УДК 004.421.5 http://doi.org/10.15407/pp2024.02-03.280

*В.О. Лесик, А.Ю. Дорошенко*

# ВИКОРИСТАННЯ НЕЙРОННИХ МЕРЕЖ ДЛЯ ГЕНЕРАЦІЇ ВИПАДКОВИХ ПОСЛІДОВНОСТЕЙ

У статті розглядається методологія використання нейронних мереж для імітації псевдовипадкових послідовностей, що дозволяє знаходити приховану структуру та алгоритми послідовності для зведення спостережуваних процесів до детермінованих. Для підвищення якості імітації послідовностей згенерованих чисел використано моделі рекурсивних нейронних мереж із урахуванням їхніх можливостей адаптації до згенерованих неперервних послідовностей. У роботі запропонований метод використання та налаштування рекурсивних нейронних мереж та впливу обраних гіперпараметрів, що визначають внутрішню структуру, розмір вхідної послідовності, довжину результуючої згенерованої послідовності, що в результаті впливає на якість та точність співпадіння нейронного псевдо випадкового генератора та реального програмного еквіваленту. Отримані результати показують можливість використання нейронних методів у процесах хешування, передбачення послідовностей статистично випадкових даних, у криптографічних алгоритмах та алгоритмах стиснення даних або знаходження оригінального процесу генерації досліджуваної послідовності.

Ключові слова: рекурсивні нейронні мережі, генератор псевдовипадкових чисел, імітація генерації послідовностей чисел, TensorFlow.

*V.O. Lesyk, A.Y. Doroshenko*

# NEURAL NETWORK APPLICATION TO PSEUDORANDOM SEQUENCE GENERATION SIMULATION

The article discusses the methodology of using neural networks to simulate pseudorandom sequences, which allows finding the hidden structure and sequence algorithms to reduce the observed processes to deterministic ones. To improve the quality of simulation of sequences of generated numbers, models of recurrent neural networks are used, taking into account their ability to adapt to the generated continuous sequences. The paper proposes a method for using and tuning recurrent neural networks and the influence of selected hyperparameters that determine the internal structure, size of the input sequence, and length of the resulting generated sequence, which ultimately affects the quality and accuracy of the matching neural pseudorandom generator sequences and the real program equivalent. The obtained results show a possibility of using neural methods in the processes of hashing, predicting sequences of statistically random data, cryptographic algorithms and data compression algorithms, or finding the original process of generating the sequence under study.

Key words: recursive neural networks, pseudorandom number generator, number sequence generation imitation, TensorFlow.

## Вступ

Генерація випадкових чисел відіграє важливу роль у сферах криптографії, академічних досліджень та використовується у процесах отримання послідовностей номерів у мережевих протоколах, одноразових номерів у криптографічних протоколах, генерації паролів для захисту даних та багатьох інших процесах. За відсутності елемента випадковості, всі криптографічні операції можна вважати передбачуваними, що в свою чергу означає появу вразливостей, можливості перехоплення повідомлень, отримання ключів захисту та шифрування, отримання доступу до захищених процесів, сеансів, каналів зв'язку.

У даному контексті гостро постає проблема створення методів реверсивної інженерії даних методів, бо достеменне знання випадкового процесу повністю нівелює елемент криптографії та надає можливість відновити вихідний процес. Наразі генератори псевдовипадкових чисел (ПВЧ)

мають детерміновану структуру та повторюють генеровану послідовність через деякий період. Але за рахунок того, що даний період є занадто великим, визначення використаного алгоритму потребує отримання дуже великої частини послідовності із повного періоду, що унеможливлює його відновлення.

Потенційною альтернативою стає використання нейронних мереж, що мають здатність нечітко визначати процеси, які стоять за генерацією ПВЧ послідовностей. Це дає змогу знизити кількість чисел для перебору та відновлення оригінального алгоритму. Дослідження застосування нейронних мереж для імітації конкретних псевдовипадкових процесів може стати достатнім елементом, що наближає процес автоматизованого визначення оригінальної необоротної функції.

Використання мереж у контексті проблеми випадкових послідовностей може висвітлити наявні випадкові процеси реального світу із елементами випадковості. Майбутнім потенційним застосуванням нейромереж до випадковості є отримання методів формалізації функцій та процесів, які слугують підгрунтям генераторів ПВЧ та істинно випадкових послідовностей, враховуючи, що випадкові послідовності чисел є однорідними, та не існує канонічного способу однозначно довести, що набір чисел справді випадковий [1].

**Метою роботи** є проведення аналізу застосування нових підходів, що базуються на принципах штучних інтелектуальних систем. Вони можуть не традиційно описувати та висвітлювати випадкові процеси, що має потенціал надання нової парадигми аналізу послідовностей та наявності випадкових процесів у детерміністичних комп'ютерних процесах.

**Актуальність**. Наразі генератори ПВЧ широко застосовуються у криптографічних алгоритмах, шифруванні та захисті даних. Отже дослідження із використанням штучного інтелекту є потенційним елементом створення систем реверсивної інженерії криптографічних методів як для виділення вразливостей та недоліків, так і для створення більш надійних методів. Даний тип нейронних мереж має додатковий потенціал розширення методів формалізації процесів та навчання за наборами даних із наближено випадковим розподілом.

Використання нейронних мереж у якості генераторів є відносно новим напрямком досліджень. Отже аналіз застосування даної технології до випадкових процесів має потенціал до знаходження нових статистичних методів перевірки наявності випадковості у послідовностях та процесах.

Ключовим елементом дослідження випадкового процесу та отримання уявлення про його алгоритмічну структуру є використання статистичних методів або новітніх елементів – нейронних мереж – для аналізу середовищ, великих даних або послідовностей. Основним використанням нейронних методів є забезпечення безпосереднього процесу імітації послідовності, результат якого можна проаналізувати на наявність відповідних математичних зв'язків та наборів функцій, що є достатніми для опису досліджуваного процесу.

За визначеннями з [2] випадкові числа неможливо обчислити, оскільки цифрові комп'ютери є детермінованими та в даному контексті обмеженими із очевидною альтернативою використання генераторів істинно випадкових чисел. Основні дослідження у напрямку створення випадкових послідовностей зосереджені на наближенні штучних методів генерації послідовностей до отримання істинної випадковості. Дана робота має на меті зміщення уваги від істинної випадковості знаходження та висвітлення можливостей штучних нейронних мереж до виявлення закономірностей у попередньо оцінених послідовностях як випадкових. Іншим важливим аспектом є оцінка даних нейронних систем з огляду на розширення інструментарію статистичних методів отримання прихованих зв'язків та алгоритмів побудови послідовностей чисел чи даних, а також - аналіз даних послідовностей щодо наявності детермінованої випадковості.

## Оцінка існуючих підходів до аналізу, генерації та імітації випадковості

Розвиток та розробка псевдовипадкових алгоритмів починається із середини

двадцятого століття, найперший з яких був розроблений Джоном фон Нейманом, який використовував квадрат числа для отримання певної середньої частини цифр даного числа, що можна вважати близьким до випадкового. Але даний алгоритм є досить неефективним, адже псевдовипадкові числа входять у періодичне кільце досить швидко. Власне наявність недоліків алгоритмів та наближення генерування послідовностей до випадковості стимулювало подальші розробки та модифікації алгоритмів, прикладом якого може слугувати [3].

З часу появи першого алгоритму було розроблено понад 30 псевдовипадкових алгоритмів[4], кожен із яких вводив додаткові покращення із наближення генерованих послідовностей до нормального або випадкового розподілу, пришвидшення, криптографічної стійкості, адаптивності та використання більшого набору знаків і параметрів та інших складових. Найбільш відомі та часто використовувані – це Лінійний конгруентний метод [5], що входить в стандартні бібліотеки компіляторів, який можна вважати тривіальним прикладом та який може бути параметризований відповідно до задачі, алгоритм Блум–Блум–Шуба або Blum Blum Shub [6, 7], що має достатню криптографічну стійкість, алгоритм Вихор Мерсенна або Mersane Twister[8], який можна вважати найбільш дослідженим та розповсюдженим алгоритмом.

Альтернативним варіантом генерації псевдовипадкових послідовностей також можна вважати обчислення ірраціональних чисел та продовження обчислення часток або цифри після відповідної кількості прорахованого порядку точності, так як результат даного обрахунку буде наближатися до випадкового. Підтвердженням даного процесу слугують роботи із розширення обчислення ірраціональних чисел [9] або із використання числа Пі для отримання випадкових чисел [10].

У результаті попереднього аналізу підходів до реалізації випадковості можна зробити припущення про постійну наявність алгоритмічної та визначеної структури, яка прямолінійно або опосередковано впливає на сам принцип генерації тих чи інших послідовностей. Схоже припущення розглядається в роботі [11], де автори намагаються описати питання, що звучить як «Чи грає Бог в псевдо кості?». Результатом оглянутої роботи є підкреслення потенційної можливості нейронних мереж передбачити продовження ірраціональних послідовностей, що можуть інтерпретуватися як випадкові. Автори припускають, що у разі порядку прогнозованості ірраціональних чисел, такі випадкові процеси як квантові генератори випадкових чисел також повинні мати таку властивість. Також автори мотивують проведення розширених досліджень випадкових процесів із використанням більш спеціалізованих методів машинного навчання.

Дана робота є логічним продовженням розглянутої публікації із продовженням відповідних експериментів, що націлені на явне виділення залежностей та патернів у псевдовипадкових послідовностях. Але для даного експерименту потрібно змістити увагу на більш тривіальні алгоритмічні методи генерації та виділення основних нейромережевих підходів , методів налаштування та підбору обраних підходів для отримання системи, що здатна імітувати такі послідовності із певною граничною точністю.

Наразі застосування нейронних мереж для імітації випадкових процесів ще не є повноцінно дослідженою та повністю розглянутою тематикою. Доцільно першим кроком використання нейромереж у поставленій задачі вважати сам аналіз вже використовуваних підходів на практиці. В публікації [11] розглядається базова структура нейронної мережі, тобто нейронні мережі прямого поширення. В роботі [12] використовується Шарова рекурентна нейронна мережа для розробки окремого нейронного методу генерації псевдовипадкових чисел. Автори здійснюють широке дослідження ефективності впливу гіперпараметрів на якість генерації випадковості. Саму модель навчають за попередньо згенерованими випадковими даними. Та основне навчання було сконцентроване на впровадженні випадковості, ніж імітації конкретного випадкового модулю. В роботі [13] увага приділяється застосуванню навчання з підкріп-

ленням та рекурсивні нейронні моделі. Ідея застосування даних підходів полягає в наданні «заохочення» для моделей, що справляються із процесом генерації псевдовипадкових чисел краще, згідно з відповідними метриками. Результатом описаного дослідження є система, яку можна вважати найбільше наближеною до істинно випадкового генератора чисел. Схожий результат отримано з роботи [14], де генератор псевдовипадкових чисел реалізовано на ПКВМ (Програмована Користувачем Вентильна Матриця) із додатковим використанням нейронної мережі Хопфілда та під впливом електромагнітного випромінювання. В даній роботі нейронну мережу можна розглядати як елемент, що розширює коло випадковості даної апаратної реалізації випадкового генератора, та має потенціал на наслідування вивченого процесу продукування випадковості без наявності додаткового впливу джерела ентропії.

Нейронні мережі можна вважати розповсюдженим методом, що використовується в розробці новітньої архітектури та методології формування випадковості. Але розглянуті роботи лише підкреслюють ефективність застосування даних мереж тільки як відокремленого генератора. Питання наслідування та імітації конкретного алгоритму чи методу випадковості залишається відкритим.

## Проведення експерименту

Основною задачею експерименту є підкреслення можливостей застосування рекурсивних моделей для передбачення впорядкованих наборів даних, так як структура даного типу нейронних мереж орієнтована саме на цей тип задачі[15]. Для проведення експерименту обрано програмну мову Python із відповідними розширеннями для роботи з даними: Matplotlib, TensorFlow, Keras, NumPy.

Для дослідження обрано наступні алгоритми: Лінійний Конгруентний Генератор(LCG) та Регістр Зсуву з Лінійним Зворотним (LFSR) зв'язком та Xorshift16, які базуються на кусково-лінійних функціях та побітових операціях відповідно. Наведені алгоритми реалізують процеси генерації послідовностей даних, що статистично відповідають випадковим. Вони реалізують базові принципи генерації випадкових послідовностей, які є основою для сучасних ускладнених алгоритмів, Успішне навчання мережі на послідовностях базових методів підкреслює можливості навчання та імітації моделі і на подальших алгоритмах.

У якості конкретних параметрів обрано коефіцієнти для LCG, що отримано з моделі ZX81, де $m = 65537$, $m = 75$, $c = 74$; у якості LFSR обрано 16-бітний Фібоначчі LFSR із відповідними відводами: [16, 14, 13, 11]. Дані генеруються в проміжку від 0 до 65535, тобто генеруються числа формату uint16. Генерація чисел за допомогою LFSR відбувається за рахунок проходження 16 кроків. Xorshift16 розроблено на основі Xorshift32, але оперує із значеннями int16 та має коефіцієнти здвигів (9, 7 13)

Задача сформульована таким чином, що відомо лише перші 80% послідовності, а наступні 20% необхідно передбачити шляхом імітації системи. Тобто вибірка розбита на приблизно 50000 послідовних значень навчання та 15000 значень для перевірки. Окреслена задача класифікується як навчання із учителем. Результуючий набір нормалізовано до меж [0 : 1]. Модель реалізовано за допомогою структурних блоків програмного пакету TensorFlow: SimpleRNN та Dence, що реалізують поведінку рекурсивного шару та шару прямого розповсюдження відповідно. Обрана довжина блоку послідовності становить 16 чисел; виходом моделі є одне значення, що відповідає конкретному числу. Модель, що задається функціональним інтерфейсом, наведено далі:

$$inputs = Input(shape = (1, 16))$$
$$x = SimpleRNN(64)(inputs)$$
$$x = Dense(64, activation = 'relu')(x)$$
$$x = Dense(32, activation = 'relu')(x)$$
$$x = Dense(1, activation = 'relu')(x)$$

Обрана функція помилки – логарифм гіперболізованого косинусу, бо необхідно мінімізувати різницю між згенерованим числом та дійсним числом послідовно-

сті. Оптимізація здійснюється за допомогою відповідного оптимізатора «Adam», швидкість навчання – 0,001 та наявним кроком зменшення в 2 рази кожні 50 епох. Основна спостережувана метрика – абсолютне відхилення. Навчання моделей проводиться за 300 епох. Результати навчання моделей висвітлено на рисунках 1-3:

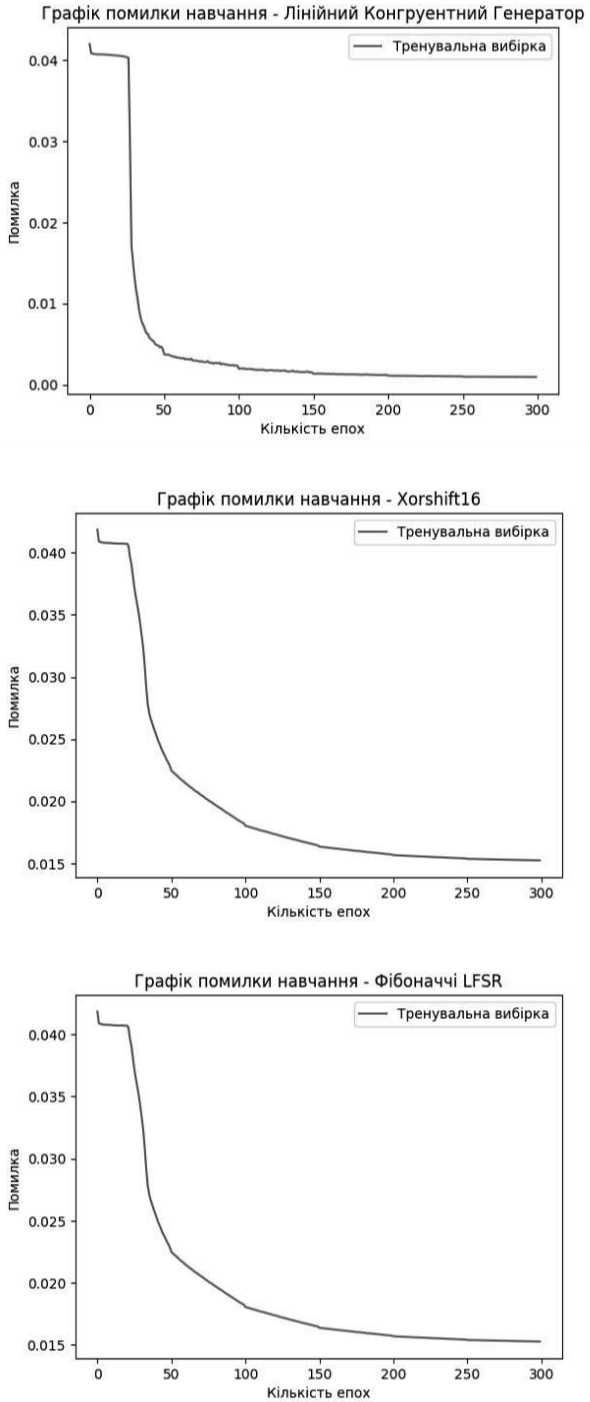


Рис. 1-3. Графіки втрат

Моделі навчаються успішно та досягають лімітованого значення похибки за встановленим набором гіперпараметрів. Помилка на графіку рис. 1 спадає стрімкіше, що свідчить про більшу генералізацію даних. Для повноцінного опису моделі доцільно навести абсолютну похибку всіх можливих значень передбачення. Дані меж похибок зображено у таблиці 1. Таблиця показує відсоток результатів передбачення, що потрапляють у наведений окіл похибки. Похибка вказана у якості цілого числа.

Таблиця 1

Межі абсолютної похибки

| Модель | Межі похибки | | | |
|---|---|---|---|---|
| | < 200 | < 2000 | < 10000 | < 20000 |
| LCG | 21% | 87.7% | 98.6% | 99.4% |
| Xorshift16 | 1.3% | 13.0% | 59.3% | 92.2% |
| Fibonacci LFSR | 1.6% | 16.7% | 72.2% | 96.2% |

За результатами встановлено, що модель найкраще імітує поведінку лінійного конгруентного генератора та у 88% передбачає наступне число послідовності із похибкою менше 2000. Це свідчить про те, що модель проявляє більшу ефективність імітації алгоритмів на основі кусково-лінійних функцій та математичних операцій. Моделі, які навчені на генераторах із використанням бітового зсуву, мають результати із більшою похибкою, але проявляють достатні здатності до генералізації моделі.

Отже використання даних моделей у точних операціях не є ефективним через наявність наближених значень, але у загальному випадку застосування рекурсивних моделей надає задовільний результат для отримання припущень або нечітких знань про процес генерації випадковості, що доцільно використовувати у подальших дослідженнях.

Завершальним методом перевірки результатів є візуальний метод. Відтворимо згенеровані передбачення та дійсні числа на одному графіку для отримання загальної картини. Достатньо представити перші 200 передбачень.

Отримані візуалізації зображено на рисунках 4-6:

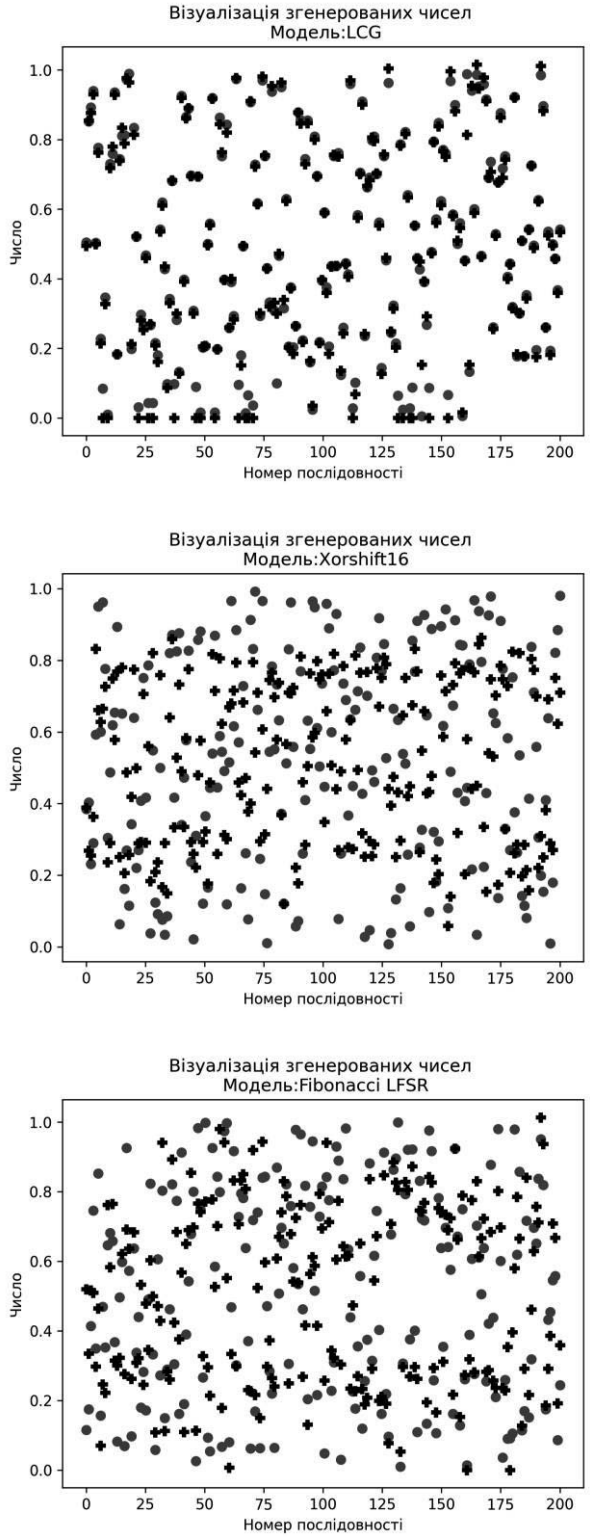


Рис. 4-6. Візуалізація результатів *Примітка: «коло» - дійсне значення, «хрестик» - передбачення.*

Але в реальності дані методи генерації чисел доцільно вважати циклічними, послідовними. Такі дані не можна чітко описати класичною парадигмою. Тому для покращення результатів та отримання точних значень доцільно використовувати відповідну структуру у самій моделі, тобто необхідно розробити відповідну функцію помилки, що враховує циклічність, достатньо гнучкі та відповідні функції активації. Треба інтегрувати елементи побітових операцій, часткову або повноцінну логіку випадкових процесів у саму структуру нейронної мережі. Варто використовувати покращені рекурсивні моделі, такі як моделі з короткою та довготривалою пам'яттю або трансформери.

## Висновки

У статті розглянуто загальні методи використання моделі рекурсивної нейронної мережі для відтворення псевдовипадкової послідовності та передбачення наступного значення послідовності. Проста реалізація нейронної мережі, що наслідує поведінку базового випадкового генератора чисел, є прикладом можливості методів машинного навчання отримувати знання про середовище та процеси, що попередньо оцінюються як випадкові. Проведені експерименти з імплементації процесу формалізації внутрішніх зв'язків свідчать про потенційну можливість визначення структури зовнішніх та внутрішніх (програмних) джерел ентропії. Експериментальна модель вказує також на можливість майбутніх реалізацій, що можуть виявляти закономірності в складних випадкових криптографічно стійких алгоритмах, слугувати елементами формалізації послідовностей у якості оборотних функцій чи генераторів із відповідними ідентифікаторами.

Дана робота є потенційним елементом, що може нівелювати проблеми базових структур автокодувальників[16] і стати елементом квантифікації даних у нейронних мережах. Розглянута модель передбачає подальше розширення та може бути покращена за допомогою інтеграції архітектури, що використовує підходи довгої та короткої пам'яті рекурсивних мереж, можливості нейроеволюційного методу, бітових "flip-flop" елементів та побітових нейронних мереж.

## References


1. CHAITIN, Gregory J. Randomness and mathematical proof. *Scientific American*, 1975, 232.5: 47-53. Accessed: 10.04.2024. https://behdad.org/download/Presentations/chaitin75/sciamer.html
2. STIPČEVIĆ, Mario; KOÇ, Çetin Kaya. True random number generators. In: Open Problems in Mathematics and Computational Science. Cham: Springer International Publishing, 2014. p. 275-315. DOI 10.1007/978-3-319-10683-0_12
3. WIDYNSKI, Bernard. Middle-Square weyl sequence RNG. arXiv preprint arXiv:1704.00358, 2017. Accessed: 10.04.2024. https://arxiv.org/pdf/1704.00358.pdf
4. List of random number generators. Accessed: 10.04.2024. https://en.wikipedia.org/wiki/List_of_random_number_generators
5. TEZUKA, Shu. Linear congruential generators. In: Uniform Random Numbers: Theory and Practice. Boston, MA: Springer US, 1995. p. 57-82.
6. BLUM, Lenore; BLUM, Manuel; SHUB, Mike. A simple unpredictable pseudo-random number generator. SIAM Journal on computing, 1986, 15.2: 364-383. Accessed: 10.04.2024. https://www.academia.edu/download/89296292/91cdc1da67c52e606cd4752472ce0db83131.pdf
7. JUNOD, Pascal. Cryptographic secure pseudo-random bits generation: The Blum-Blum-Shub generator. Unpublished, 1999. Accessed: 10.04.2024. https://www.academia.edu/download/30436611/bbs.pdf
8. MATSUMOTO, Makoto; NISHIMURA, Takuji. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Transactions on Modeling and Computer Simulation (TOMACS), 1998, 8.1: 3-30. Accessed: 10.04.2024. https://dl.acm.org/doi/pdf/10.1145/272991.272995
9. GHODOSI, Hossein, et al. Pseudorandom sequences obtained from expansions of irrational numbers. Department of Computer Science Center for Computer Security Research, University of Wollongong Australia, 1995. Accessed: 10.04.2024. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=b3fb3b8824fa60d785dc0633df6ec0323dbb02a7
10. ROGERS, Ilya; HARRELL, Greg; WANG, Jin. Using [pi] digits to Generate Random Numbers: A Visual and Statistical Analysis. In: *Proceedings of the International Conference on Scientific Computing (CSC)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2015. p. 251.
11. FAN, Fenglei; WANG, Ge. Learning from pseudo-randomness with an artificial neural network–does god play pseudo-dice?. IEEE Access, 2018, 6: 22987-22992. Accessed: 10.04.2024. https://ieeexplore.ieee.org/iel7/6287639/6514899/08350369.pdf
12. DESAI, Veena; PATIL, Ravindra; RAO, Dandina. Using layer recurrent neural network to generate pseudo random number sequences. International Journal of Computer Science Issues, 2012, 9.2: 324-334. Accessed: 10.04.2024. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=9302db89e8e94e3d93608c2c4498af09e5446144
13. PASQUALINI, Luca; PARTON, Maurizio. Pseudo random number generation through reinforcement learning and recurrent neural networks. Algorithms, 2020, 13.11: 307. Accessed: 10.04.2024. https://www.mdpi.com/1999-4893/13/11/307/pdf
14. YU, Fei, et al. Design and FPGA implementation of a pseudo-random num-

ber generator based on a Hopfield neural network under electromagnetic radiation. Frontiers in Physics, 2021, 9: 690651. Accessed: 10.04.2024. https://www.frontiersin.org/articles/10.3389/fphy.2021.690651/pdf

15. BLOG, Andrej Karpathy. The unreasonable effectiveness of recurrent neural networks. May, 2015, 21: 31. Accessed: 10.04.2024. URL: https://karpathy.github.io/2015/05/21/rnn-effectiveness/
16. LESYK, V. O.; DOROSHENKO, A. Yu. Image compression module based neural network autoencoders. PROBLEMS IN PROGRAMMING, 2023, 1: 48-57.

Одержано: 20.04.2024
Внутрішня рецензія отримана: 30.04.2024
Зовнішня рецензія отримана: 04.05.2024

***Про авторів***:

*Лесик Валентин Олександрович*,
аспірант
http://orcid.org/0000-0002-8307-5707.

*Дорошенко Анатолій Юхимович*,
доктор фізико-математичних наук, професор, завідувач відділу
http://orcid.org/0000-0002-8435-1451.

***Місце роботи авторів:***

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»,
Берестейський проспект, 37, м. Київ, Україна, індекс 03056

E-mail: lesykvalentine@gmail.com