УДК 004.89

Д. Паєвський, К. Остапченко, О. Лісовиченко

ГЕНЕТИЧНІ АЛГОРИТМИ В АВТОМАТИЗАЦІЇ СТВОРЕННЯ НЕЙРОННИХ МЕРЕЖ

Анотація: Предметом розгляду статті є процеси проектування моделей нейронних мереж при розв’язанні прикладних задач. Представлено огляд основних підходів до автоматизованого проектування штучних нейронних мереж, проаналізовано вплив різних параметрів на результат проектування ШНМ. Запропонована методика застосування генетичних алгоритмів організації оптимального пошуку моделей нейронних мереж з акцентом на збалансування продуктивності та використання обчислювальних ресурсів. Проведено експериментальне дослідження методики визначення параметрів, що впливають на процеси пошуку оптимальних моделей.

Ключові слова: штучні нейронні мережі, автоматизоване проектування моделей ШНМ, генетичні алгоритми, математичне моделювання, засоби моделювання ШНМ.

Вступ

Штучні нейронні мережі (ШНМ), завдяки своїй здатності до навчання, можуть адаптуватися до нових умов та виявляти приховані закономірності в даних, що робить їх незамінними у багатьох галузях, включаючи медицину, фінансові технології, промисловість та розробку інтелектуальних систем. Однак, ефективне проектування моделей ШНМ є складним завданням, яке вимагає глибоких знань, розуміння принципів роботи алгоритмів навчання, а також багатократного налаштування параметрів, що є трудомістким і часовитратним процесом, який часто потребує великої кількості експериментів і тонкого налаштування. Крім того, процес проектування вимагає великої кількості обчислювальних ресурсів, що також є значним обмеженням для дослідників та проектувальників [1].

Актуальність теми зумовлена необхідністю підвищення ефективності та точності проектування моделей ШНМ, що дозволить не лише спростити процес створення та налаштування моделей, а й забезпечити їхню адаптацію до змінних умов, що особливо важливо для задач прогнозування та аналізу великих обсягів даних.

Аналіз існуючих рішень

Сучасні засоби моделювання нейронних мереж є важливим інструментом для розробників і науковців, які займаються створенням та вдосконаленням моделей ШНМ. Розвиток технологій у галузі штучного інтелекту та машинного навчання призвів до появи численних програмних платформ, які значно полегшують процес

проектування, аналізу та налаштування моделей ШНМ. Використання таких засобів дозволяє швидко і ефективно створювати моделі, тестувати їх на великих обсягах даних, проводити оптимізацію та виявляти оптимальні моделі ШНМ.

Сучасні підходи до синтезу ШНМ охоплюють використання класичних алгоритмів оптимізації та евристичних методів, зокрема генетичних алгоритмів.

Класичні методи, такі як градієнтний спуск, є ефективними у випадках, коли цільова функція є гладкою, а навчання можна проводити в детермінованих умовах. Однак ці методи часто схильні до застрягання у локальних мінімумах, що заважає знаходженню оптимального рішення. Це обмеження особливо відчутне під час синтезу складних ШНМ, де значна кількість параметрів потребує оптимізації.

Евристичні методи, такі як генетичні алгоритми, пропонують більш гнучкий підхід до синтезу моделей ШНМ. Вони здатні уникати локальних мінімумів завдяки випадковим мутаціям та схрещуванням, що дозволяє шукати глобально оптимальні рішення. Однак значним обмеженням таких підходів є високі обчислювальні витрати. Виконання кожної ітерації генетичного алгоритму передбачає навчання великої кількості моделей-кандидатів, що може бути надзвичайно ресурсномістким та вимагати тривалого часу обчислення, особливо для складних моделей ШНМ.

Ще однією проблемою сучасних підходів до синтезу моделей ШНМ є налаштування гіперпараметрів, серед яких розглядаються швидкість навчання, розмір пакетів даних, кількість епох, тип і параметри оптимізатора. Наявність автоматизованих методів, таких як Bayesian Optimization та Random Search, ці підходи залишаються трудомісткими і не завжди гарантують оптимальні результати. А невідповідний вибір топології може призвести до таких проблем, як перенавчання або недонавчання, що негативно впливає на ефективність моделі. Генетичні алгоритми також використовуються для пошуку параметрів, однак їхня ефективність суттєво залежить від початкових умов і параметрів, таких як розмір популяції та ймовірність мутації.

Серед сучасних засобів моделювання ШНМ з використання евристичних методів виділяють наступні підходи, які мають специфічні можливості для вирішення різних типів прикладних задач:

1. NEAT (NeuroEvolution of Augmenting Topologies) – це інструмент, який використовує генетичний алгоритм для еволюції архітектури ШНМ. Він дозволяє автоматично змінювати топологію мережі, що сприяє підвищенню адаптивності та продуктивності моделі [2].

2. HyperNEAT – розширений NEAT, спеціалізований на еволюції велико-масштабних архітектур ШНМ. Він особливо ефективний у задачах обробки зображень та інших завдань, де структура мережі має важливе значення [3].

3. ES-HyperNEAT (Evolution Substrate HyperNEAT) – застосовує еволюційні стратегії для оптимізації параметрів ШНМ. Це дозволяє знаходити оптимальні параметри за допомогою паралельних обчислень [4].

4. Deep Genetic Programming – використовує генетичне програмування для автоматичної еволюції структури та параметрів глибоких ШНМ. Це забезпечує швидке та ефективне навчання мережі для різних задач машинного навчання [5].

5. CoDeepNEAT – використання генетичних алгоритмів для еволюції мережі в спільному зі структурою задачі, що дозволяє побудувати ШНМ, оптимально пристосовану до конкретної задачі [6].

6. MM-NEAT (Multiobjective Modular Neural Evolution of Augmenting Topologies) – використання модульної структури для еволюції ШНМ, де кожен модуль може еволюціонувати окремо, а потім об'єднуватись у готову мережу [7].

В табл.1 наведений порівняльний аналіз ключових характеристик та показники різних підходів автоматизації проектування моделей ШНМ. Серед ключових характеристик виділені наступні:

– швидкість – описує час за яких відбується синтез моделі ШНМ, де висока швидкість передбачає майже миттєву обробку задач, тоді як середня швидкість може бути повільнішою, але залишається практичною;

– якість – описує точність і продуктивність роботи моделі ШНМ, де висока якість означає максимальну точність і продуктивність моделей, тоді як для специфічних завдань, наприклад малих або великих мереж, це може мати уточнення;

– гнучкість – відображає здатність методу адаптуватися до різних задач, наприклад, висока гнучкість дозволяє більше варіантів використання, а обмежена – значно зменшує можливості;

– витрати ресурсів – відображає необхідні обчислювальні ресурси, які варіюються від низьких (економних) до високих (ресурсоємних алгоритмів);

– складність – характеристика, що може бути критичною для вибору методу: складні алгоритми потребують більше знань, тоді як помірні і середні є доступнішими.

Таблиця 1. **Показники підходів автоматизації проектування моделей ШНМ**

Підхід	**Швидкість**	**Якість**	**Гнучкість**	**Витрата ресурсів**	**Складність**
NEAT	Середня	Висока для малих мереж	Обмежена	Низька	Помірна
HyperNEAT	Висока	Висока для великих мереж	Висока	Висока	Висока

Закінчення табл. 1

Підхід	**Швидкість**	**Якість**	**Гнучкість**	**Витрата ресурсів**	**Складність**
ES-HyperNEAT	Висока	Висока (залежить від задачі)	Висока	Висока	Складна
Deep Genetic Programming	Висока	Висока	Помірна	Середня	Складна
CoDeepNEAT	Висока	Висока	Висока	Висока	Середня
MM-NEAT	Середня	Висока	Висока (модульна структура)	Висока	Помірна

Зазначені підходи мають свої унікальні можливості та переваги, що робить їх придатними для різних завдань у галузі моделювання та використання ШНМ. Проте, кожен із них має свої обмеження, такі як складність реалізації або високі витрати обчислювальних ресурсів. Це підкреслює необхідність розробки нових рішень для подолання цих проблем.

Постановка задачі

Проблема проектування моделей ШНМ полягає у складності вибору оптимальної архітектури та гіперпараметрів для досягнення найвищої ефективності у вирішенні конкретної прикладної задачі. Існуючі методи оптимізації потребують значних обчислювальних ресурсів та багатократного ручного налаштування, що ускладнює процес проектування, особливо для великих і складних мереж [8].

З огляду на ці виклики, ставиться задача створення методики проектування моделей ШНМ, яка використовує генетичні алгоритми для автоматизованого пошуку оптимальних архітектур і налаштування гіперпараметрів. Сформульована задача передбачає розробку алгоритму, який би інтегрував генетичні алгоритми з процесом навчання моделей ШНМ, дозволяючи ефективно комбінувати етапи вибору архітектури, гіперпараметрів та їх подальшого навчання. Перспективи подальшого розвитку в цій галузі включають поєднання різних методів оптимізації, таких як генетичні алгоритми і градієнтний спуск, що дозволить підвищити ефективність пошуку оптимальних моделей ШНМ. Використання розподілених обчислень і хмарних технологій сприятиме зниженню обчислювальних витрат, а застосування навчання з підкріпленням може забезпечити адаптивність моделей до зміни середовища та підвищення їх стійкості.

Математичне моделювання задачі пошуку параметрів

Процес конструювання моделей ШНМ можна представити, як сукупність чітко визначених етапів, формалізація яких забезпечить автоматизацію проектування і підвищить ефективність розробки моделі. Формалізація процесу є критичною для точної послідовності дій і мінімізації впливу людського фактора та включає наступні етапи (рис. 1):

1. вибір архітектури моделі ШНМ визначається шляхом максимізації цільової функції $\max_S Q(S, D)$, де S – кількість шарів, нейронів у кожному шарі, типи функцій активації за умови, де D - це по оцінює мережі на даному навчальному наборі D на ітерації конструювання t;

2. налаштування параметрів – задача пошуку в просторі параметрів із застосуванням методів Bayesian Optimization або генетичних алгоритмів, щоб мінімізувати функцію втрат $\min_P L(P, S, D)$ за умови $S = S_{opt}$, $D = D_t$, де P – вектор гіперпараметрів, D – навчальний набір, S – архітектура моделі;

3. навчання моделі – мінімізація функції втрат шляхом корекції ваг моделі ШНМ, щоб зменшити значення функції втрат $\min_W L(W, P, S, D)$ за умови $P = P_{opt}$, $S = S_{opt}, D = D_t$, де W представляє ваги моделі, D – навчальні дані, S – архітектура моделі, P – гіперпараметри моделі.

Рисунок 1. Схема етапів конструювання моделей ШНМ

Задача оптимального пошуку архітектури ШНМ полягає у виборі параметрів моделі M, що безпосередньо впливають на її якість, до яких віднесені (написати простіше): кількість внутрішніх шарів ШНМ, кількість нейронів у внутрішньому шарі, активаційна функції у внутрішньому шарі, використання зміщення для кожного нейрону у внутрішньому шарі.

Оптимізація здійснюється за умов виконання декількох обмежень, що стосуються параметрів моделі:

– мінімальна та максимальна кількість шарів (k_{min}, k_{max}) – обмеження на кількість шарів для забезпечення стабільності навчання;

– мінімальна та максимальна кількість нейронів у шарі (n_{min}, n_{max}) – обмеження на кількість нейронів для уникнення надмірного спрощення або перенавчання;

– діапазон значень для гіперпараметрів – визначення діапазонів для таких гіперпараметрів, як швидкість навчання (v_{min}, v_{max}), кількість епох (e_{min}, e_{max}) тощо.

Задача оптимального пошуку моделі ШНМ формально визначається залежністю:

$$M = Neuro(S_{input}, L_{internal}, L_{output}), \quad (1)$$

де S_{input} – параметр вхідного шару ШНМ, $L_{internal}$ – параметр внітрішніх шарів ШНМ, L_{output} – параметр вихідного шару ШНМ. Змінним (пошуковим) у формулі (1) є параметр внутрішнього шару, який подається, як $L_{internal} = (l_1, l_2, \dots, l_k)$ набір внутрішніх шарів з кількість шарів k.

Кожен шар l_i у $L_{internal}$ описується наступною трійкою параметрів:

$$l_i = (n_i, f_i, b_i) \quad (2)$$

де n_i – кількість нейронів у шарі, f_i – активаційна функція шару з множини $F = \{\text{ReLU}, \text{sigmoid}, \tanh, \text{softmax}, \text{linear}\}$, b_i – параметр, що вказує, чи використовується зміщення (*bias*), $b_i \in \{0, 1\}$.

Пошук моделі здійснюється за умов:

– критерію мінімізації функції втрат $L(M)$ або максимізації точності $A(M)$;

– підпорядкування обмеженням $k_{min} \le k \le k_{max}$, $n_{min} \le n\_i \le n_{max}$, $f_i \in F$, де F – множина можливих активаційних функцій.

Генетичний алгоритм використовує аналоги біологічних операцій схрещування, мутації та відбору для пошуку оптимальних архітектур.

Ітераційний процес алгоритму включає наступні етапи (рис. 2):

1. Ініціалізація популяції моделей із випадковими значеннями параметрів. На початковому етапі генетичного алгоритму створюється початкова популяція моделей $M_1, M_2, \dots, M_n$, кожна з яких має випадкові значення параметрів, що відповідають за архітектуру.

 ISSN 1560-8956

2. Оцінка якості кожної моделі за обраною цільовою функцією. Оцінка якості дозволяє встановити, наскільки ефективна кожна конкретна архітектура.

3. Відбір найкращих моделей, наприклад, турнірним методом або методом рулетки (пропорційної селекції). Результат відбору стає основою для створення нової популяції моделей.

4. Процес завершується, коли виконується одна з наступних умов:

– досягнення встановленого порогу якості;

– досягнення максимальної кількості ітерацій;

– відсутність покращень протягом кількох поколінь.

5. Мутація параметрів для утворення нової популяції. Параметри змінюють випадково з певною ймовірністю:

– кількість шарів – значення кількості може бути збільшена або зменшена;

– кількість нейронів у шарах змінюється у заданому діапазоні;

– активаційна функція шару – змінюється на іншу з множини можливих функцій;

– параметр зміщення – змінюється на протилежний з визначною ймовірністю.

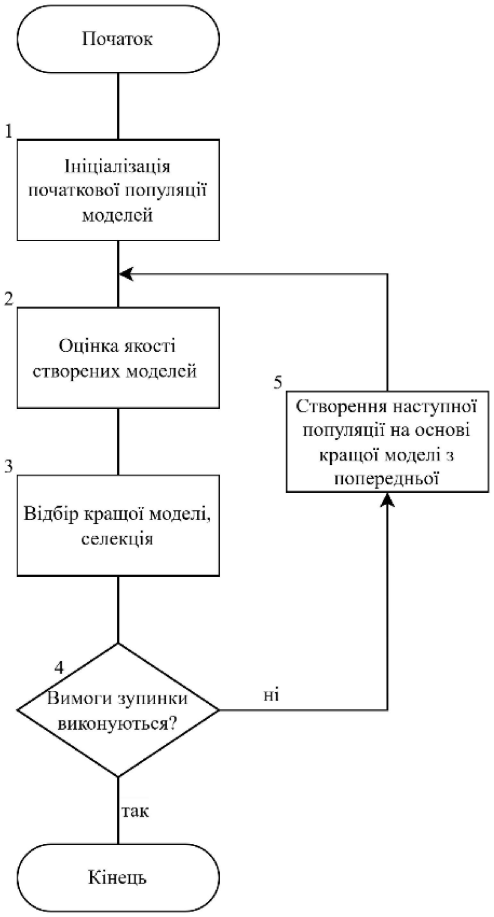

Рисунок 2. Схема етапів генетичного алгоритму

Отже, процес генетичного алгоритму формально можна подати, як послідовне застосування операторів мутації, схрещування та відбору до поточної популяції моделей:

$$P_{t+1} = Select\big(Mutate(P_t)\big), \tag{3}$$

де P_t – популяція на t-й ітерації, $Mutate$ – оператор мутації, $Select$ – оператор відбору. Цей процес повторюється до виконання умов зупинки.

Розглянемо детально подання оператора мутації. Мутаційні зміни параметрів моделі відбуваються з ймовірністю, значення якої від 0 до 1 трактується наступним чином: 0 – зміни відсутні, а 1 – зміни точно стануться. Параметри моделі визначають стадії процедури мутації моделі.

Мутація активаційної функції – для кожного шару активаційна функція f_i змінюється з ймовірністю p_{act}:

$$f_i' = Random(p(F \setminus f_i) = p_{act}),$$

де F – множина всіх можливих активаційних функцій.

Мутація кількості нейронів – кількість нейронів n_i у шарі змінюється в межах заданого діапазону з ймовірністю $p_{neurons}$:

$$n_i' = Random(p([n_i - \Delta, n_i + \Delta]) = p_{neurons}),$$

де Δ – допустиме відхилення.

Мутація кількості шарів – зміна у структурі внутрішніх шарів $L_{internal}$ може відбуватися внаслідок видалення існуючого l_i або додавання нового шару l_{new} з ймовірністю p_{layers}:

$$L_{internal}' = Random(p(L_{internal}) = p_{layers}).$$

Якщо здійснюється видалення, то один із шарів l_i вибирається випадково. Якщо здійснюється додавання, то новий шар l_{new} відповідно до (2) визначається такими змінами:

$$n_{new} = Random([\bar{n} - \Delta, \bar{n} + \Delta]),$$
$$f_{new} = Random(F),$$
$$b_{new} = Random(\{0, 1\}),$$

де $\bar{n}$ – середня кількість нейронів у наявних шарах.

Мутація параметра нейрона зміщення – з ймовірністю p_{bias} параметр b_i змінюється на протилежне значення:

$$b_i' = Random(p([1 - b_i]) = p_{bias}).$$

В результаті процедуру мутації можна описати оператором $\mathcal{M}$, який перетворює модель M у нову конфігурацію M':

$$M' = \mathcal{M}\big(M, p_{act}, p_{neurons}, p_{layers}, p_{bias}\big),$$

а сам оператор мутації $\mathcal{M}$ стає композицією складових:

 ISSN 1560-8956

$$\mathcal{M} = \mathcal{M}_{act} \circ \mathcal{M}_{neurons} \circ \mathcal{M}_{layers} \circ \mathcal{M}_{bias},$$

де $\mathcal{M}_{act}$ – модифікатор активаційної функції, $\mathcal{M}_{neurons}$ – модифікатор кількості нейронів, $\mathcal{M}_{layers}$ – модифікатор складу внутрішніх шарів, $\mathcal{M}_{bias}$ – модифікатор зміщення *bias*.

Для забезпечення стабільності та валідності моделі ШНМ модифікатор повинен гарантувати виконання наступних умов:

1. для всіх шарів i кількість нейронів $n_i \geq 1$;
2. мінімум один внутрішній шар $|L_{internal}| \geq 1$;
3. $f_i \in F, b_i \in \{0, 1\}$.

Побудована математична модель алгоритму генетичного пошуку забезпечує контрольовану та ефективну мутацію, що стає основою для автоматизації проектування моделей ШНМ.

Експериментальне дослідження

Проведено серію експериментів з метою дослідження впливу ймовірності мутації різних параметрів моделі ШНМ на її якість. Досліджувався вплив, як окремих параметрів, так і комбінований вплив всіх одночасно за двома критеріями оцінки точності Acc(%) та функції втрат Loss(MSE). В якості експериментальних даних використано набори даних задачі про виживання пасажирів Титаніка, яка широко використовується в задачах машинного навчання [9].

Результати продемонстрували, що оптимальна ймовірність мутації всіх параметрів у 80% дозволила досягти точності 85% з мінімальною середньоквадратичною помилкою MSE=0,13. Збільшення ймовірності до 100% призводило до зниження точності через надмірну випадковість. Результати цього експерименту наведені у табл.2.

Таблиця 2. **Дослідження комбінованого впливу ймовірності мутації**

№	Вірогідність мутації функції активації	Вірогідність мутації нейронів	Вірогідність мутації внутрішніх шарів	Вірогідність мутації bias	Acc (%)	Loss (MSE)
1	20%	20%	20%	20%	83,8%	0,13
2	40%	40%	40%	40%	83,4%	0,13
3	60%	60%	60%	60%	84,3%	0,14
4	80%	80%	80%	80%	85,2%	0,13
5	100%	100%	100%	100%	84,8%	0,13

Дослідження окремих параметрів (з почерговим виставленням ймовірності мутації кожного окремого параметра у 100%, а інших у 15%) показало, що 100%

ймовірність мутації функцій активації та нейронів забезпечила найкращі результати точності 87,4% та 85,2% відповідно. Це підкреслює важливість ретельного налаштування параметрів мутацій для досягнення високої ефективності. Результати експериментів з окремими параметрами наведені у табл.3.

Таблиця 3. **Дослідження впливу ймовірності мутації окремих параметрів**

№	Вірогідність мутації функції активації	Вірогідність мутації нейронів	Вірогідність мутації внутрішніх шарів	Вірогідність мутації bias	Acc (%)	Loss (MSE)
1	100%	15%	15%	15%	85,2%	0,12
2	15%	100%	15%	15%	87,4%	0,11
3	15%	15%	100%	15%	84,7%	0,13
4	15%	15%	15%	100%	84,3%	0,16

Аналіз критеріїв відбору моделей виявив компроміс між точністю та втратами: критерій на основі втрат забезпечив максимальну точність 84,3% та втрати 0,133, тоді як критерій на основі точності дав менші втрати 0,124 при цьому точність 82,5%.

Порівняння з алгоритмом випадкового підбору параметрів підтвердило переваги використання генетичних алгоритмів, які забезпечують більшу точність до 85% проти 82% та більшу гнучкість у налаштуванні моделі.

Висновки

Через нелінійність і складність простору параметрів, пошук глобального оптимуму для ШНМ є недоцільним, а генетичні алгоритми дозволяють знаходити наближені локальні рішення, які забезпечують баланс між точністю і обчислювальними ресурсами.

Результати проведених експериментів демонструють, що використання генетичних алгоритмів у проектуванні моделей ШНМ значно підвищує точність та стабільність моделей порівняно з випадковим підбором параметрів. Оптимальна ймовірність мутації у 80% забезпечила найкращі результати, а окреме налаштування параметрів дозволило досягти точності до 87,4%. Вибір критерію відбору моделей впливає на баланс між точністю та втратами, підкреслюючи важливість адаптації методу до конкретних завдань прикладних задач.

Подальший розвиток методів оптимізації може включати використання гібридних підходів, які комбінують переваги генетичних алгоритмів із класичними методами оптимізації, такими як градієнтний спуск. Інтеграція хмарних обчислень і розподілених систем дозволить значно знизити обчислювальні витрати, розширюючи

 ISSN 1560-8956

практичну застосовність таких підходів у завданнях прогнозування, обробки великих обсягів даних і створення адаптивних систем.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. *Паєвський Д.В.* Автоматизоване проектування моделей нейронних мереж з використанням генетичних алгоритмів / Д.В. Паєвський, наук.керівник К.Б. Остапченко // *Сучасні аспекти та перспективні напрямки розвитку науки*: матеріали VI Міжнар. студ. наукова конф., м. Харків, 19 січня 2024р. – Вінниця: Укрлогос Груп, 2024. – С. 273-274 website URL: https://archive.liga.science/index.php/conference-proceedings/issue/view/inter-19.01.2024/61

2. NEAT Overview – NEAT-Python 0.92 documentation. *Welcome to NEAT-Python 0.92 documentation! – NEAT-Python 0.92 documentation*. URL: https://neat-python.readthedocs.io/en/latest/neat_overview.html (application date: 25.12.2024).

3. *Shevchenko E.* HyperNEAT Approach in Neuroevolution. *Medium*. website URL: https://medium.com/@eugenesh4work/hyperneat-approach-in-neuroevolution-d2ead10aad33 (application date: 25.12.2024).

4. Evolutionary Substrate HyperNEAT (ES-HyperNEAT). *Schneppat AI*. website URL: https://schneppat.com/evolutionary-substrate-hyperneat_es-hyperneat.html (application date: 24.12.2024).

5. Genetic Programming – DEAP 1.4.1 documentation. *DEAP documentation – DEAP 1.4.1 documentation*. website URL: https://deap.readthedocs.io/en/master/tutorials/advanced/gp.html (application date: 24.12.2024).

6. GitHub - sash-a/CoDeepNEAT: An implementation of CoDeepNEAT using pytorch with extensions. *GitHub*. website URL: https://github.com/sash-a/CoDeepNEAT (application date: 05.01.2025).

7. GitHub - schrum2/MM-NEAT: Modular Multiobjective (Hyper) Neuro-Evolution of Augmenting Topologies + MAP-Elites: Java code for evolving intelligent agents in Ms. Pac-Man, Tetris, and more, as well as code for Procedural Content Generation in Mario, Zelda, Minecraft, and more!. *GitHub*. website URL: https://github.com/schrum2/MM-NEAT (application date: 24.12.2024).

8. *Паєвський, Д.В.* Система автоматизованого проектування моделей нейронних мереж для задач прогнозування : дипломний проєкт бакалавра : 126 Інформаційні системи та технології / Паєвський Дмитро Вікторович. – Київ: КПІ ім.Ігоря Сікорського, 2023. – 85 с. website URL: https://ela.kpi.ua/handle/ 123456789/58750

9. GitHub - Technical-PD/sdnn_ga_api at develop. *GitHub*. website URL: https://github.com/Technical-PD/sdnn_ga_api/tree/develop (application date: 07.12.2024).