigorousness

Existing research often suffers from an insufficiently rigorous evaluation. Models are typically evaluated using overly simplistic metrics, such as the percentage of guessed

passwords or the number of unique passwords generated, within restricted and similarly simplistic scenarios, often relying only on widely used datasets like RockYou or LinkedIn. Consequently, the current literature offers an incomplete evaluation that is biased toward these simplistic settings, underscoring the need for more robust methodology incorporating diverse datasets, varied scenarios, and more complex settings that reflect a broader spectrum of real-world contexts.

2.3. Lack of Characterization

The methodological issues outlined above lead to a broader problem: the lack of a systematic characterization of password-guessing models. Beyond aggregate performance metrics, current research fails to offer in-depth insights into crucial aspects of generative models, such as the human-likeness of generated passwords, the structural properties of real passwords these models capture, the types of passwords they generate (and those they fail to generate), the distinct distributions that different models learn, and how model behavior varies across different experimental conditions. Such insights are essential for a comprehensive understanding of these models, their true capabilities, and potential applications beyond password guessing.

2.4. Why Trawling Attacks

Password attacks are typically categorized based on the adversary’s scope: targeted and trawling. Targeted attacks aim to compromise specific user accounts by leveraging personal identifiable information (PII), and have gained increasing attention in recent years due to their growing effectiveness [35], [61], [43], [60], [27], [64]. In contrast, trawling attacks seek to guess as many user passwords as possible within a dataset, without targeting any specific individual. This attack model has long been the focus of password security research, with numerous techniques developed over time, including rule-based systems, PCFGs, Markov models, traditional neural networks, and deep generative models. In this work, we focus exclusively on trawling attacks, as they represent the most general and widely applicable class of password-guessing attacks, providing a broad evaluation framework. Moreover, since targeted attacks can be viewed as trawling attacks conditioned on PII, our evaluation methodology can effectively support both lines of research.

3. MAYA

This section presents MAYA, our unified and plug-and-play benchmarking framework tailored for password-guessing generative models in the context of trawling attacks. The framework offers an intuitive environment for training and testing generative models with minimal setup, enabling the research community to move in a unified direction. It also includes a comprehensive set of experimental settings, providing thorough benchmarking and detailed characterization

across multiple key metrics. MAYA features a highly modular and easily extendable architecture that enables the integration of new models and customized testing scenarios to support future research. While MAYA focuses on generative models, its underlying evaluation methodology (Section 3.1) provides a standardized testing environment that is applicable to all types of trawling password-guessing methods, ranging from traditional techniques to the latest approaches. MAYA currently implements six state-of-the-art password-guessing generative models (Section 3.2), eight real-world password datasets (Section 4), and a comprehensive set of testing scenarios aimed at answering seven key research questions (Section 3.3).

3.1. Methodology

MAYA addresses the methodological shortcomings identified in Section 2 by introducing a standardized and rigorous evaluation methodology. This methodology enables fair comparisons across approaches and supports thorough benchmarking and detailed characterization of trawling password-guessing methods.

3.1.1. Standardized Data Preprocessing and Settings.

We propose the following procedure to standardize data preprocessing: (1) open and read datasets using UTF-8 encoding while ignoring errors, (2) remove passwords that exceed the specified maximum length, contain non-ASCII characters, or include characters that are not in the provided vocabulary. (3) split the dataset following the standard 80% training and 20% testing ratios, (4) remove duplicates from the testing dataset, and (5) remove from the training dataset any overlap with the testing dataset. This approach ensures wide language compatibility through UTF-8 encoding, avoids double-counting by eliminating duplicates, and provides a fixed, consistent testing set across experiments. We note that existing works typically define the training set first, and then derive the testing set by removing any overlapping samples. This approach is undesirable, as varying the training set leads to changes in the testing set as well, thereby limiting the comparability across experiments. We further define our vocabulary to include all uppercase and lowercase letters, digits, and the following widely-accepted symbols: `~!@#$%^&*() , . <> / ? ' " { } [] \ _ = + ; : \ ``.

3.1.2. Advanced Evaluation Scenarios. We have designed a set of advanced and realistic evaluation scenarios to accurately and comprehensively assess the models’ capabilities and limitations, with the goal of providing a complete and nuanced understanding of their performance. Each scenario has been envisioned to address one of the seven research questions outlined in Section 3.3.

3.2. Models

We selected six state-of-the-art generative models for trawling password guessing, aiming to provide a comprehensive evaluation of different architectural approaches:

(1) FLA [40], based on an LSTMs, (2) PassGAN [29], and (3) PLR-GAN [45], both GAN-based approaches, (4) PassFlow [42], a flow-based model, (5) VGPT2 [7], which combines a VAE with GPT2-derived encoder and decoder blocks, and (6) PassGPT [48], an autoregressive transformer model based on the GPT2 architecture. To ensure standardization, each model was adapted and re-implemented in strict accordance with the procedures described in the respective papers, minimizing discrepancies arising from implementation variations. We provide further implementation details in Appendix A. We validated the accuracy of our implementations by comparing our results with those reported in the original papers.

FLA. FLA [40] (Fast, Lean, and Accurate), was the first approach to apply neural networks to the password-guessing task. It is the only approach we selected based on a recurrent neural network, specifically an LSTM, allowing us to examine how models designed for sequential data processing perform compared to other, more recent generative architectures.

PassGAN. PassGAN [29] is based on a Generative Adversarial Network (GAN) architecture, which, unlike other designs, follows an adversarial training approach. As they are implicit models, GANs learn to generate data by capturing the underlying distribution of the training set without explicitly defining a probability distribution, offering a unique perspective in password generation.

PLR-GAN. PLR-GAN [45] is an enhanced version of PassGAN and represents the current state-of-the-art for GAN-based methods. PLR-GAN further adopts a Dynamic Password Guessing (DPG) strategy, which allows the model to adapt its guesses based on the distribution of successfully guessed passwords, increasing the likelihood of generating relevant guesses.

PassFlow. Passflow [42] represents the first and only attempt to integrate flow-based generative models into the field of password guessing. Flow networks [44] offer an explicit latent space and an invertible mapping between a data point and its latent representation, enabling complex operations such as interpolation and exact latent variable inference. PassFlow adopts and further enhances DPG by integrating Gaussian Smoothing in the generation process, reducing the likelihood of generating duplicated passwords while maintaining the benefits of DPG.

VGPT2. VGPT2 [7] combines a Variational Autoencoder (VAE) with an encoder-decoder architecture based on GPT2. The VAE provides an explicit representation of the latent space, while GPT2 excels at processing sequential information and capturing long-term dependencies, making it a unique approach to analyze.

PassGPT. PassGPT [48] proposes a GPT-2-based language model for password generation. Similarly to FLA, PassGPT

employs a sequential generation process. However, GPT-2 is built upon an attention mechanism, which allows it to capture long-range dependencies more effectively.

3.3. Research Questions

This section presents the research questions and scenarios that guided the design of our evaluation, enabling the comprehensive characterization of model behavior and further comparison of their performance.

RQ1 - How Do Different Settings Influence Models Performance?. Generative password-guessing models have two primary settings: the maximum password length and the number of generated passwords. This RQ explores the impact of these two factors on guessing performance. We test three different maximum lengths (8, 10, and 12) and eleven generation quantities, from 10^6 to 5×10^8 . For each dataset, all models were trained three times—once per length—and evaluated across all generation quantities.

RQ2 - Are Generative Models Truly Better than Traditional Tools?. Despite recent advancements in generative architectures, there is limited evidence indicating whether generative models outperform traditional approaches, and few direct experimental comparisons have been conducted. We address this gap by conducting a comprehensive evaluation of generative models on 8 different datasets and comparing them to traditional tools such as John the Ripper (JtR) and Hashcat, offering a thorough, direct comparison between generative and rule-based methods.

RQ3 - How Sensitive are Models to Training Dataset Size?. In real-world scenarios, attackers often have limited or partial access to leaks. Therefore, evaluating model performance across varying training data subset sizes is essential for gaining a clear understanding of the effectiveness of generative models. This RQ examines the ability of generative models to reconstruct the full target data distribution starting from varying portions of the source dataset and execute a successful attack. We explore this capability by training models on up to seven different data subsets on four distinct datasets.

RQ4 - Can Models Generalize To Different Communities and/or Cultures?. A common real-world scenario involves attackers obtaining leaked passwords from a distribution different from their intended target, often due to cultural or community differences. This RQ investigates whether generative models can effectively generalize to unseen password distributions. We assess this capability through extensive cross-dataset analysis, testing the models on datasets distinct from those used during training. Specifically, we examine their ability to guess passwords across (1) different communities (i.e., same language but different online services) and (2) different cultures (i.e., different languages and cultural backgrounds), which, as demonstrated in Section 4.2, significantly impacts password distributions.

RQ5 - Are Models Limited to Guessing Only Simple and Common Passwords?. Simple passwords are easily guessed using traditional tools, whereas rare and complex passwords present a significantly greater challenge. The ability of generative models to arbitrarily sample from different areas of the password distribution offers the potential to guess even rare and complex passwords. We explore this RQ by dividing and categorizing the test dataset based on password frequency and assessing each model’s performance on the different subsets. Additionally, we analyze how well the models perform in guessing passwords of varying lengths and patterns.

RQ6 - To What Extent Do the Distributions Learned by Different Models Align? Can We Combine Models to Maximize Effectiveness?. Generative models are trained to generate data that matches the target distribution, learning this distribution either implicitly or explicitly. However, it remains unclear to what extent distributions learned by different models align, as various factors influence the learning process. If the distributions are not fully aligned, a multi-model attack could enhance guessing capabilities.

RQ6.1 - To What Extent Do the Distributions Learned by Different Models Align?. We explore the first part of this RQ using two metrics computed across all model pairs (M_1, M_2) : Jaccard Index and Mergeability Index. Jaccard Index measures the ratio between the intersection and union of the passwords generated by each model, providing a metric for the overall diversity of all passwords generated by M_1 and M_2 . Additionally, we introduce the Mergeability Index, which quantifies the ratio between the marginal gain achieved by combining the outputs of M_1 and M_2 over the best performance between M_1 and M_2 (see Eq. 2). This metric assesses the weighted improvement gained by combining M_1 and M_2 , compared to using only the best-performing model.

RQ6.2 - Can We Combine Models to Maximize Effectiveness?. We follow an iterative elimination approach to gain valuable insights into whether and to what extent combining multiple models enhances guessing effectiveness. We begin by combining passwords generated by all models and progressively remove those generated by the least effective model in terms of additional matched passwords. This process allows us to iteratively refine and identify effective combinations of models for any desired number of models to combine.

RQ7 - Do Models Truly Capture the Characteristics of Human-Like Passwords?. Despite ample research, it remains unclear to what extent the distribution learned by generative models aligns with that of human-created passwords. Evaluation metrics such as guess percentage are inherently limited, as they only measure whether generated passwords match the test data, without offering insights into the distribution of non-matching guesses. A major challenge in answering this RQ is how to quantify the distance between generative models’ password distribution and the overall distribution of human-like passwords. In

TABLE 1: Details of the selected datasets.

Dataset	#Pass	#Unique	Loc	Lang	Year	Service
Rockyou [51]	32,600,024	14,311,994	USA	EN	2009	Gaming
Linkedin [2]	60,650,662	60,591,405	Global	EN	2012	Social
Mail.ru [1]	3,723,472	2,260,454	RU	RU	2014	Mail
000webhost [11]	15,269,739	10,587,879	USA	EN	2015	Forum
Taobao [56]	7,492,035	6,165,957	CHN	ZH	2012	Ecomm
Gmail [41]	4,912,520	3,122,573	RU	RU	2014	Mail
Ashley Madison [66]	375,846	375,738	CA	EN	2015	Social
Libero [17]	667,680	418,400	IT	IT	2016	Mail

related fields, metrics such as Fréchet Inception Distance (FID) [28] and Inception Score (IS) [50] are commonly used to assess the generated data [10]. However, these metrics rely on pre-trained classifiers and are unsuitable for the password domain. We address this gap by identifying and adopting four alternative metrics: (1) CNN Divergence [21], (2) IMD [57], (3) α -Precision β -Recall Authenticity [4], and (4) MTopDiv [6]. These metrics were carefully selected for their ability to capture different aspects of the considered distribution. Appendix B provides a detailed explanation of each metric. We computed these metrics for each model across all datasets and averaged the results, offering a comprehensive assessment of the human-likeness of the generated passwords. Additionally, we complement this analysis by examining the length distribution of the generated passwords and comparing it to that of real passwords.

4. Datasets

Our framework incorporates eight real-world leaked password datasets, carefully chosen to ensure diversity in size, geographic origin, language, time of leakage, and service type. In the following sections, we provide a detailed description of each dataset and perform a statistical analysis to examine the distribution and characteristics of the passwords across these datasets, providing insights that form the basis for our subsequent experiments.

4.1. Datasets Selection

Table 1, presents the datasets selected for this study. To ensure that the selected datasets provide generalizable insights, the following criteria were considered in the selection process:

Dataset Size. We included datasets of varying sizes to evaluate model performance under different dimensions. The collection ranges from small datasets with a few hundred thousand passwords, such as Libero [17], to large-scale datasets with tens of millions of passwords, such as Rock-You [51], LinkedIn [2] and 000webhost [11].

Geographic Diversity. The selected datasets vary in their geographical provenance. Previous studies have shown that users from different countries exhibit distinct password creation behaviors, leading to diverse password distributions [58], [36], [18], [23], [59].

TABLE 2: Distribution of password lengths.

Dataset	1-5	6	7	8	9	10	11	12	13+
rockyou	4.33	26.06	19.30	19.99	12.12	9.06	3.56	2.10	3.49
linkedin	0.00	8.95	11.01	24.60	14.47	12.54	6.56	4.19	17.68
mailru	1.78	22.90	14.68	23.92	11.64	8.81	5.22	4.31	6.74
000webh	0.06	5.71	7.94	21.88	15.41	14.50	10.48	7.66	16.36
taobao	0.48	12.93	13.14	16.90	16.51	16.10	10.77	6.80	6.39
gmail	4.13	18.73	13.49	28.92	13.85	13.85	3.10	1.89	2.05
ashley	9.97	19.57	18.15	24.87	13.21	9.66	2.19	1.26	1.12
libero	0.09	15.49	11.34	31.30	16.34	11.32	5.54	3.83	4.76
Average	2.60	16.29	13.63	24.05	14.19	11.98	5.93	4.01	7.32
CDF	2.60	18.90	32.53	56.58	70.77	82.75	88.68	92.68	100.00

Language and Cultural Background. We selected datasets representing diverse languages and cultural contexts, as users from different backgrounds exhibit variations in password choices [58], [5], [36], [59].

Temporal Span. We consider the dataset leak date as an important factor, as password policies and user awareness have evolved over time, with stronger requirements introduced in recent years [39], [33], [16]. Consequently, we expect older leaks to contain weaker passwords, while more recent datasets are expected to contain more secure passwords. Our dataset collection spans from 2009 to 2016.

Service Types and Community Background. We selected datasets from a wide range of services and communities, such as social networks, forums, e-commerce sites, dating sites, email services, and gaming platforms. The type of service and the nature of the user community can significantly influence password selection strategies, with more sensitive service and close-knit communities often prompting users to adopt different password behaviors [54], [3], [30].

4.2. Dataset Analysis

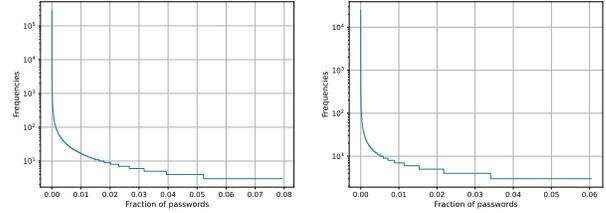
This section presents the main insights obtained from our statistical analysis of the selected datasets.

4.2.1. Password Length Analysis. Table 2 shows that the most common password lengths are between 6 and 10 characters. Taobao, 000webhost, LinkedIn, and Libero enforce a stricter policy (len \geq 6), with very few passwords shorter than 6 characters. The Cumulative Distribution Function (CDF) reveals that over 50% of passwords are at most 8 characters long, with 8 being, on average, the most frequently chosen length. Notably, the CDF rapidly increases up to 12 characters, suggesting that passwords longer than 12 characters are rare. Based on these observations, we selected passwords with maximum lengths of 8, 10, and 12 in our experiments.

4.2.2. Password Patterns Analysis. Based on common password rules and user behaviors, we defined 19 patterns, listed in Table 3, to characterize password patterns across the datasets. For a detailed distribution analysis, see Table 10 in the Appendix. In most datasets, letter-only passwords are highly prevalent and are almost always exclusively lowercase, whereas uppercase-only or mixed-case passwords appear

TABLE 3: Selected patterns and their description.

ID	Description	ID	Description
r1	Letters only.	r10	Starts letter ends digit.
r2	Lowercase letters only.	r11	Starts letter ends special.
r3	Uppercase letters only.	r12	Starts digit then only letters.
r4	Digits only.	r13	Starts digit ends special.
r5	Special only.	r14	Starts and ends with digit.
r6	Letters and digits.	r15	Starts special, then only letters.
r7	Letters and special.	r16	Starts and ends with special.
r8	Digits and special.	r17	Starts special, ends digit.
r9	Letters, digits, and special.	r18	Ends with '!'.
		r19	Ends with '1'.



(a) RockYou

(b) 000webhost

Figure 1: Password frequency distribution. The x-axis value is computed over the total number of passwords without duplicates.

infrequently, apart from 000webhost, likely influenced by a stricter password policy enforced prior to the breach. In European and American datasets, digit-only passwords are less frequent than letter-only ones, whereas Taobao (from China) shows a higher prevalence of digit-only passwords, consistent with prior studies [58], [5], [36]. This is likely due to users being less familiar with the Latin alphabet. Passwords comprised entirely of special characters are rarely used, as they are hard to remember, and users prioritize usability over security. A common pattern across all datasets is the combination of letters and digits. In 000webhost \approx 93% of passwords follow this pattern, suggesting a policy requiring at least two character classes. Users often find password policies frustrating [53], [52], [32] and tend to fall into predictable patterns to comply with these requirements, such as appending a trailing digit [58], [59]. Our analysis supports this observation, as passwords ending with a digit are highly prevalent, with nearly half of all passwords following this pattern and ‘1’ being widely used as a final character.

4.2.3. Top-10 Passwords Analysis. We analyzed the top 10 passwords in each dataset, finding that common choices like ‘123456’ appear across almost all datasets. Notably, password preferences vary by region: European users favor lowercase-only passwords, Chinese users often choose digit-based passwords resembling words phonetically, and Russian users prefer keyboard patterns. Table 11 in the Appendix presents the top 10 most common passwords.

4.2.4. Frequency Distribution. Figure 1 illustrates the frequency distribution of passwords for RockYou and 000webhost. To enhance readability and reduce the long-tail effect caused by less frequent passwords, we focus on passwords

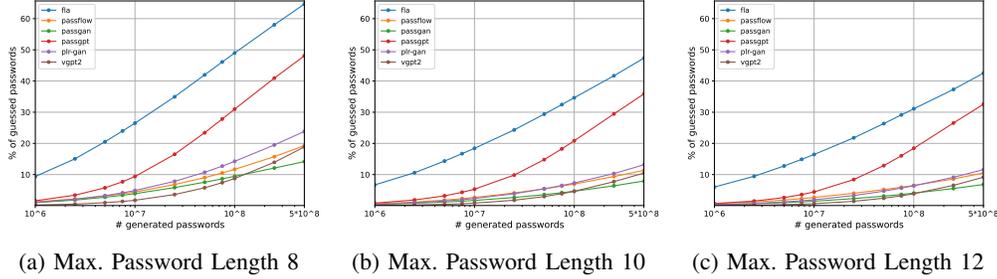


Figure 2: Impact of maximum password length on guessing performance over varying generation quantities.

appearing at least three times in the dataset. A clear trend clearly emerges: a small subset of passwords are highly common, while the majority are rare. This behavior is characteristic of a Zipf-like distribution, where the frequency of a sample is inversely proportional to its rank. This observation aligns with the findings of Wang et al. [58], who first demonstrated that Zipf’s law accurately models the distribution of human-chosen passwords.

5. Experiments

We provide a thorough characterization of each selected model (Sections 5.1–5.7), addressing the research questions from Section 3.3, and laying the groundwork for the benchmark in Section 5.8.

5.1. RQ1 - How Do Different Settings Influence Models Performance?

We examine the impact of two key parameters—maximum password length and the number of generated passwords—on model performance. Specifically, we compute the weighted average percentage of guessed passwords across all datasets, considering maximum lengths of 8, 10, and 12 characters. For each length, we evaluate 11 generation quantities, ranging from 10^6 to 5×10^8 passwords. Results are shown in Figure 2. As expected, performance declines with increasing password length. However, this decline exhibits diminishing returns: while the drop from 8 to 10 characters is substantial, averaging 10.46%, the decrease from 10 to 12 characters is markedly smaller, at just 2.15%, indicating a plateau as length increases. Among all models, FLA consistently outperforms the others, successfully guessing a substantial portion of passwords with relatively few guesses. Notably, FLA requires fewer than 10^7 generated passwords to outperform all other models except PassGPT. PassGPT, while initially comparable to PLR-GAN, PassFlow, and PassGAN, exhibits a much steeper growth curve, ultimately closing the gap with FLA after 5×10^7 guesses. When considering few guesses, PLR-GAN, PassFlow, and PassGAN exhibit nearly identical performance, while VGPT2 initially underperforms. As the number of guesses increases, differences between the models become more pronounced: PLR-GAN begins to outperform PassFlow, and VGPT2, despite a slow start,

improves faster than both. In contrast, PassGAN gains the least from additional guesses, with a relatively slow improvement rate.

While it is evident that generating more passwords leads to an increased number of matches, we delve deeper into this trend by analyzing the *marginal gain*—defined as the percentage increase in guessed passwords between two generation intervals. We analyze both *total gains* (relative to the overall number of correct matches) and *relative gains* (relative to the previous number of matches). Due to space constraints, detailed results are reported in Table 12 in the Appendix. Overall, all models exhibit clear diminishing returns, with sub-linear growth in successful guesses. Match rates grow rapidly in the early stages, then taper off as the space of common passwords is exhausted. Since this pattern is consistent across all maximum lengths, and considering that over 92% of passwords are 12 characters or fewer (see Table 2), we limit our subsequent analysis to this length.

5.2. RQ2 - Are Generative Models Truly Better Than Traditional Tools?

We compare generative models against two categories of traditional password-guessing techniques: (1) rule-based attacks, namely, Hashcat and John the Ripper, both operating in wordlist mode, using ‘Unicorn Rules’ and ‘Wordlist’ rulesets, respectively; and (2) machine learning approaches, specifically a PCFG model and OMEN, a Markov-based attack. Additional details on these tools are provided in Appendix A.

As shown in Figure 3, rule-based tools demonstrate strong performance on smaller datasets, such as Ashley Madison and Libero. In these cases, Hashcat outperforms all models except FLA. Both OMEN and PCFG achieve results comparable to PassGPT, successfully guessing a significant portion of the test passwords, while JtR follows closely behind.

However, as dataset size increases, the advantage shifts toward generative and machine-learning-based models, with the performance gap over rule-based tools widening considerably. On average, Hashcat underperforms compared to the top four learning-based approaches, while JtR consistently ranks as the least effective. We further investigated the rationale behind this declining performance, hypothesizing it stems from their underlying generation strategies. Unicorn Rules are ordered by efficacy, and we selected the first

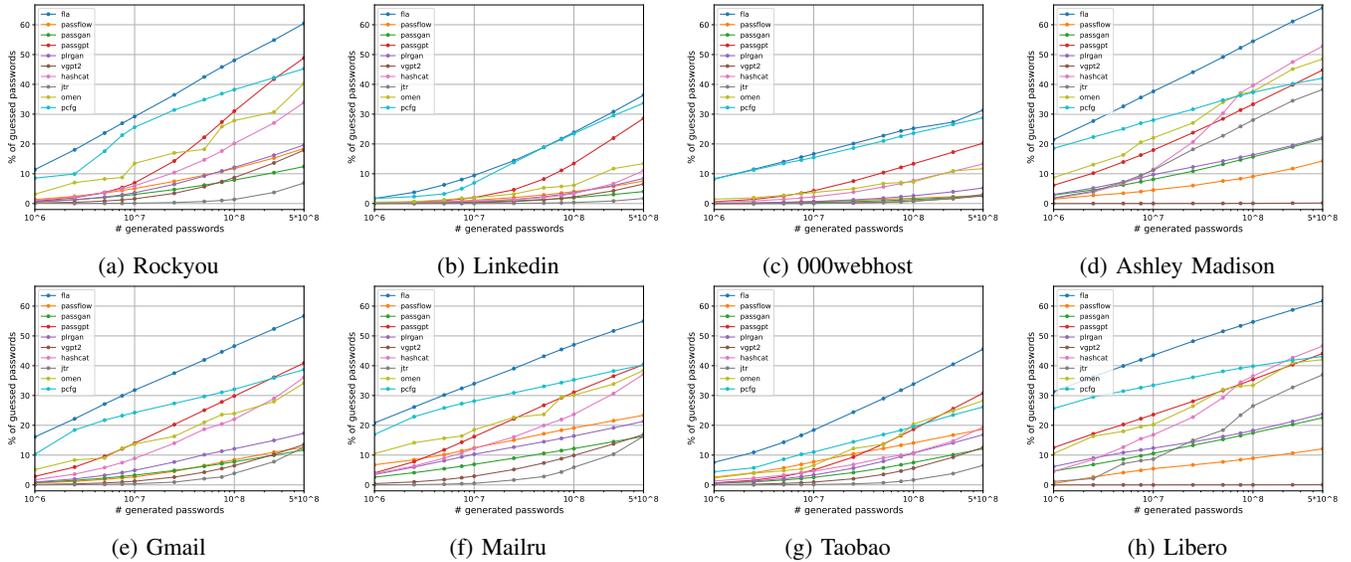


Figure 3: Comparison between traditional methods and generative models across 8 datasets.

TABLE 4: Impact of training dataset size on performance. Values expressed as percentage of guessed test set passwords.

Train Size	Mailru					Taobao					RockYou					LinkedIn								
	PGAN	PLR	PFLW	PGPT	VGPT	FLA	PGAN	PLR	PFLW	PGPT	VGPT	FLA	PGAN	PLR	PFLW	PGPT	VGPT	FLA	PGAN	PLR	PFLW	PGPT	VGPT	FLA
1e6	12.73	21.26	23.27	32.93	6.25	53.14	13.15	16.45	17.58	27.51	4.24	42.94	9.60	19.29	18.21	30.98	5.97	56.13	4.42	7.83	7.06	12.49	1.48	18.48
2e6	16.16	21.30	23.24	40.35	16.90	54.95	10.16	16.29	18.65	29.11	11.28	44.12	11.31	19.21	18.35	41.72	13.75	57.36	4.38	8.08	7.12	17.97	4.44	24.61
3e6	-	-	-	-	-	-	10.80	16.05	17.21	29.81	11.37	44.79	12.78	17.69	19.32	44.01	15.25	60.45	3.79	7.92	6.88	19.75	4.55	29.30
5e6	-	-	-	-	-	-	12.11	16.82	18.49	30.80	12.56	45.53	12.22	18.06	19.35	46.40	16.49	60.08	2.52	8.38	6.84	22.00	5.38	37.89
1e7	-	-	-	-	-	-	-	-	-	-	-	-	12.36	19.65	18.46	48.85	17.90	60.47	3.33	6.53	7.00	25.27	5.97	41.05
2e7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	4.87	7.12	7.13	26.91	6.33	35.10
4e7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3.99	8.44	7.10	28.58	6.56	36.37

n passwords generated by Hashcat accordingly. However, Hashcat applies a rule-first strategy: it processes all rules for the current password before moving to the next. As the dataset size increases, a smaller portion of the test set passwords are transformed, concentrating Hashcat’s guesses in a small subset of the overall password space. JtR follows a password-first strategy, but WordList’s rules are unsorted, so we randomly sampled n passwords from the generated set. In larger datasets, the number of generated candidates increases substantially, diluting the proportion of successful guesses and making it less likely to sample correct guesses.

On the two most challenging datasets, LinkedIn and 000webhost, where all models exhibit the lowest guess rates, PCFG closely matches FLA’s performance, trailing by only a few percentage points. PassGPT is the only other approach nearing their effectiveness, while Hashcat and OMEN show comparable but noticeably lower performance. Notably, except for these two datasets, PCFG successfully guesses a large number of passwords in the early stages, nearly matching FLA, but its guessing curve increases more slowly over time, likely due to early saturation.

Overall, learning-based approaches outperform traditional tools, especially as scale and complexity grow. Among them, on average, FLA and PassGPT lead, while older ML techniques like PCFG still show strong performance.

To streamline subsequent analyses, we fix the number of generated passwords at 5×10^8 , as it yields the best results.

5.3. RQ3 - How Sensitive are Models to Training Dataset Size?

We assess the ability of generative models to capture the full password distribution when trained on different data subset sizes. Models were trained on up to seven different subset sizes across four datasets, with a minimum initial size of 1M passwords. Results in Table 4 illustrate the models’ varying performance. Transformer-based models (PassGPT and VGPT2) consistently improve as the training dataset size increases, with VGPT2, in particular, struggling on smaller subsets. FLA also generally benefits from larger training subsets, although the rate of improvement tapers off between $3e6$ and $1e7$ passwords. We also observe an anomalous behavior on LinkedIn, where performance drops significantly after $1e7$ —or 25% of the dataset size. PLR-GAN excels on small training subsets, but its performance declines around 30% of the dataset size before improving again, ultimately achieving its best performance on the full dataset. PassFlow remains the most consistent model across subset sizes, with minimal performance variation. In contrast, PassGAN exhibits highly variable results, with performance peaks at different dataset sizes, making its behavior less predictable. Overall, transformer models show the most significant improvement with increasing training data, whereas other architectures tend to exhibit flat or minimal gains, challenging the conventional assumption that

TABLE 5: Cross-community generalization ability. Values expressed as percentage of guessed test set passwords.

Train / Test	PassGAN			PLR-GAN			PassFlow			PassGPT			VGPT2			FLA		
	Link	Rock	000W	Link	Rock	000W	Link	Rock	000W	Link	Rock	000W	Link	Rock	000W	Link	Rock	000W
LinkedIn	3.99	6.51	1.80	8.44	12.65	3.65	7.10	6.61	1.93	28.58	36.21	16.90	6.56	11.48	2.95	36.37	45.09	19.01
RockYou	4.72	12.36	1.59	8.77	19.65	4.65	8.16	18.46	3.55	22.08	48.85	13.15	6.93	17.90	2.84	31.53	60.47	17.31
000Webhost	2.27	4.10	2.63	3.70	5.56	5.22	6.58	11.44	2.69	10.41	13.77	20.24	1.87	3.15	2.59	22.81	26.72	32.28

TABLE 6: Cross-culture generalization ability. Values expressed as percentage of guessed test set passwords.

Train / Test	PassGAN			PLR-GAN			PassFlow			PassGPT			VGPT2			FLA		
	Rock.	Mail.	Taob.	Rock.	Mail.	Taob.	Rock.	Mail.	Taob.	Rock.	Mail.	Taob.	Rock.	Mail.	Taob.	Rock.	Mail.	Taob.
RockYou	12.36	8.43	5.57	19.65	13.22	9.10	18.46	14.86	9.83	48.85	22.30	13.65	17.90	11.40	6.52	60.47	30.10	18.71
Mailru	7.39	16.16	4.72	10.78	21.30	7.11	16.35	23.24	13.25	15.82	40.35	10.74	6.89	16.90	4.11	26.98	54.95	16.36
Taobao	6.41	7.39	12.11	9.37	10.11	16.82	19.55	20.20	18.49	13.26	13.61	30.80	7.74	8.16	12.56	23.77	20.94	45.53

more training data universally leads to better performance.

5.4. RQ4 - Can Models Generalize To Different Communities and/or Cultures?

We investigate the generalization capabilities of generative models by evaluating them in two cross-dataset scenarios: (1) cross-community and (2) cross-culture. In the cross-community setting, we use three datasets—RockYou, 000webhost, and LinkedIn—that share the same language but represent distinct user communities. For the cross-cultural setting, we used RockYou, Mailru, and Taobao, which differ in language, cultural background, and community.

5.4.1. Cross-community. As shown in Table 5, Cross-community generalization is strongly model- and dataset-dependent. PassGAN, PLR-GAN, PassFlow, and VGPT2 achieve their highest performance on LinkedIn when trained on RockYou, suggesting that the distribution learned from RockYou generalizes well despite community differences. However, stronger models such as PassGPT and FLA experience notable performance drops in the same scenario. This contrast leads us to hypothesize that the apparent generalization success of weaker models stems from their limited capacity to capture fine-grained details of the training distribution—resulting in broader, but less accurate, generalization. We also note how all approaches struggle to generalize when training on 000webhost, highlighting that performance significantly degrades when the source and target distributions diverge too strongly (see Table 10).

5.4.2. Cross-culture. In the cross-culture scenario, model performance follows a more expected pattern, as illustrated in Table 6: models trained on one cultural or linguistic context tend to perform significantly worse when evaluated on datasets from a different one. This highlights the challenges of generalizing across culturally distinct password distributions and underscores the importance of training data alignment with the target population. However, some interesting behaviors still emerge. Despite the challenging setting, generative models still manage to guess a non-negligible portion of the target passwords. Notably, models that typically

underperform in other settings—such as PassFlow—can surpass stronger models like PassGPT when there is a significant mismatch between source and target distributions. This suggests that certain models may possess greater flexibility or robustness in the face of distributional shifts, even if they are less effective under ideal, in-distribution conditions.

5.5. RQ5 - Are Models Limited to Guessing Only Simple and Common Passwords?

We evaluate the models’ ability to guess both common/rare and simple/complex passwords through three complementary analyses.

5.5.1. Analysis by Password Frequency. From each test dataset, we created three subsets based on password frequency: top 5%, top 10%, and bottom 90%. Since nearly 90% of test passwords across all datasets are unique (see Figure 1), these subsets allow us to evaluate models under three scenarios: (1) very common, (2) common, and (3) rare passwords. We excluded LinkedIn and Ashley Madison, as they consist entirely of unique passwords. Results, shown in Figure 4, are reported as weighted averages. As expected, all models perform better on common passwords. While the drop from the top 5% to the top 10% is small, performance declines sharply when focusing on the bottom 90%. Still, models are surprisingly capable of guessing rare passwords, with PassGPT and FLA guessing close to 40% and 50%, respectively. In Figure 5, we leverage PassFlow’s encoder to visualize password embeddings and observe that the bottom 90% of passwords tend to form clusters around the top 10%. Since PassFlow’s latent space is smooth [42], this behavior suggests that many rare passwords are slight variations of frequently used ones, which may explain why models are still able to guess a non-negligible portion of them despite their low frequency.

5.5.2. Analysis by Password Length. We analyzed matches by password length, ranging from 4 to 12 characters. Results, computed using a weighted average across all datasets, are shown in Figure 6. Notably, all models perform well on short passwords, with FLA nearly achieving a 100% match

TABLE 7: Guessing performance by password pattern. Values expressed as percentage of guessed passwords for each pattern.

Model	r1	r2	r3	r4	r5	r6	r7	r8	r9	r10	r11	r12	r13	r14	r15	r16	r17	r18	r19
PassGAN	6.94	7.65	0.16	28.31	0.30	3.55	0.10	0.23	0.08	4.12	0.10	0.53	0.12	24.17	0.10	0.01	0.07	0.06	7.53
PLR-GAN	12.60	13.74	1.35	39.95	0.67	7.31	0.77	0.74	0.27	8.39	0.64	2.14	0.47	34.17	0.35	0.03	0.15	1.02	12.49
PassFlow	10.38	11.35	1.41	27.92	5.34	4.82	0.70	0.30	0.05	5.01	0.41	2.29	0.21	24.02	1.79	0.45	0.16	0.39	9.54
PassGPT	38.47	39.66	33.74	46.55	25.82	29.81	20.04	19.90	11.19	33.43	17.88	29.62	10.39	41.68	18.19	12.72	5.04	21.63	36.16
VGPT2	12.09	13.11	2.38	29.39	2.59	5.37	1.22	0.90	0.33	6.10	0.95	2.15	0.53	25.19	0.61	0.04	0.09	1.07	9.74
FLA	46.51	48.14	38.49	67.56	17.51	39.38	21.02	26.28	14.20	44.63	21.32	26.39	11.88	59.87	13.92	8.50	4.58	24.57	48.09

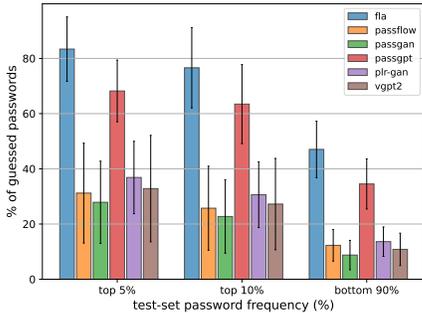


Figure 4: Guessing performance by password frequency group.

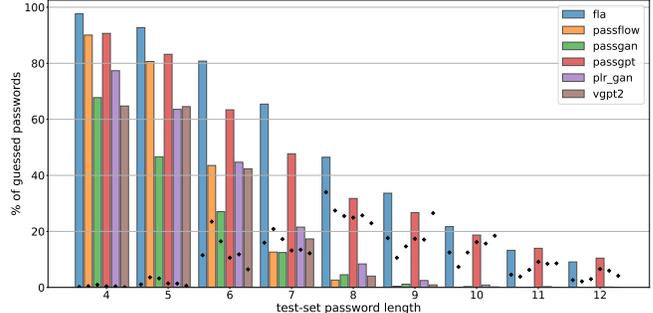


Figure 6: Guessing performance by password length. Dots represent the percentage of generated passwords for each length.

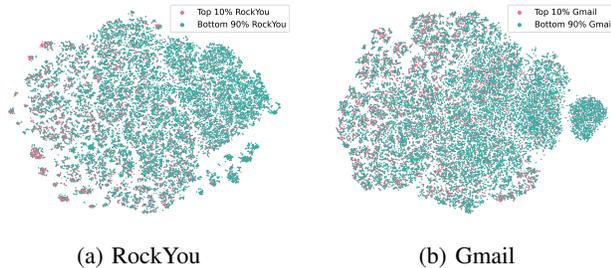


Figure 5: t-SNE plot showing the projection of Top 10% and Bottom 90% passwords in PassFlow’s latent space. Embedding computed using PassFlow’s encoder.

rate. However, performance declines as length increases. Only FLA and PassGPT maintain a substantial guessing rate beyond 8 characters. Notably, consistent with findings from RQ1, PassGPT surpasses FLA from length 11 onward. PassFlow initially performs on par with PassGPT for shorter lengths but undergoes a sharp decline, becoming the weakest performer from length 8 onward.

5.5.3. Analysis by Password Patterns. We evaluated the models’ ability to guess passwords based on the structural patterns defined in Table 3. The results, presented in Table 7 as a weighted average across all datasets, show that PassGPT and FLA consistently achieve the strongest performance, standing out as the only models capable of successfully guessing passwords across all defined patterns. In contrast, the remaining models show limited pattern coverage, particularly struggling with complex password structures that include special characters (r5, r7, r8) or uncommon character combinations (r11, r12, r13). GAN-based models fare

especially poorly on passwords composed entirely of special characters (r5), significantly underperforming PassFlow and VGPT2. This suggests a clear limitation in capturing rare or unconventional character sets. In contrast, all models exhibit strong performance on digit-only passwords (r4), indicating that simple patterns are consistently learned regardless of the architecture.

5.6. RQ6 - To What Extent Do the Distributions Learned by Different Models Align? Can We Combine Models to Maximize Effectiveness?

We assess the extent to which generative models’ learned distributions overlap and investigate the effectiveness of combining multiple models to enhance guessing performance.

5.6.1. RQ6.1 - To What Extent Do the Distributions Learned by Different Models Align?. We investigate the first aspect of this RQ utilizing two primary metrics: the Jaccard Index and the Mergeability Index. Their mathematical definitions are provided in Appendix B, in Equations 1 and 2, respectively. The Jaccard Index provides a quantitative measure of the overlap between two sets of generated passwords, with a value closer to 1 indicating that the two models produce similar distributions, and a value closer to 0 indicating that they generate distinct sets of passwords. As illustrated in Figure 7a, the FLA-PassGPT pair is the only combination with a Jaccard Index greater than 0.1, suggesting that these two models share some similarities in their password generation patterns. All other pairs exhibit much lower Jaccard values. Interestingly, despite PassGAN and PLR-GAN both being based on GANs, their Jaccard

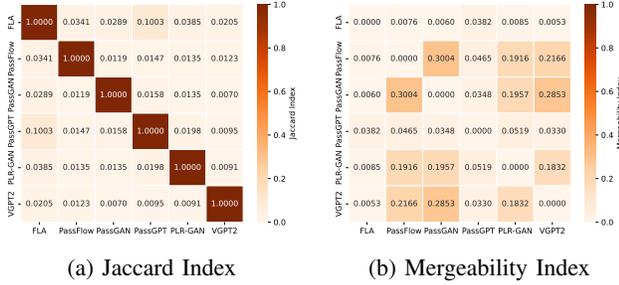


Figure 7: Heatmaps of the Jaccard Index (a) and Mergeability index (b).

Index is quite low. Similarly, VGPT2-PassGPT also shows a low Jaccard Index. These results highlight that, even within the same architectural family, models may produce highly distinct password distributions, suggesting that combining multiple models could be beneficial for maximizing coverage of the password space.

While the Jaccard Index provides insights into the overlap of generated passwords, it does not account for their effectiveness in terms of actual matches with real-world data. To address this, we introduce the Mergeability Index, a complementary metric that measures the benefits of combining the output of two models based on their successful guesses. It captures the performance improvement achieved by merging the guesses relative to the best-performing model. A Mergeability Index close to 0 indicates that the models guess mostly the same set of passwords, while a value close to 1 indicates largely distinct matching password sets. The results are shown in Figure 7b. Interestingly, FLA exhibits the lowest Mergeability across all models, indicating that its guessed passwords are generally a superset of those generated by the other models. This suggests that FLA has already captured a significant portion of the password space, and merging it with other models provides only limited additional benefit. In contrast, PassFlow, PassGAN, PLR-GAN, and VGPT2 show high Mergeability, with their varying combinations improving successful guesses by 18% to 30%. This suggests that these models guess highly distinct sets of passwords, meaning they are each capturing different aspects of the password space.

5.6.2. RQ6.2 - Can We Combine Models to Maximize Effectiveness?. We assess the potential of a multi-model attack by evaluating the performance gain achieved through the combination of the generated password sets from n models. Our results, presented in Figure 8, show the relative gain with respect to the best-performing single model, computed as percentage points on the test set. We identify the following optimal model combinations for different values of n : $n_1 = \{FLA\}$, $n_2 = n_1 + PassGPT$, $n_3 = n_2 + PLR-GAN$, $n_4 = n_3 + PassFlow$, $n_5 = n_4 + PassGAN$, and $n_6 = n_5 + VGPT2$. In datasets like LinkedIn and RockYou, combining models yields significant gains over FLA alone—especially PassGPT. Remarkably, combining PassGPT and FLA provides a larger average improvement than adding the

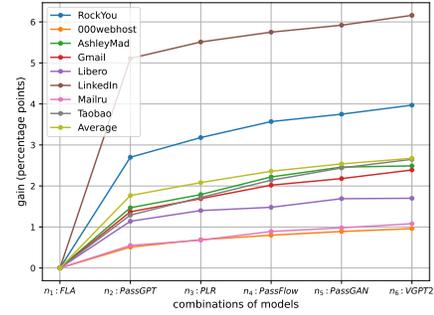


Figure 8: Multi-model guessing performance. Expressed as percentage points increase of guessed passwords relative to FLA baseline.

remaining four models combined. These results mirror those presented in Figure 7b, indicating that FLA and PassGPT capture somewhat complementary parts of the password space. However, in datasets such as 000webhost and Mailru, FLA alone performs nearly as well as the combination of all models, with only a 1 percentage point improvement in the multi-model attack.

5.7. RQ7 - Do Models Truly Capture the Characteristics of Human-Like Passwords?

We assess the similarity between the generated password distribution and that of human-created passwords using multiple metrics and an analysis of their length distributions.

5.7.1. Metrics-Based Evaluation. We identified four metrics to evaluate the human-likeness of the generated passwords, each capturing different aspects and characteristics:

CNN Divergence [21]. CNN Divergence utilizes a critic network trained to differentiate between real and synthetic data, with the loss serving as an estimator of their divergence. However, its reliability is influenced by the dataset size, and it may show biases when used to evaluate models trained with similar objectives, like GANs.

IMD [57]. IMD is an intrinsic, multi-scale metric that compares the data manifolds of real and generated samples and is applicable across diverse domains. However, its effectiveness is highly sensitive to the choice of feature representation, which can introduce bias into the distance estimation.

α -Precision β -Recall Authenticity [4]. This metric characterizes distributions along three key dimensions: fidelity (precision) and diversity (recall) of the generated data, and the generalization (authentication) capabilities of the models. It is effective in identifying different types of mode failures, making it a versatile tool applicable across various domains. However, as it relies on a neural network to embed data into a hypersphere, it may introduce bias through the learned latent representation, potentially affecting the estimation of the radius used to distinguish inliers from outliers.

TABLE 8: Distance between human- and generative model-created passwords.

Models	CNN Div	α -Precision	β -Recall	Auth	IMD	MTopDiv
PassGAN	16%	19%	4%	14%	65%	1%
PLR-GAN	6%	4%	3%	11%	3%	0%
PassFlow	56%	61%	52%	16%	200%	36%
PassGPT	2%	3%	1%	6%	0%	0%
VGPT2	29%	53%	34%	4%	135%	12%
FLA	12%	-15%	-1%	31%	172%	0%

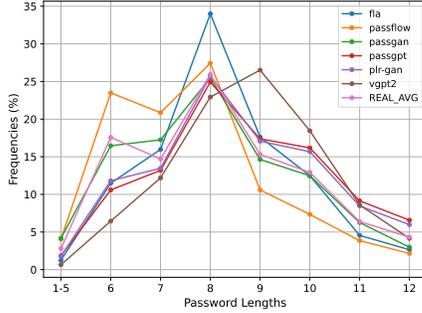


Figure 9: Password length distributions.

MTopDiv [6]. MTopDiv measures topological discrepancies between real and synthetic distributions at multiple scales, effectively detecting mode failures, without relying on pre-trained networks.

To contextualize the metric values, we define two baselines: (1) a soft lower bound (optimal case), defined as the metric value between test and training passwords; (2) a soft upper bound (worst case), defined as the metric value between randomly generated and test passwords. For each model, we normalize the metric value between lower (0%) and upper (100%) bounds, as displayed in Table 8. Values close to zero indicate closer alignment between the generated and human-created password distributions. GAN-based models demonstrate generally good performance across all metrics, with PLR consistently achieving lower values than PassGAN. PassGPT generates passwords that closely resemble human-created ones, likely due to its underlying GPT2 architecture, which is well-suited for modeling textual distributions. In contrast, PassFlow demonstrates the weakest performance, particularly in the IMD metric, where its score exceeds the soft upper bound by a factor of two. We attribute this behavior to PassFlow’s Gaussian Smoothing (GS) strategy, which introduces random perturbations in the generation process. While GS enhances uniqueness [42], it artificially distorts the distribution of generated passwords, ultimately reducing their quality. VGPT2 and FLA exhibit similar trends, performing strongly across most metrics except for IMD, where elevated values across most models suggest that they all struggle to fully capture certain characteristics of human passwords. Conversely, MTopDiv consistently yields low values, indicating that the models effectively avoid common mode failures, such as dropping, collapse, and invention.

5.7.2. Length Distribution in Generated Passwords. Figure 9 compares the length distribution of generated passwords

TABLE 9: Comparison summary of the selected approaches.

Metric	PassGAN	PLR-GAN	PassFlow	PassGPT	VGPT2	FLA
Overall Performance						
Scenario: In-Distribution \uparrow	8.85	14.88	12.76	37.21	11.79	49.21
Scenario: Cross-Community \uparrow	3.62	6.16	7.13	17.72	4.76	27.72
Scenario: Cross-Culture \uparrow	6.52	9.70	16.17	14.24	7.01	23.27
Performance by Frequency						
Frequency: Common (Top 5%) \uparrow	27.87	36.86	31.20	68.20	32.83	83.37
Frequency: Common (Top 10%) \uparrow	22.73	30.63	25.73	63.45	27.23	76.60
Frequency: Rare (Bottom 90%) \uparrow	8.75	13.61	12.28	34.51	10.83	47.03
Performance by Length						
Length: Short (4-7 Chars) \uparrow	19.91	32.96	28.16	55.62	29.65	72.97
Length: Medium (8-10 Chars) \uparrow	2.48	4.47	1.33	26.93	2.16	36.38
Length: Long (11-12 Chars) \uparrow	0.07	0.24	0.00	12.58	0.05	11.6
Performance by Pattern						
Pattern: Simple (1 Char Class) \uparrow	14.06	21.71	16.22	41.16	17.85	53.52
Pattern: Moderate (2 Char Classes) \uparrow	3.46	7.15	4.71	29.56	5.26	38.94
Pattern: Complex (3 Char Classes) \uparrow	0.08	0.27	0.05	11.19	0.33	14.20
Generalizability						
Train Set Size Sensitivity (%) \downarrow	12.56	3.54	1.90	13.16	35.49	7.58
Cross-Community Loss (%) \downarrow	24.16	32.42	6.84	42.15	32.49	30.13
Cross-Culture Loss (%) \downarrow	48.67	49.27	14.58	66.15	57.11	58.53
Generated Password Quality						
Uniqueness (%) \uparrow	54.59	71.13	90.10	73.22	91.41	100.00
Humanness Distance (%) \downarrow	19.83	4.80	70.16	2.40	44.00	38.50

with that of real passwords. Among the models, PassFlow, FLA, and VGPT2 show the largest divergence from the real distribution: PassFlow tends to generate shorter passwords, FLA overestimates the probability of 8-character passwords, and VGPT2 of longer ones. In contrast, PassGAN, PLR-GAN, and PassGPT produce distributions that more closely align with real data, reinforcing previous observations.

5.7.3. Why Does IMD Yield High Values?. Figure 9 reveals a clear correlation between length distribution and IMD values. PassFlow, which exhibits the largest deviation from the real distribution, also records the highest IMD score. A similar trend is observed for FLA and VGPT2. Conversely, models such as PassGAN, PLR-GAN, and PassGPT, which more closely replicate the real-world length distribution, achieve lower IMD scores. Interestingly, Table 8 shows that models such as PassFlow and FLA yield higher IMD scores than those associated with the random-password soft upper bound. We posit that this counterintuitive result stems from the uniform length distribution of random passwords. Although this distribution diverges significantly from that of real passwords, it nonetheless includes longer passwords that PassFlow and FLA struggle to generate. These observations suggest that IMD is highly sensitive to length distribution, favoring models that replicate it more accurately. For an extended analysis of IMD, we refer to Appendix B.

5.8. General Benchmarking

This section presents a comprehensive benchmark of the selected generative models, leveraging prior evaluations to facilitate direct comparisons across key metrics: performance, generalizability, and generated password quality. Table 9 summarizes all aggregated results, reflecting the trends observed in the preceding sections. Regarding performance metrics, computed as the weighted average percentage of guessed passwords, FLA consistently achieves the best results across all tasks, with PassGPT being its sole direct competitor. The remaining models exhibit greater task dependency: for instance, PLR-GAN leads in in-distribution scenarios, while PassFlow excels in cross-community and cross-culture set-

tings. However, when evaluating generalizability, the overall ranking shifts. We consider three metrics: sensitivity to training set size, measured using the coefficient of variation, and performance loss in cross-community and cross-culture scenarios, quantified as weighted average loss relative to the in-distribution setting. PassFlow consistently emerges as the most robust model across generalization tasks. Surprisingly, FLA and PassGPT exhibit the highest loss in the cross-cultural setting, with PassGPT additionally demonstrating the lowest robustness in the cross-community scenario, whereas FLA maintains an average value. Finally, concerning the quality metrics, we analyzed both uniqueness and humanness of generated passwords. While not previously emphasized in our analysis, uniqueness is a commonly adopted metric for evaluating the diversity of generated passwords. While FLA’s generation approach guarantees 100% uniqueness by design, PassFlow, leveraging the GS technique, and VGPT2 produce around 90% unique passwords. PLR-GAN and PassGPT achieve slightly above 70% uniqueness, and PassGAN demonstrates the lowest uniqueness at 54.59%. Regarding humanness distance, measured as the average percentage distance between generated and real passwords, results align with our previous observations: models such as PassGPT and PLR-GAN generate passwords closely resembling human-created ones, making them highly suitable for applications like honeywords, where it is important to deceive adversaries using realistic passwords, and as potential sources of synthetic data for future research, thereby addressing challenges with real-world password datasets (e.g., difficult to obtain, outdated, ethical concerns).

6. Insights and Lessons Learned

This section outlines key insights obtained in our study.

Sequential Architectures Perform Best. FLA and PassGPT, based on LSTM and GPT2, consistently outperform other architectures, underscoring the advantages of sequential models in capturing dependencies and generating more accurate, complex guesses.

Generative Models Surpass Traditional Tools. Early generative approaches—such as PassGAN, PassFlow, and PLR-GAN—struggles to match the performance of traditional tools. However, the emergence of transformer-based architectures has shifted the research landscape toward models capable of consistently outperforming them. PassGPT, the most recent approach evaluated in our study, generally surpasses JtR, Hashcat, PCFG, and OMEN. As research progresses, more advanced LLMs are likely to further improve password-guessing effectiveness, whereas traditional tools have likely reached their performance ceiling.

Stricter Policies Mean Safer Passwords. Our analysis reveals that each dataset exhibits a distinct distribution, with some being significantly easier to guess than others. Notably, 000webhost and LinkedIn—both of which follow stricter password policies regarding length and complexity—emerge

as the most challenging datasets. These findings reinforce the conventional understanding that even modestly strict password policies enhance security—an effect that holds true even in the context of generative models.

Guessing Complex/Long Passwords Remains Challenging.

Four out of six models struggle to guess complex passwords—those containing special characters or multiple character classes—and longer passwords exceeding eight characters. Only FLA and PassGPT achieve a non-negligible success rate in these cases, highlighting the persistent challenge posed by long and complex passwords and revealing model limitations.

Rare Does Not Mean Hard to Guess. Rare passwords, such as those appearing only once in the dataset, are not necessarily hard to guess, especially if they are semantically similar to common passwords.

Models Go Beyond Memorization. While performance generally declines relative to in-distribution settings, models still generalize well to unseen distributions, indicating they go beyond simple memorization. Additionally, some models exhibited greater robustness to distribution shifts, even if they performed worse under ideal, in-distribution settings.

Models Generate and Match Distinct Passwords. Even when trained on the same dataset, models generate and match distinct sets of passwords. This diversity enables multi-model attacks that outperform the best individual model.

Models Generate Human-Like Passwords. Generative models effectively learn to generate passwords that closely resemble human-created ones, making them valuable for a variety of tasks beyond just password-guessing.

Beyond Guessing Rate. Prior research primarily assesses password-guessing tools by focusing on guessing rate, thus providing a narrow view of their capabilities. We argue that the emergence of generative password-guessing requires a broader scope, including additional key metrics to assess the quality of the generated data (e.g., humanness, uniqueness) and the generalizability of the models (e.g., train set size sensitivity, cross-dataset loss). Together with guessing performance, these metrics provide a more comprehensive evaluation and characterization of generative models.

7. Conclusion

This work presented MAYA, a framework designed to comprehensively evaluate password-guessing approaches. It includes standardized guidelines and advanced testing scenarios to ensure fair, in-depth comparisons and reveal the strengths and limitations of different models. By analyzing eight real-world password leaks and thoroughly evaluating various password-guessing approaches, we addressed seven key research questions, offering insights into the factors influencing user password choices and the current state of generative password-guessing research. We believe future

research could greatly benefit from MAYA, as it can accelerate the development of new approaches beyond password-guessing, such as enhancing password security mechanisms like honeywords and password strength meters. MAYA is intended to foster academic advancements in password security, not to facilitate or promote malicious activity.

References

- [1] "Mail.ru," *CyberInsurance*, 2014, accessed: Apr. 14, 2025. [Online]. Available: <https://www.cyberinsurance.com/breaches/mailru/>
- [2] "2012 linkedin breach had 117 million emails and passwords stolen, not 6.5m," *Trend Micro*, 2016, accessed: Apr. 14, 2025. [Online]. Available: <https://www.trendmicro.com/vinfo/us/security/news/cyber-attacks/2012-linkedin-breach-117-million-emails-and-passwords-stolen-not-6-5m>
- [3] Y. Abdrabou, J. Schütte, A. Shams, K. Pfeuffer, D. Buschek, M. Khamis, and F. Alt, "“ your eyes tell you have used this password before”: Identifying password reuse from gaze and keystroke dynamics," in *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, 2022, pp. 1–16.
- [4] A. Alaa, B. Van Breugel, E. S. Saveliev, and M. Van Der Schaar, "How faithful is your synthetic data? sample-level metrics for evaluating and auditing generative models," in *International Conference on Machine Learning*. PMLR, 2022, pp. 290–306.
- [5] 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.
- [6] S. Barannikov, I. Trofimov, G. Sotnikov, E. Trimbach, A. Korotin, A. Filippov, and E. Burnaev, "Manifold topology divergence: a framework for comparing data manifolds," *Advances in neural information processing systems*, vol. 34, pp. 7294–7305, 2021.
- [7] D. Biesner, K. Cvejosi, and R. Sifa, "Combining variational autoencoders and transformer language models for improved password generation," in *Proceedings of the 17th International Conference on Availability, Reliability and Security*, 2022, pp. 1–6.
- [8] J. Bonneau, "The science of guessing: Analyzing an anonymized corpus of 70 million passwords," in *2012 IEEE Symposium on Security and Privacy*, 2012, pp. 538–552.
- [9] J. Bonneau, C. Herley, P. C. Van Oorschot, and F. Stajano, "The quest to replace passwords: A framework for comparative evaluation of web authentication schemes," in *2012 IEEE symposium on security and privacy*. IEEE, 2012, pp. 553–567.
- [10] A. Borji, "Pros and cons of gan evaluation measures: New developments," *Computer Vision and Image Understanding*, vol. 215, p. 103329, 2022.
- [11] T. Brewster, "13 million passwords appear to have leaked from this free web host - updated," *Forbes*, 2015, accessed: Apr. 14, 2025. [Online]. Available: <https://www.forbes.com/sites/thomasbrewster/2015/10/28/000webhost-database-leak/>
- [12] H. Cheng, W. Li, P. Wang, and K. Liang, "Improved probabilistic context-free grammars for passwords using word extraction," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 2690–2694.
- [13] 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.
- [14] M. Dell’Amico, P. Michiardi, and Y. Roudier, "Password strength: An empirical analysis," in *Proceedings of the 2010 IEEE INFOCOM*. IEEE, 2010, pp. 1–9.
- [15] 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 7*. Springer, 2015, pp. 119–132.
- [16] S. Furnell, "Assessing password guidance and enforcement on leading websites," *Computer Fraud & Security*, vol. 2011, no. 12, pp. 10–18, 2011.
- [17] R. Gagliardi, "Libero.it password leak - an analysis in-depth," *Scip AG*, 2016, accessed: Apr. 14, 2025. [Online]. Available: <https://www.scip.ch/en/?labs.20180913>
- [18] X. Gan, D. Li, and H. Chen, "Analysis of words in passwords from three different countries," in *2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, vol. 10, 2022, pp. 1775–1781.
- [19] M. Golla, M. Wei, J. Hainline, L. Filipe, M. Dürmuth, E. Redmiles, and B. Ur, "“ what was that site doing with my facebook password?” designing password-reuse notifications," in *Proceedings of the 2018 acm sigsac conference on computer and communications security*, 2018, pp. 1549–1566.
- [20] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," *Advances in neural information processing systems*, vol. 30, 2017.
- [21] I. Gulrajani, C. Raffel, and L. Metz, "Towards gan benchmarks which require generalization," 2020. [Online]. Available: <https://arxiv.org/abs/2001.03653>
- [22] X. Guo, Y. Liu, K. Tan, W. Mao, M. Jin, and H. Lu, "Dynamic markov model: Password guessing using probability adjustment method," *Applied Sciences*, vol. 11, no. 10, p. 4607, 2021.
- [23] 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.
- [24] 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, 2020.
- [25] hashcat. [Online]. Available: <https://hashcat.net/hashcat/>
- [26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2016.
- [27] X. He, H. Cheng, J. Xie, P. Wang, and K. Liang, "Passtrans: An improved password reuse model based on transformer," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 3044–3048.
- [28] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017.
- [29] B. Hitaj, P. Gasti, G. Ateniese, and F. Perez-Cruz, "Passgan: A deep learning approach for password guessing," in *Applied Cryptography and Network Security: 17th International Conference, ACNS 2019, Bogota, Colombia, June 5–7, 2019, Proceedings 17*. Springer, 2019, pp. 217–237.
- [30] S. Jin and M. Dupuis, "Password usage behavior of online users," in *2024 Cyber Awareness and Research Symposium (CARS)*. IEEE, 2024, pp. 1–6.
- [31] P. G. Kelley, S. Komanduri, M. L. Mazurek, R. Shay, T. Vidas, L. Bauer, N. Christin, L. F. Cranor, and J. Lopez, "Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms," in *2012 IEEE symposium on security and privacy*. IEEE, 2012, pp. 523–537.
- [32] S. Komanduri, R. Shay, P. G. Kelley, M. L. Mazurek, L. Bauer, N. Christin, L. F. Cranor, and S. Egelman, "Of passwords and people: measuring the effect of password-composition policies," in *Proceedings of the sigchi conference on human factors in computing systems*, 2011, pp. 2595–2604.
- [33] B. T. Kuhn and C. Garrison, "A survey of passwords from 2007 to 2009," in *2009 Information Security Curriculum Development Conference*, ser. InfoSecCD '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 91–94. [Online]. Available: <https://doi.org/10.1145/1940976.1940994>

- [34] lakiw, “pcfg-cracker,” accessed: Apr. 11, 2025. [Online]. Available: https://github.com/lakiw/pcfg_cracker
- [35] 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.
- [36] Z. Li, W. Han, and W. Xu, “A {Large-Scale} empirical analysis of chinese web passwords,” in *23rd USENIX Security Symposium (USENIX Security 14)*, 2014, pp. 559–574.
- [37] S. G. Lyastani, M. Schilling, M. Neumayr, M. Backes, and S. Bugiel, “Is fido2 the kingslayer of user authentication? a comparative usability study of fido2 passwordless authentication,” in *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2020, pp. 268–285.
- [38] J. Ma, W. Yang, M. Luo, and N. Li, “A study of probabilistic password models,” in *2014 IEEE Symposium on Security and Privacy*. IEEE, 2014, pp. 689–704.
- [39] P. Mayer, J. Kirchner, and M. Volkamer, “A second look at password composition policies in the wild: Comparing samples from 2010 and 2016,” in *Thirteenth Symposium on Usable Privacy and Security (SOUPS 2017)*. Santa Clara, CA: USENIX Association, Jul. 2017, pp. 13–28. [Online]. Available: <https://www.usenix.org/conference/soups2017/technical-sessions/presentation/mayer>
- [40] 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.
- [41] J. Pagliery, “5 million gmail passwords leaked,” *CNN*, 2014, accessed: Apr. 14, 2025. [Online]. Available: <https://money.cnn.com/2014/09/10/technology/security/gmail-hack/index.html>
- [42] G. Pagnotta, D. Hitaj, F. De Gaspari, and L. V. Mancini, “Passflow: guessing passwords with generative flows,” in *2022 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2022, pp. 251–262.
- [43] B. Pal, T. Daniel, R. Chatterjee, and T. Ristenpart, “Beyond credential stuffing: Password similarity models using neural networks,” in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 417–434.
- [44] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, “Normalizing flows for probabilistic modeling and inference,” *Journal of Machine Learning Research*, vol. 22, no. 57, pp. 1–64, 2021.
- [45] D. Pasquini, A. Gangwal, G. Ateniese, M. Bernaschi, and M. Conti, “Improving password guessing via representation learning,” in *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2021, pp. 1382–1399.
- [46] S. Pearman, J. Thomas, P. E. Naeini, H. Habib, L. Bauer, N. Christin, L. F. Cranor, S. Egelman, and A. Forget, “Let’s go in for a closer look: Observing passwords in their natural habitat,” in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 2017*, pp. 295–310.
- [47] A. Radford, “Unsupervised representation learning with deep convolutional generative adversarial networks,” *arXiv preprint arXiv:1511.06434*, 2015.
- [48] J. Rando, F. Perez-Cruz, and B. Hitaj, “Passgpt: Password modeling and (guided) generation with large language models,” in *European Symposium on Research in Computer Security*. Springer, 2023, pp. 164–183.
- [49] J. T. Ripper. [Online]. Available: <https://www.openwall.com/john/>
- [50] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved techniques for training gans,” *Advances in neural information processing systems*, vol. 29, 2016.
- [51] J. Schofield, “32.6m passwords may have been compromised in rockyou hack,” *The Guardian*, 2009, accessed: Apr. 14, 2025. [Online]. Available: <https://www.theguardian.com/technology/blog/2009/dec/15/rockyou-hacked-passwords>
- [52] R. Shay, S. Komanduri, A. L. Durity, P. Huh, M. L. Mazurek, S. M. Segreti, B. Ur, L. Bauer, N. Christin, and L. F. Cranor, “Designing password policies for strength and usability,” *ACM Transactions on Information and System Security (TISSEC)*, vol. 18, no. 4, pp. 1–34, 2016.
- [53] R. Shay, S. Komanduri, P. G. Kelley, P. G. Leon, M. L. Mazurek, L. Bauer, N. Christin, and L. F. Cranor, “Encountering stronger password requirements: user attitudes and behaviors,” in *Proceedings of the sixth symposium on usable privacy and security*, 2010, pp. 1–20.
- [54] E. Stobert and R. Biddle, “The password life cycle,” *ACM Transactions on Privacy and Security (TOPS)*, vol. 21, no. 3, pp. 1–32, 2018.
- [55] X. Su, X. Zhu, Y. Li, Y. Li, C. Chen, and P. Esteves-Veríssimo, “Pagpassgpt: Pattern guided password guessing via generative pre-trained transformer,” in *2024 54th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2024, pp. 429–442.
- [56] T. Team, “What happened in the taobao data breach?” *Twingate*, 2024, accessed: Apr. 14, 2025. [Online]. Available: <https://www.twingate.com/blog/tips/taobao-data-breach>
- [57] A. Tsitsulin, M. Munkhoeva, D. Mottin, P. Karras, A. Bronstein, I. Oseledets, and E. Müller, “The shape of data: Intrinsic distance for data distributions,” 2020. [Online]. Available: <https://arxiv.org/abs/1905.11141>
- [58] 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.
- [59] D. Wang and P. Wang, “On the implications of zipf’s law in passwords,” in *European Symposium on Research in Computer Security*. Springer, 2016, pp. 111–131.
- [60] D. Wang, Y. Zou, Y.-A. Xiao, S. Ma, and X. Chen, “{Pass2Edit}: A {Multi-Step} generative model for guessing edited passwords,” in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 983–1000.
- [61] D. Wang, Y. Zou, Z. Zhang, and K. Xiu, “Password guessing using random forest,” in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 965–982.
- [62] J. Xie, H. Cheng, R. Zhu, P. Wang, and K. Liang, “Wordmarkov: A new password probability model of semantics,” in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 3034–3038.
- [63] M. Xu, C. Wang, J. Yu, J. Zhang, K. Zhang, and W. Han, “Chunk-level password guessing: Towards modeling refined password composition representations,” in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 5–20.
- [64] 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 *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 1001–1018.
- [65] K. Yang, X. Hu, Q. Zhang, J. Wei, and W. Liu, “Vaepass: A lightweight passwords guessing model based on variational auto-encoder,” *Computers & Security*, vol. 114, p. 102587, 03 2022.
- [66] K. Zetter, “Hackers finally post stolen ashley madison data,” *Wired*, 2015, accessed: Apr. 14, 2025. [Online]. Available: <https://www.wired.com/2015/08/happened-hackers-posted-stolen-ashley-madison-data/>

Ethical Considerations

In line with prior research [40], [45], [48], [64], we consider the use of leaked datasets to be ethical, as: (1) they are publicly available, (2) their usage does not cause additional harm, (3) we do not use any additional sensitive information, such as email addresses, phone numbers, or usernames, that

could link specific passwords to individual users, and (4) such data are essential for advancing research. We discourage any usage of MAYA for illegal or unethical purposes. The framework is developed exclusively for academic research to drive advancements in password security.

Appendix A. Models and Tools Implementation

This appendix provides additional implementation details on all models. Each generative model was ported to PyTorch 2.6.0, standardizing dependencies and ensuring reproducibility. All training parameters used were as specified by the original papers/implementation, unless otherwise stated.

FLA. We implemented FLA following the description of the large model mentioned in [40]. Specifically, the architecture consists of three LSTM layers with 1000 cells each, followed by two FC layers. Each checkpoint was trained for 20 epochs, processing input backwards. Unlike other generative approaches, FLA requires a probability threshold as input, filtering out passwords whose overall probability falls below it. We set this threshold to 10^{-8} for up to 10^6 generated passwords, 10^{-9} for 10^7 , and 10^{-10} for 5×10^8 . After generation, we sort guesses by probability in descending order and select the first n ones, equivalent to finding the optimal threshold to generate exactly n passwords.

PassGAN. We implemented PassGAN as described in the original work [29]: using an IWGAN [20] with both the generator and discriminator composed of 5 residual blocks [26]. Following the original setup, we trained PassGAN’s models for 200,000 iterations, evaluating checkpoints every 10,000 steps.

PLR-GAN. PLR-GAN [45] is an enhanced version of PassGAN aimed at improving training instability. PLR injects low-magnitude noise into the one-hot character encodings during training, enabling the integration of deeper architectures and longer training. While the original work states that PLR can be trained up to 4 million iterations, we capped our training at 400,000 iterations due to computational constraints and the diminishing returns observed beyond this point. All our experiments utilize the Dynamic Password Guessing strategy proposed by the authors.

PassFlow. We optimized the original PassFlow implementation [42] to improve efficiency. We introduced an early-stopping mechanism that halts the training if the model fails to improve its performance by at least 5% over the current best result for 10 consecutive epochs, starting from epoch 100. Otherwise, training continues for 200 epochs. All our experiments were conducted using PassFlow’s Gaussian Smoothing technique. The original GS implementation adds noise to the generated passwords until a unique one is generated. However, this approach is highly inefficient on certain datasets, significantly slowing down generation (up to weeks of time for a single run). We modified the GS

algorithm, introducing an early stopping mechanism if no unique password is generated after 100 iterations.

VGPT2. We implemented VGPT2 [7] without introducing any architectural or training modifications.

PassGPT. We used the original PassGPT implementation, applying minimal changes to ensure compatibility with our framework. Following the authors’ recommendations [48], we trained PassGPT on a deduplicated dataset.

OMEN. We used the publicly available implementation of OMEN [15] from [34]. We configured it as a 4-gram Markov model with a coverage set to 0, thereby relying exclusively on guesses from the Markov attack.

PCFG. We used the publicly available implementation of PCFG from [34]. We set the coverage parameter to 1, effectively relying solely on guesses produced by the PCFG model.

Appendix B. Additional Details On RQs

B.1. Additional Details On RQ6

To address RQ6, we employed two metrics: the Jaccard Index and the Mergeability Index. We now present their mathematical definitions:

Jaccard Index: Given two models m_1 and m_2 and a set of datasets $D = \{D_1, D_2, \dots, D_z\}$, let $f(m_i, D_j^{train}, s) = P_{i,j}$ denote the set of passwords generated by model m_i after being trained on dataset D_j^{train} , using settings s . We define the Jaccard Index between m_1 and m_2 as:

$$J(m_1, m_2, D) = \frac{1}{|D|} \sum_{j=1}^{|D|} \frac{|(P_{1,j} \cap P_{2,j})|}{|(P_{1,j} \cup P_{2,j})|} \quad (1)$$

$$\text{where: } P_{i,j} = f(m_i, D_j^{train}, s)$$

Mergeability Index: Let $G_{i,j} = P_{i,j} \cap D_j^{test}$ denote the set of passwords generated by model m_i , after being trained on D_j^{train} , that match those in the corresponding testing set D_j^{test} . The Mergeability Index is defined as follows:

$$MI(m_1, m_2, D) = \frac{1}{|D|} \sum_{j=1}^{|D|} \left(\frac{|(G_{1,j} \cup G_{2,j})| - G_{MAX}}{G_{MAX}} \right) \quad (2)$$

$$\text{where: } G_{MAX} = \max(|G_{1,j}|, |G_{2,j}|).$$

TABLE 10: Distribution of password patterns from r1 to r19.

Dataset	r1	r2	r3	r4	r5	r6	r7	r8	r9	r10	r11	r12	r13	r14	r15	r16	r17	r18	r19
rockyou	44.28	41.89	1.51	15.93	0.02	36.26	1.64	0.15	1.67	31.64	1.11	0.67	0.10	16.54	0.04	0.17	0.07	0.47	9.38
linkedin	20.38	18.02	0.78	19.59	0.01	53.22	1.28	0.16	5.36	43.10	1.97	0.66	0.25	21.23	0.07	0.35	0.31	0.82	10.02
mailru	27.19	24.37	0.27	18.54	0.00	52.19	0.62	0.35	1.12	27.00	0.14	0.37	0.16	21.23	0.03	0.04	0.06	0.03	5.75
000web	0.42	0.18	0.01	0.04	0.02	92.89	1.20	0.40	4.97	69.11	1.77	0.66	0.50	3.31	0.00	0.02	0.02	0.43	13.10
taobao	15.70	15.42	0.13	27.90	0.01	55.89	0.09	0.10	0.31	50.18	0.06	0.04	0.08	28.46	0.00	0.01	0.01	0.01	9.92
gmail	39.78	39.78	0.00	15.70	0.02	42.38	0.86	0.14	1.13	33.36	0.53	0.58	0.09	16.93	0.12	0.07	0.17	0.00	9.05
ashley	35.04	33.18	0.98	12.32	0.00	52.24	0.12	0.02	0.27	41.79	0.11	1.05	0.01	13.43	0.00	0.01	0.01	0.05	10.11
libero	42.04	39.01	1.86	13.10	0.00	42.41	0.63	0.08	1.69	34.06	0.75	0.24	0.09	15.94	0.02	0.09	0.05	0.24	6.95

TABLE 11: Top 10 passwords for each dataset. The password at the 8th position in 000webhost (indicated as “*”) is “YfDbUfNjH10305070”: the letter portion of the password can be mapped to a Russian word meaning “Navigator”. The reasons for its unexpected popularity remain unclear [59].

Dataset	1	2	3	4	5	6	7	8	9	10
Rockyou	123456	12345	123456789	password	iloveyou	princess	rockyou	1234567	12345678	abc123
LinkedIn	linkedin	123456	123456789	abc123	idontknow	ilovelinkedin	Godisgood	jaimatadi	linkedin1	iloveindia
Mailru	qwerty	qwertyuiop	123456	qwe123	qweqwe	klaster	1qaz2wsx	1q2w3e4r	qazwsx	1q2w3e
000webhost	abc123	123456a	12qw23we	123abc	a123456	123qwe	secret666	*	asd123	qwerty123
Taobao	123456	1111111	123456789	123123	000000	5201314	wangyut2	123	123321	12345678
Gmail	123456	password	123456789	12345	qwerty	12345678	1111111	abc123	123123	1234567
Ashley M.	eatpussy	opensaysme	christina	longing	nastygirl	steve	11inches	2ofus	69sex	99wmp
Libero	123456	popopo90	francesco	123456789	12345678	napoli	alessandro	amoremio	andrea	francesca

TABLE 12: Marginal gain in successful matches as the number of generated passwords increases, with each column representing generation intervals (e.g., 1M-2.5M indicates the marginal gain when expanding from 1M to 2.5M guesses). For a given interval X-Y, *Total* indicates the marginal gain achieved when increasing the number of guesses from X to Y, expressed as a percentage of the total number of test passwords. *Relative* indicates the marginal gain relative to the number of matches obtained at X. Results are averaged across all password lengths.

Model	1M-2.5M		2.5M-5M		5M-7.5M		7.5M-10M		10M-25M		25M-50M		50M-75M		75M-100M		100M-250M		250M-500M	
	Total	Relative	Total	Relative	Total	Relative	Total	Relative	Total	Relative	Total	Relative	Total	Relative	Total	Relative	Total	Relative	Total	Relative
PassGAN	+0.49	+83.75	+0.55	+50.94	+0.40	+24.84	+0.32	+15.95	+1.23	+53.74	+1.14	+32.85	+0.75	+16.47	+0.56	+10.69	+1.96	+34.15	+1.63	+21.38
PLR-GAN	+0.62	+97.50	+0.74	+59.45	+0.57	+28.94	+0.48	+18.91	+1.94	+65.41	+1.96	+40.34	+1.34	+19.54	+1.03	+12.55	+3.67	+40.28	+3.21	+25.59
PassFlow	+0.62	+69.67	+0.73	+48.88	+0.56	+25.15	+0.44	+15.72	+1.72	+52.72	+1.58	+31.53	+1.05	+15.85	+0.80	+10.38	+2.84	+33.66	+2.48	+21.97
PassGPT	+1.20	+114.72	+1.60	+71.91	+1.34	+35.32	+1.18	+22.97	+5.21	+83.58	+5.45	+48.49	+3.66	+22.15	+2.73	+13.56	+8.89	+39.13	+6.53	+20.63
VGPT2	+0.18	+137.88	+0.28	+90.08	+0.25	+43.08	+0.24	+28.34	+1.18	+110.51	+1.45	+65.38	+1.11	+30.52	+0.92	+19.44	+3.65	+65.02	+3.43	+37.03
FLA	+4.33	+58.54	+4.17	+35.55	+2.66	+16.71	+1.93	+10.40	+6.57	+32.14	+5.55	+20.62	+3.33	+10.28	+2.34	+6.56	+7.44	+19.62	+5.84	+13.00

B.2. Additional Details On RQ7

Let $D^{train} = \bigcup_{j=1}^{|D|} D_j^{train}$ and $D^{test} = \bigcup_{j=1}^{|D|} D_j^{test}$ denote the union of all training and testing datasets, respectively. Let R be the set of randomly generated passwords, each between 6 and 12 characters in length. The lower-bound baseline L and upper-bound baseline U for a given metric $dist$ are computed as follows:

$$L = dist(D^{test}, D^{train}), \quad U = dist(D^{test}, R) \quad (3)$$

Next, for each model m_i , let P_i be the set of passwords generated by model m_i across all datasets. The value d_i for a model m_i on a given metric $dist$ is then computed using the following equation:

$$d_i = \frac{dist(D^{test}, P_i) - L}{U - L} * 100 \quad (4)$$

We now provide further details regarding the four selected metrics, followed by an additional analysis on IMD.

CNN Divergence [21]. Neural networks can be used to estimate the divergence between two distributions, making them useful for evaluating generative models. The idea is to employ an independent critic network trained to distinguish between real and generated samples. After sufficient training, the critic’s loss, based on WGAN-GP [20], reflects the distance between the two distributions. Our convolutional neural network (CNN) follows the architecture and settings described in [21], based on the DCGAN discriminator [47].

IMD [57]. IMD, which stands for Intrinsic Multi-scale Distance, is a metric designed to compare the data manifolds of two distributions. IMD provides an intrinsic method to lower-bound the spectral Gromov-Wasserstein distance between two manifolds. Unlike other approaches that focus on extrinsic properties and are uni-scale, IMD is intrinsic, meaning it is not dependent on the transformation of the manifold and multi-scale, capturing both local and global properties.

α -Precision β -Recall Authenticity [4]. The α -Precision β -Recall Authenticity is a three-dimensional metric, with each dimension corresponding to a distinct property:

TABLE 13: IMD outputs as the distributions P and Q vary.

IMD(P,Q)	Output
P = 000webhost - Q = Random	7.9919
P = 000webhost - Q = PassFlow 000webhost	47.0466
P = PassFlow 000webhost - Q = Random	49.0287
P = PassFlow 000w. - Q = Rand with PassFlow 000w. Length	30.9260

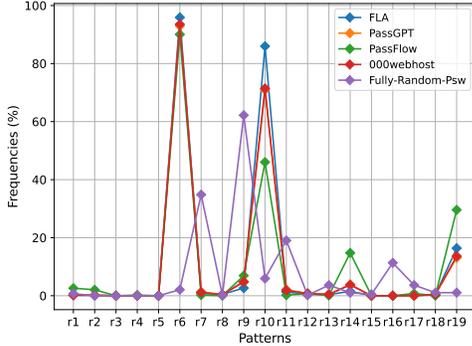


Figure 10: Distribution of 19 patterns in the 000webhost test dataset and in passwords generated by FLA, PassGPT, and PassFlow, all trained on 000webhost.

- **Fidelity (α -Precision):** Measures how closely the generated samples resemble the most typical fraction (α -support) of real data.
- **Diversity (β -Recall):** Represents the fraction of real samples that lie within the β -support of the generated data distribution.
- **Generalization (Authenticity):** Reflects the model’s ability to generalize, ensuring that the output is not limited to mere copies of the training data.

Each dimension is computed after mapping generated and real data into a hypersphere using a feature embedding, where most of the data is concentrated near the center, while outliers are positioned closer to the boundaries.

MTopDiv [6]. MTopDiv introduced a method that tracks multiscale topology discrepancies between two distributions, P and Q , in a high-dimensional space. Unlike IMD, MTopDiv also considers extrinsic properties, using position and translation to capture structural differences between the two distributions.

IMD Analysis. To better understand the rationale behind the high values obtained with IMD across several models, we repeat the metric analysis using 10^7 generated passwords. The results, shown in Table 14, differ significantly from those obtained with 5×10^8 generated passwords (Table 8), particularly for PassFlow, PLR-GAN, and FLA, which exhibit notably higher IMD scores in the smaller sample. To investigate this discrepancy, we compare the length distributions of the 10^7 and 5×10^8 generated samples. We observe that, in the smaller set, models such as PassFlow and PLR-GAN—based on GS and DPG techniques, respectively—exhibit a strong

bias toward shorter passwords. This behavior stems from the TABLE 14: Distance between human- and generative model-created passwords when generating 10^7 passwords.

Models	CNN Div	α -Precision	β -Recall	Auth	IMD	MTopDiv
PassGAN	16%	38%	7%	13%	62%	1%
PLR-GAN	9%	-13%	5%	9%	18%	0%
PassFlow	69%	31%	59%	35%	500%	52%
PassGPT	6%	-9%	4%	4%	0%	0%
VGPT2	33%	26%	36%	6%	120%	15%
FLA	24%	39%	3%	48%	347%	2%

tendency of such models to initially saturate the space of simpler (i.e., shorter) passwords before extending to more complex ones. A similar pattern is observed with FLA, which outputs the most probable passwords. Consequently, when fewer samples are generated, the length distribution exhibits a strong bias toward shorter lengths, as shorter passwords are more probable. These findings confirm our hypothesis that IMD is highly sensitive to length distribution differences: the greater the deviation of the generated passwords’ length distribution from the real one, the higher the distance measured by IMD.

We now focus on the 000webhost dataset, which yields a particularly high IMD score for PassFlow when evaluating 10^7 generated passwords. Although this score exceeds that obtained from random passwords, it does not necessarily indicate that PassFlow’s outputs are random or even close to random. To demonstrate this, we compute the IMD metric between PassFlow’s generated passwords and random passwords. The results, shown in Table 13, are compared against the two baselines and reveal that PassFlow’s passwords are farther from random passwords than from real ones. We further examined the effect of aligning the length distribution of the random passwords with that of PassFlow’s 000webhost-generated passwords. As reported in Table 13, this adjustment results in a higher IMD score, thereby confirming that IMD is sensitive to differences in length distribution.

Lastly, we investigate the impact of various degrees of mode failure, such as mode dropping and mode invention, on the IMD metric. For this analysis, we used the 19 patterns listed in Table 3. As shown in Figure 10, the pattern distribution of random passwords deviates substantially from the others. Mode invention is observed in patterns r7, r9, r11, r13, r16, and r17, while mode dropping occurs in commonly observed patterns such as r6, r10, and r19. These findings suggest that the IMD metric does not adequately capture the varying degrees of mode failure and further support the hypothesis that IMD primarily outputs high values due to a model’s inability to replicate the real password length distribution. Interestingly, PassGPT performs particularly well in approximating the pattern distribution of 000webhost, with its pattern distribution closely aligning with that of the real passwords. Additionally, FLA outperforms PassFlow in capturing this distribution, with the latter struggling to replicate patterns r10, r14, and r19. These observations are consistent with the results obtained using the MTopDiv metric in Table 8, which explicitly accounts for such mode-related discrepancies.