Exploring Patterns and Assessing the Security of Pseudorandom Number Generators with Machine Learning

Sara Boancă
Babeş-Bolyai University Cluj-Napoca, Romania

Keywords: Machine Learning, Neural Networks, Pattern Learning, Pseudorandom Number Generators.

Abstract: In recent years, Machine Learning methods have been employed for testing the security of pseudorandom number generators. It is considered that successful learning from pseudorandom data implies the existence of some detectable pattern within it, thus reducing the generator security. As the number and complexity of such approaches has reported important growth, the aim of the present paper is to synthesize current results, discuss perspectives and challenges and provide relevant guidelines for future study. To the best of our knowledge, this is the first comprehensive analysis on the current state of the research into the problem of pseudorandomness exploration by means of Machine Learning.

1 INTRODUCTION

It is known that pseudorandom number generators (PRNGs) produce numbers that, despite appearing indistinguishable from true random numbers and following certain statistical properties, are the result of a deterministic process. Effort has been invested into providing some statistical guideline for evaluating PRNG security by means of statistical test suites such as NIST (Rukhin et al., 2001), TestU01 (L'Ecuyer and Simard, 2007) or Dieharder (Robert G. Brown, 2017) that take into consideration properties like equidistribution, gap and birthday spacings (among many others) to determine whether the generated pseudorandom sequence is uniformly distributed or may be predicted with more than chance accuracy by random guessing. While such tests provided tremendous advances for PRNG evaluation in the past, their ability of understanding patterns is reduced. With the advent of Machine Learning, one may pose the question of whether more than just statistical properties can be extracted from PRNG data.

Thus far, Machine Learning approaches have been used for performing cryptography and cryptanalysis related tasks, such as pseudorandom number generation (Pasqualini and Parton, 2020) and differential cryptanalysis (Gohr, 2019) bridging the gap between theoretical algebra and the innovative possibilities of intelligent algorithms. In the context of PRNG security assessment through pattern exploration, Machine Learning methods have been used sporadically until (Fischer, 2018) formally proposed neural networks for the task. As a consequence, a new area of research has emerged and the number of studies on this matter has significantly increased.

The aim of this paper is to bring to light existing research on PRNG security exploration by means of Machine Learning in an attempt to synthesize current knowledge and results as well as discuss potential challenges and opportunities for the future. As this field is still in its infancy, approaches are included based on the novelty and original perspective they bring to the problem. To the best of our knowledge, this is the first comprehensive study discussing the current state of the research in Machine Learning methods applied for PRNG pattern exploration and security assessment.

The rest of the paper is structured as follows. Section 2 discusses research premises in terms of problem formulation. Section 3 presents the existing approaches of Machine Learning methods applied for PRNG security assessment. Section 4 displays synthesized results. Section 5 proposes a discussion on future work. Section 6 presents the conclusions of the study.

2 PROBLEM FORMULATION

The present section describes existing formulations for the problem of learning from pseudorandom data.

Boancă, S.
Exploring Patterns and Assessing the Security of Pseudorandom Number Generators with Machine Learning.
DOI: 10.5220/0012312900003636

In *Proceedings of the 16th International Conference on Agents and Artificial Intelligence (ICAART 2024) - Volume 3*, pages 186-193
ISBN: 978-989-758-680-4; ISSN: 2184-433X

Over the years, emerging research in the area of pseudorandom pattern exploration by means of Machine Learning has been organized in a number of directions:

- next in sequence prediction;
- entropy estimation;
- inversion;
- sequence randomness classification (strong vs. weak).

Next in sequence prediction may be the most common formulation for the problem of learning from pseudorandom data. It consists of training classification or regression models on pseudorandom number sequences in order to determine, for a given input sequence, which value is most likely to be generated next. The approach is based on the intuition that weak generators may exhibit more easily detectable patterns (see randograms for Linear Congruential Generators (O'neill, 2014)), thus enabling learning and offering a greater possibility of success when predicting future outputs. In this case, the strength of the learning algorithm is measured by its prediction accuracy. However, despite the apparent simplicity of the task, the problem of predicting next PRNG values considering past sequences is a difficult one and in practice PRNG simplfication is often used in order to facilitate learning.

Another emerging direction of research is PRNG *entropy estimation*, often measured in terms of the min-entropy score. Higher entropy values are associated with a higher quantity of information betrayed by the PRNGs. Studies approaching the problem in this way use a next in sequence prediction component to estimate subsequent PRNG values and compute entropy scores based on the obtained results. The method has gained popularity since its introduction in the late 2010s and interest in developing complex Machine Learning models for PRNG entropy estimation is increasing.

PRNG *inversion* by means of Machine Learing is an interesting, yet underdeveloped direction in PRNG security assessment which involves training models to learn form PRNG data to the point of determining their hidden parameter configuration (such as seed, modulus, tap positions or feedback polynomial for LFSRs). Due to its difficulty, there are few attempts to approach the problem in this way, yet some of the cited studies may be considered to have (at least partially) achieved it.

Another under-examined direction is the one of *sequence randomness classification*, which involves training Machine Learning models to differentiate between strong and weak pseudorandom sequences. A strong pseudorandom sequence exhibits high unpredictability, namely it is difficult to find correlations within its data and consequently, to learn from it, while in the case of a weak sequence, such correlations are more apparent. The problem is formulated in (Savicky and Robnik-Šikonja, 2008) and its difficulty is acknowledged by the authors. Like in the case of other PRNG security assessment approaches, no security proof can be emitted for the sequences that pass such tests, but security may be dismantled for those that fail.

The aforementioned problem formulations have been introduced to provide improved understanding of the task in relation to the presented approaches.

3 RELATED WORK

The current section organizes existing research in the field of pseudorandomness exploration and security assessment by means of Machine Learning. The speed with which interest in this field has increased as well as the modernization and complexity of the techniques used are highlighted by the timeline display of the studied approaches.

Early attempts at PRNG prediction with Machine Learning revolve around Decision Trees.

The authors of (Kant and Khan, 2006) predict subsequent bits from a Fibonacci Linear Feedback Shift Register (LFSR). Given an initial state, the LFSR constructs a pseudorandom sequence by performing the exclusive or (XOR) operation on bits located at tap positions. The tap positions of the LFSR can be described by a polynomial modulo 2. The powers of the polynomial denote the positions of the taps to be XOR-ed within the LFSR. The authors frame the next in sequence prediction task as a classification problem and attempt to solve it through the C4.5 algorithm. It is observed that a number of bits equal to the degree of the polynomial is needed as input size in order to outperform random guessing. The model reaches 100% classification accuracy. The authors introduce a Bit Prediction Ratio formula that describes the number of input bits necessary for predicting the subsequent one, which they find increases with the number of taps. Moreover, the rules generated by the C4.5 algorithm can be used to infer the initial feedback polynomial, which may be considered a first attempt to PRNG inversion.

The work of (Savicky and Robnik-Šikonja, 2008) explores patterns in a widely used PRNG, namely Matlab *rand()* in the case the *state* method is used (denoted rand_state). The authors propose a Random Forest (RF) based framework and formulate 4

different approaches for PRNG learning: two involving classification, (following different thresholding), one involving regression and one involving feature importance evaluation. Comparisons are performed with PRNGs such as MINSTD, ANSIC, the rand() method in the Microsoft C compiler (denoted as ms_rand) and RANDU (with odd seeds) as well as Mersenne Twister (MT) based PRNGs such as MRG32k5a and WELL19937a, which describe an upper bound for security comparison. Statistical tests lead to the rejection of the hypothesis according to which rand_state and RANDU values are independent and identically distributed. For regression, it is considered that if the ratio between the current error and the average error of the test set is smaller than 1, the model has successfully learned an approximation. Root Mean Squared Error (RMSE) is used for measurement. Both rand_state and RANDU (with even and odd seeds) fail, while the rest of the generators pass. Dependencies are detected in all PRNGs, but are strongest in rand_state and RANDU.

The work of (Kant et al., 2009) expands on (Kant and Khan, 2006), by employing a variety of Machine Learning techniques, namely Naive Bayes (NB), Average One Dependence Estimators (AODE), C4.5 and Artificial Neural Networks (ANNs), to predict future bits for an Alternating Step and a Geffe LFSR as well as for a number of keystreams in the eSTREAM project. It is observed that the performance of NB is insufficient for the given task and the ANN needs a large number of instances for training. For the Alternating Step and the Geffe LFSR, C4.5 and AODE obtain over 90% accuracy, with C4.5 performing better. It is important to note that previous experiments (Kant and Khan, 2006) were unable to attain high prediction accuracy for the Geffe LFSR. The authors observe that the number of bits needed for prediction increases with the degree of the polynomial and the sparsity of the taps. Studied keystreams could not be predicted with more than chance accuracy.

Following these initial approaches, the literature has seen a shift in focus as approaches began to lean towards the increasingly popular neural networks.

The work of (Hashim and Abdulhussien, 2015) describes an attempt at strong vs. weak PRNG classification with ANNs. While such classification is deemed feasible, the approach involves rather few pseudorandom number sequences. Moreover, since class labels used in training are obtained through statistical test evaluation, one may say the neural networks capacity for learning is reduced to the baseline of statistical tests. The approach was included, however, on the base of its originality and for being, to the best of our knowledge, the first attempt at strong vs. weak PRNG classification.

The work of (Fan and Wang, 2018) studies the ability of an ANN to find correlations in natural sequences such as $\pi, \sqrt{2}$ and e and the MT PRNG. Natural sequences follow a thresholding for binarization. The authors perform statistical testing on the obtained results and claim enough evidence exists for considering that thresholded π digits present a subtle correlation that can be learned by means of an ANN. The approach is extended to a Linear Congruential Generator (LCG) and a Quantum Random Number Generator (QRNG) in the work of (Feng and Hao, 2020). It is observed that slightly more predictable results are obtained for the LCG, while insufficient information can be extracted from other generators.

The study of (Fischer, 2018) is considered an inflection point in the literature since PRNGs to be evaluated are extended to include cryptographically secure PRNGs and learning is performed by advanced Machine Learning techniques such as Long Short-Term Memory Networks (LSTMs), Gated Recurrent Units (GRUs) and their bidirectional variations (BiLSTM, BiGRU). The studied generators are Debian rand, Python random.randint, arc4random, SHA1PRNG and /dev/urandom (with arc4random and SHA1PRNG passing the Dieharder test suite). Training involves a simple data mean fitting step followed by a more complex estimation step based on the networks ability of learning variation. The Mean Average Error (MAE) score is obtained and an evaluation framework based on multiple indicators is built. The first indicator is the computed mean. If the MAE score is lower than the expected average, learning is considered successful. Due to model construction, it is believed that a good model provides low MAE levels and a good variance score in order to prove the model did not simply learn the mean of the training set. Thus, the second indicator is the obtained variance. In the case variance is small, the system has likely overfit the data. The authors describe some variance bounds within which learning is deemed successful. The third indicator is p-value testing. Statistical tests are performed for p-values < 0.01 in order to reject the null hypothesis of the data belonging to an independent and identical distribution. If one indicator raises concerns, the result is unknown. If more indicators suggest the model has learned from the PRNG sequence, the result is fail. The tests pass only if no indicators are able to detect issues.

Following this framework, vulnerabilities are discovered in all studied PRNGs, with higher degree of insecurity for rand and random.randint. The LSTM model is deemed the most suitable for the task.

The paper (Truong et al., 2018) describes an at-

tempt to evaluate the security of a Quantum Random Number Generator (QRNG) by means of min-entropy estimation and, in the process, provides an evaluation for a LCG as well. The authors use a hybrid model consisting of Convolutional and LSTM layers (CNN-LSTM). The problem is approached as a classification in terms of the output layer, where classes represent the corresponding bits of the following number. The authors observe that up to a certain period the model displays better than random performance, but the accuracy declines as the period is increased beyond that threshold.

The work of (Lv et al., 2020) proposes a Recurrent Neural Network (RNN) and an ANN for the task of min-entropy estimation. This comes as a refinement to the traditional approach (Kelsey et al., 2015) that, due to its design, is prone to underestimates. The proposed estimators are used to evaluate a number of sequences, including pseudorandom sequences such as those from random.org, Ubld.it, linux kernel entropy source, linux /dev/urandom and Windows random bit generator with Crypto API. The approach consists of evaluating network performance while still updating model parameters. When compared to the traditional NIST package evaluation, the results obtained by both the RNN and the ANN are considered better. Among the two, RNNs prove superior excepting the linux kernel entropy source, where ANNs provide better estimation. Due to the fact that the generator draws its entropy from the pattern of repetitiveness of user interaction with the operating system and given the nature of ANN learning, this led the authors to believe periodicity is produced in the random number generator.

The authors of (Li et al., 2020) propose a LSTM model with Temporal Pattern Attention (TPA) to learn from a QRNG and a LCG. The problem is approached as in (Truong et al., 2018), with a final classification step for the bits of the following number. Comparisons with models similar to those in (Truong et al., 2018) and (Lv et al., 2020) are provided and the results are evaluated with respect to the length of the input sequence as well. It is observed that both the ANN and the TPA-LSTM are able to detect patterns for sequence lengths as small as 3.2×10^6. While in this case the ability of the ANN is significantly higher than that of the TPA-LSTM, as the sequence grows to 8×10^6 the use of attention is considered influential in elevating the TPA-LSTM model performance to an accuracy of over 95%.

The work of (John Labelle, 2020) explores the deterministic nature of PRNGs and attempts to determine subsequent values generated by the Xorshift128 PRNG using an LSTM. While the developed model exceeds 95% bitwise accuracy, (Mostafa Hassan, 2021) observes that using an ANN for the task can significantly reduce the number of model parameters while maintaining (and even increasing) the accuracy. Thus, an ANN is trained and achieves 100% bitwise accuracy. Moreover, the author displays the values of the learned weights, showing that the model was able to capture essential information in the data in what may be considered an early stage of an inversion attempt. These approaches are considered noteworthy as they discuss the trade off between model complexity, accuracy and domain knowledge in the context of PRNG learning.

The work of (Gupta et al., 2021) addresses the same problem as (Kant and Khan, 2006) for the Fibonacci LFSR. The authors attempt to decrease the number of bits the traditional Berlekamp Massey (BM) algorithm needs for decryption, then train an ANN on the reduced dataset to perform next in sequence prediction. This is possible through the introduction of two pattern generation algorithms. The problem is again framed as a classification task. Training is first conducted in order to find the minimum number of bits needed to obtain sufficient accuracy by gradually decreasing this number and evaluating model performance. As the lower bound on the number of bits is established, the value is retained and the model is retrained with this configuration. Thus, the authors succeed in finding a new lower bound for the number of bits needed for learning to make accurate next in sequence predictions for the LFSR.

The work of (Kim and Kim, 2021) performs a theoretical analysis on the relation between the ANN input size and tap positions of Fibonacci LFSRs in the context of learning from PRNG data, validating their results in practice via a number of simulations. After training on the proposed configuration, model insight is provided through weight visualization, which may be considered a proof for the model ability towards inversion.

The paper (Amigo et al., 2021) approaches the problem of predicting subsequent values for some LCG variations using deep ANNs. The task is framed as a regression, where the next in sequence prediction problem is modeled in a Markovian way (considering only one previous state) and is reduced to the approximation of a piecewise continuous function with a finite number of discontinuities. The authors successfully train the models and compute the Mean Squared Error (MSE) for evaluation. Errors in approximation are found to occur in very few cases, for points close to the discontinuity edge. This is assumed to happen because of the nature of the training process where only one previously generated seed is used and con-

secutive values near discontinuity points lead to prediction results at opposite sides of the codomain. The approach is claimed to be the first to successfully use neural networks for estimating piecewise continuous functions.

4 EVALUATION AND RESULTS

The present section expands on the evaluation methods and results obtained in the studied approaches.

It is important to note the difficulty of training Machine Learning models to learn from PRNG data. There have been cases when certain approaches produced no results (Zanin, 2022). For this reason, any attempt that manages to extract information from such data may be deemed, at least to some degree, successful. Section 3 describes a number of approaches for performing this admittedly difficult task. It is apparent that the focus of the scientific community has been on framing it as either a next in sequence prediction or an entropy estimation problem. While the next in sequence prediction approaches are more numerous, those concerning entropy estimation tend to employ increasingly complex models.

Next in sequence prediction is generally performed through an initial simplification (binarization) of the sequences and consequently regarded as a binary classification problem (Savicky and Robnik-Šikonja, 2008), (Fan and Wang, 2018), (Feng and Hao, 2020). Attempts at performing regression on real valued inputs have been successful, but have required more complex architectures as well as heavier preprocessing (Amigo et al., 2021).

The computational techniques used range from Decision Trees in early approaches to more sophisticated neural network models (ANNs, LSTMs) and the majority focus on PRNGs based on shift registers or LCGs, while there are some that evaluate PRNGs that come with a security claim as well. The generic approach involves computing the next value of the sequence and retaining the corresponding prediction accuracy that would, in the end, be subject to a statistical test (Savicky and Robnik-Šikonja, 2008), (Fan and Wang, 2018), (Feng and Hao, 2020). However, in some cases the values may be considered in ensembles of indicators to obtain the final verdict on the PRNG security (Fischer, 2018).

It is observed that ANNs are preferred for learning on LFSRs and produce better results than their recurrent variations.

For regression tasks, metrics such as MSE (Amigo et al., 2021) or RMSE (Savicky and Robnik-Šikonja, 2008) are used to evaluate the accuracy of the prediction.

Regarding the formulation of the problem as an entropy estimation task, evaluated PRNG generators range from simple LCGs to some that include quantum noise components. It is considered that higher entropy scores signify larger amounts of learned information. For this method, all employed models rely on neural networks and comparisons may be performed not only against a baseline distribution, but

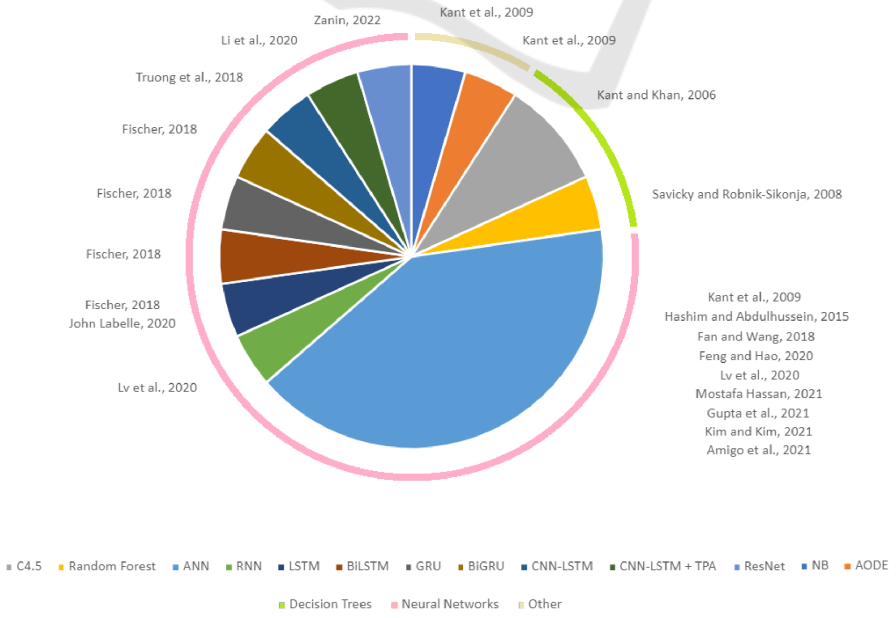

Figure 1: Machine Learning techniques used in the corresponding studies.

Table 1: Overview of the studied Machine Learning approaches for PRNG exploration.

Work	Technique	PRNG	Training dataset size	Results
(Kant and Khan, 2006)	C4.5	Fibonacci LFSR, Geffe LFSR	10^5	100% prediction accuracy, polynomial inference
(Savicky and Robnik-Šikonja, 2008)	RF	rand_state, ms_rand, RANDU, MINSTD, ANSIC, MRG32k5a, WELL19937a	10^4	dependencies in all but MRG32k5a and WELL19937a, worst in rand_state
(Kant et al., 2009)	C4.5, NB (no results), AODE, ANN	alternating step LFSR, Geffe LFSR, eSTREAM	90% of period, 10^6 for eSTREAM	up to 100% prediction accuracy for LFSRs
(Hashim and Abdulhussien, 2015)	ANN	random bit generator	10^3	low MSE score
(Fan and Wang, 2018)	ANN	thresholded natural sequences, MT, QRNG	10^4	higher prediction accuracy for π
(Feng and Hao, 2020)	ANN	thresholded natural sequences, MT, LCG, QRNG	10^6	higher prediction accuracy for π, LCG
(Fischer, 2018)	LSTM, GRU, BiLSTM, BiGRU	random.randint, Debian rand(), arc4rand, SHA1PRNG, /dev/urandom	10^7	vulnerabilities in all but /dev/urandom
(Truong et al., 2018)	CNN-LSTM	LCG, QRNG	10^6	better accuracy for LCG for smaller periods
(Lv et al., 2020)	ANN, RNN	random.org, Ubld.it, linux kernel entropy source, /dev/urandom, Windows RNG	$10^6 - 10^8$	entropy evaluation, possible periodicity in /dev/urandom
(Li et al., 2020)	CNN-LSTM with TPA	LCG, QRNG	10^8	high entropy for LCG for smaller periods
(John Labelle, 2020)	LSTM	Xorshift128	10^6	over 95% bitwise accuracy
(Mostafa Hassan, 2021)	ANN	Xorshift128	10^6	100% bitwise accuracy
(Gupta et al., 2021)	ANN	Fibonacci LFSR	7 to 45% less than BM	up to 100% prediction accuracy
(Kim and Kim, 2021)	ANN	Fibonacci LFSR	10^4	0 Bit Error Rate value
(Amigo et al., 2021)	Deep ANN	variations of a LCG	10^6	good model fit, low MSE score

also against similar approaches. The work of (Li et al., 2020) evaluates model performance with respect to various data sizes as well. Moreover, it provides a comprehensive comparison to min-entropy scores obtained by network architectures similar to those in (Truong et al., 2018) and (Lv et al., 2020).

In terms of training, the above described approaches use datasets of size ranging from $10^3 - 10^5$ in early studies to 10^8 in the later ones. The training time varies from hours to weeks, being generally performed on GPUs.

Table 1 provides a synthesized overview of existing approaches. It covers computational techniques, PRNGs, dataset size and obtained results for each of the discussed works.

Figure 1 displays a distribution of the Machine Learning techniques used the included studies.

5 DISCUSSION AND FUTURE WORK

The present section expands on some of the discussed approaches and presents a number of directions for future study.

The task of training Machine Learning models for PRNG pattern exploration and security assessment is undoubtedly difficult, hence the small number of studies performed on the subject in the past decades. Recent advances in Machine Learning have prompted growing interest in performing such experiments as they reveal not only the complexity of PRNGs, but also that of the models employed for the learning task. For this reason, study in the direction of Machine Learning analysis of loosely correlated data such as that of PRNGs may be regarded as a means of testing (and expanding) the frontier of how much information Machine Learning models are capable to extract. It may be considered that future studies performed in this direction will have an enormous impact when extending their results to other problem domains that share in dealing with subtle correlations (Fan and Wang, 2018). To some extent, this was already achieved in the case of (Kim and Kim, 2021), where the actual application domain of the study was intended to be optical communication systems.

Interest has been invested in employing current PRNG pattern exploration models to verify quantum random number generators as well.

In terms of current results, the studied approaches involve a rather small number of generators mostly belonging to classes of weak PRNGs (Shift Registers, LCGs), yet interesting learning tasks have been formulated for them. The most widely used approach is that of next in sequence prediction, while entropy estimation problems have sparked growing interest by capitalizing on the use of increasingly complex models. Few studies have so far involved inversion and strong vs. weak PRNG classification. Despite the widespread tendency of leaning towards "one-size-fits-all" black box models, given the difficulty of the problem, one may at this stage benefit from developing tailored models with domain knowledge, targeting some precise PRNG. When such approaches are applied, especially in the case of stronger PRNGs, generator simplification may be required to perform learning.

It is believed that successful models can be developed using PRNG simplification and domain knowledge to capture the underlying deterministic processes of stronger generators. It may be assumed that once captured, such deterministic rules can be learned to perfection. It would be interesting to perform such attempts on stronger PRNGs with next in sequence prediction and inversion tasks, especially since the latter has been under-examined until this point.

Different robust neural network models such as Generative Adversarial Networks (GANs) and Transformers, as well as other hybrid model frameworks may be evaluated with respect to this task.

Despite important progress, the time when Machine Learning testing will be performed instead of (or complimentary to) traditional statistical test suite evaluation may still be far. The main reason for this is the complexity in the design of PRNGs regarded from the Machine Learning methods perspective. Another important issue is the high training time which may deem implementation impractical. However, with the advent of GPU and TPU computation this drawback is expected to be mitigated in the future.

It is important to acknowledge the progress attained through the use of Machine Learning for PRNG pattern exploration and security evaluation in the past decades. The number and complexity of approaches to tackle this task is rapidly increasing as is the interest in this emerging niche. While current domain applicability of the developed solutions may be reduced to theoretical exploration, there is reason to believe that in the future, knowledge derived from these studies may actually be transferred to tackle more practical problems.

6 CONCLUSIONS

The present study discusses Machine Learning methods in the context of learning from PRNG data. While existing approaches are able to perform satisfacto-

rily in the case of this admittedly difficult task, improved computational approaches are proposed and may be used in the future. As this field is still in its infancy, more complex studies are expected to be performed covering a wider range of PRNGs and Machine Learning models. The present work is, to the best of our knowledge, the first to highlight and synthesize existing results on PRNG exploration by means of Machine Learning in an attempt to organize knowledge and popularize this emerging niche.

REFERENCES

Amigo, G., Dong, L., and Ii, R. J. M. (2021). Forecasting pseudo random numbers using deep learning. In *2021 15th International Conference on Signal Processing and Communication Systems (ICSPCS)*, pages 1–7. IEEE.

Fan, F. and Wang, G. (2018). Learning from pseudo-randomness with an artificial neural network–does god play pseudo-dice? *IEEE Access*, 6:22987–22992.

Feng, Y. and Hao, L. (2020). Testing randomness using artificial neural network. *IEEE Access*, 8:163685–163693.

Fischer, T. (2018). Testing cryptographically secure pseudo random number generators with artificial neural networks. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 1214–1223. IEEE.

Gohr, A. (2019). Improving attacks on round-reduced speck32/64 using deep learning. In *Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part II 39*, pages 150–179. Springer.

Gupta, S., Singh, P., Shrotriya, N., and Baweja, T. (2021). Lfsr next bit prediction through deep learning. *Journal of Informatics Electrical and Electronics Engineering (JIEEE)*, 2(2):1–9.

Hashim, K. M. and Abdulhussien, W. R. (2015). Binary sequences randomness test using neural networks.

John Labelle (2020). Everyone Talks About Insecure Randomness, But Nobody Does Anything About It. https://www.airza.net/2020/11/09/everyone-talks-about-insecure-randomness-but-nobody-does-anything-about-it.html. Online; accessed 21 November 2022.

Kant, S. and Khan, S. S. (2006). Analyzing a class of pseudo-random bit generator through inductive machine learning paradigm. *Intelligent Data Analysis*, 10(6):539–554.

Kant, S., Kumar, N., Gupta, S., Singhal, A., and Dhasmana, R. (2009). Impact of machine learning algorithms on analysis of stream ciphers. In *2009 Proceeding of international conference on methods and models in computer science (ICM2CS)*, pages 251–258. IEEE.

Kelsey, J., McKay, K. A., and Sönmez Turan, M. (2015). Predictive models for min-entropy estimation. In *Cryptographic Hardware and Embedded Systems–CHES 2015: 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings 17*, pages 373–392. Springer.

Kim, J. and Kim, H. (2021). Length of pseudorandom binary sequence required to train artificial neural network without overfitting. *IEEE Access*, 9:125358–125365.

L'Ecuyer, P. and Simard, R. (2007). Testu01: Ac library for empirical testing of random number generators. *ACM Transactions on Mathematical Software (TOMS)*, 33(4):1–40.

Li, C., Zhang, J., Sang, L., Gong, L., Wang, L., Wang, A., and Wang, Y. (2020). Deep learning-based security verification for a random number generator using white chaos. *Entropy*, 22(10):1134.

Lv, N., Chen, T., Zhu, S., Yang, J., Ma, Y., Jing, J., and Lin, J. (2020). High-efficiency min-entropy estimation based on neural network for random number generators. *Security and Communication Networks*, 2020:1–18.

Mostafa Hassan (2021). Cracking Random Number Generators using Machine Learning – Part 1: xorshift128. https://research.nccgroup.com/2021/10/15/cracking-random-number-generators-using-machine-learning-part-1-xorshift128/. Online; accessed 21 November 2022.

O'neill, M. E. (2014). Pcg: A family of simple fast space-efficient statistically good algorithms for random number generation. *ACM Transactions on Mathematical Software*.

Pasqualini, L. and Parton, M. (2020). Pseudo random number generation: A reinforcement learning approach. *Procedia Computer Science*, 170:1122–1127.

Robert G. Brown (2017). Dieharder, A Random Number Test Suite. http://webhome.phy.duke.edu/~rgb/General/dieharder.php. Online; accessed 4 October 2022.

Rukhin, A., Soto, J., Nechvatal, J., Smid, M., and Barker, E. (2001). A statistical test suite for random and pseudorandom number generators for cryptographic applications. Technical report, Booz-allen and hamilton inc mclean va.

Savicky, P. and Robnik-Šikonja, M. (2008). Learning random numbers: A matlab anomaly. *Applied Artificial Intelligence*, 22(3):254–265.

Truong, N. D., Haw, J. Y., Assad, S. M., Lam, P. K., and Kavehei, O. (2018). Machine learning cryptanalysis of a quantum random number generator. *IEEE Transactions on Information Forensics and Security*, 14(2):403–414.

Zanin, M. (2022). Can deep learning distinguish chaos from noise? numerical experiments and general considerations. *Communications in Nonlinear Science and Numerical Simulation*, 114:106708.