Известия СПбГЭТУ «ЛЭТИ». 2026. Т. 19, № 2. С. 51–61
LETI Transactions on Electrical Engineering & Computer Science. 2026. Vol. 19, no. 2. P. 51–61

УДК 004.89 Научная статья

https://doi.org/10.32603/2071-8985-2026-19-2-51-61

Оптимизация архитектуры многослойного перцептрона с помощью алгоритма роя частиц для повышения эффективности обнаружения сетевых атак на IoT-устройства

Д. С. Добровольский✉, Ю. Н. Кожубаев

Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, Россия

✉ d.dobrovolskiy4137@gmail.com

Аннотация. Проводится анализ современных исследований в области использования нейронных сетей в связке с алгоритмом роя частиц (PSO) для обнаружения аномалий в сетевом трафике IoT-устройств. Цель статьи состоит в исследовании потенциала алгоритма роя частиц для автоматизированного проектирования архитектуры нейронной сети. Материалы и методы: проведен анализ научной литературы, а также выполнен сравнительный эксперимент для двух моделей многослойного перцептрона (MLP). Архитектура одной из моделей была спроектирована с помощью оптимизации алгоритмом роя частиц, а другой – эмпирическим способом. Эксперимент проводился для решения задачи классификации сетевого трафика IoT-устройств. В ходе исследования были получены следующие результаты: экспериментально подтверждено, что архитектура MLP, оптимизированная с помощью алгоритма роя частиц, превосходит по качеству решения задачи классификации архитектуру MLP, спроектированную эмпирическим способом. В заключении отмечается, что метод автоматизированного проектирования архитектуры нейронной сети на основе алгоритма роя частиц показал свою эффективность, превзойдя эмпирический подход. Работа вносит вклад в развитие направления Neural Architecture Search (NAS), открывая перспективы для создания более эффективных моделей безопасности IoT-устройств и применения PSO для оптимизации архитектур других типов нейронных сетей.

Ключевые слова: алгоритм роя частиц, нейронные сети, многослойной перцептрон, интернет вещей, оптимизация, обнаружение аномалий, поиск нейронной архитектуры

Для цитирования: Добровольский Д. С., Кожубаев Ю. Н. Оптимизация архитектуры многослойного перцептрона с помощью алгоритма роя частиц для повышения эффективности обнаружения сетевых атак на IoT-устройства // Изв. СПбГЭТУ «ЛЭТИ». 2026. Т. 19, № 2. С. 51–61. doi: 10.32603/2071-8985-2026-19-2-51-61.

Конфликт интересов. Авторы заявляют об отсутствии конфликта интересов.

Review article

Optimization of Multilayer Perceptron Architecture Using Particle Swarm Optimization for Enhanced Network Attack Detection on IoT Devices

D. S. Dobrovolskii✉, Yu. N. Kozhubaev

Peter the Great St. Petersburg Polytechnic University (SPbPU), Saint Petersburg, Russia

✉ d.dobrovolskiy4137@gmail.com

Abstract. The study conducts an analysis of recent research of the use of neural networks in conjunction with the Particle Swarm Optimization (PSO) algorithm for detecting anomalies in IoT device network traffic. The aim of the study is to investigate the potential of the Particle Swarm Optimization algorithm for the automated design of neural network architecture. Materials and methods: an analysis of scientific literature was carried out, and a comparative experiment was conducted for two Multilayer Perceptron (MLP) models. The architecture of one model was designed using Particle Swarm Optimization, while the other was designed empirically. The ex-

periment was conducted for the task of classifying IoT device network traffic. The following results were obtained during the study: it was experimentally confirmed that the MLP architecture optimized using the PSO algorithm outperforms the empirically designed MLP architecture in terms of classification task quality. In conclusion, it is noted that the method of automated neural network architecture design based of the PSO algorithm demonstrated its effectiveness, surpassing the empirical approach. The work contributes to the development of the Neural Architecture Search (NAS) field, opening prospects for creating more efficient IoT security models and applying PSO for optimizing architectures of other types of neural networks.

Keywords: particle swarm algorithm, neural networks, multilayer perceptron, Internet of things, optimization, anomaly detection, neural architecture search

For citation: Dobrovolskii D. S., Kozhubaev Yu. N. Optimization of Multilayer Perceptron Architecture Using Particle Swarm Optimization for Enhanced Network Attack Detection on IoT Devices // LETI Transactions on Electrical Engineering & Computer Science. 2026. Vol. 19, no. 2. P. 51–61. doi: 10.32603/2071-8985-2026-19-2-51-61.

Conflict of interests. The authors declare no conflicts of interest.

Введение. Современные системы интернета вещей (IoT) характеризуются экспоненциальным ростом количества подключенных устройств, которые генерируют непрерывные потоки сетевого трафика. Так, по исследованию компании «IoT Analystics» количество подключенных устройств IoT в 2019 г. составляло 10 млрд и с каждым годом наблюдается рост в среднем на 13–15 %. К концу 2023 г. число подключенных устройств интернета вещей достигло 16.6 млрд – это рост на 15 % по сравнению с 2022 г. Исследователи прогнозируют дальнейший экспоненциальный рост подключенных IoT-устройств (ось *y* на рис. 1) до 24.5 млрд в 2026 г. и до 41.1 млрд в перспективе до 2030 г. [1]. Результаты исследования представлены на рис. 1.

Такое количество устройств интернета вещей создает беспрецедентные вызовы в области кибербезопасности. Разнородность экосистемы IoT, которая включает в себя устройства с различными аппаратными возможностями и протоколами связи, усложняет задачу построения единых систем обнаружения вторжений.

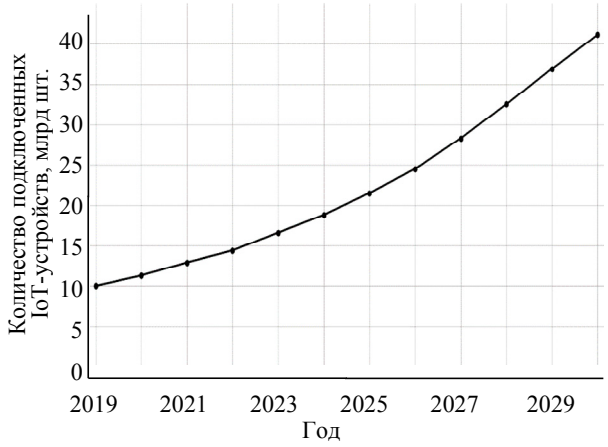

Рис. 1. Результаты исследования количества подключенных IoT-устройств по годам
Fig. 1. Research results on the number of connected IoT devices by year

Особенностью IoT-трафика служит выраженный дисбаланс классов, где легитимные события составляют более 99 % от общего объема данных, в то время как аномальная активность – событие редкое, но критически важное для обнаружения. В таком случае традиционные методы классификации демонстрируют высокую точность лишь за счет корректного распознавания мажоритарного класса, т. е. класса с нормальным сетевым трафиком. Они неэффективны в обнаружении реальных атак на устройства интернета вещей.

Большинство современных исследований в области обнаружения аномалий в IoT-трафике можно разделить на 3 категории:

1. Сигнатурные методы – основаны на сравнении активности IoT-трафика с базой известных паттернов сетевых атак.

2. Статистические методы – используют аппарат статистического анализа для построения профилей нормального поведения системы. Выявление аномалий в них происходит на основе статистических тестов. Применение методов статистики для оценки сетевого трафика, в частности для обнаружения атак типа DDoS (распределенный отказ в обслуживании), позволяет сформировать математическую модель сетевой атаки для потенциальной анализируемой системы. В результате открывается возможность рассмотреть потенциал анализируемой системы, в течение которого она способна выдержать атаку. Это в свою очередь позволяет сформировать потенциал атаки и ее мощность, после чего создать математическую модель для ее обнаружения [2].

3. Методы машинного обучения демонстрируют наилучшие результаты благодаря способности адаптироваться к изменяющимся паттернам. Исследование, проведенное в 2021 г. [3], демонстрирует превосходство использования нейронных сетей для выявления аномалий сетевого трафика над традиционными подходами, поскольку большинство традиционных систем обнаружения вторжений полагаются на базы правил и не могут адаптироваться к новой сетевой среде. Однако эффективность нейросетевых моделей во многом зависит от правильного выбора архитектуры, что традиционно осуществляется через трудоемкий процесс ручного проектирования и эмпирического подбора гиперпараметров.

В контексте нейронных сетей проблема проектирования архитектуры фундаментальна. Недостаточная емкость нейросетевой модели приводит к неспособности уловить сложные зависимости в данных, в то время как ее избыточная сложность вызывает переобучение. Т. е. модель может достичь высокой точности, при этом игнорируя миноритарный класс аномалий.

В данной статье будут проанализированы различные исследования в области построения систем обнаружения аномалий сетевого трафика на устройства интернета вещей с использованием природоподобных алгоритмов оптимизации с целью предложить решение по автоматизированному проектированию архитектуры нейронной сети.

Анализ существующих исследований. В марте 2025 г. группа исследователей опубликовала статью по теме «Повышение эффективности обнаружения ботнетов в интернете вещей с помощью методов роевого интеллекта и нейросетевых архитектур» [4]. В ней предлагается гибридная система обнаружения ботнетов, которая интегрирует алгоритм роя частиц (PSO) с нейросетевыми архитектурами для решения задачи классификации. Однако метод оптимизации на основе алгоритма роя частиц используется лишь для выявления наиболее релевантных признаков из высокоразмерных наборов данных сетевого трафика, в то время как нейронные сети используются для захвата сложных паттернов вторжений. Авторы не рассматривали проблему структурной оптимизации количества слоев и нейронов при проектировании нейросетевой модели.

Аналогичный пример характерен и для исследования «Оценка производительности нейронных классификаторов на основе роевого интеллекта для обнаружения аномалий в интернете вещей» [5], опубликованного в сентябре 2025 г. В этом исследовании изучается возможность интеграции алгоритмов роевого интеллекта с нейронными классификаторами для улучшения точности обнаружения аномалий в устройствах интернета вещей. Однако алгоритмы роевого интеллекта в данном исследовании были использованы лишь для выбора признаков и настройки гиперпараметров нейронных сетей. Для проектирования и оптимизации архитектуры они не применялись.

Проблема автоматизации проектирования нейронных сетей активно исследуется в области под названием Neural Architecture Search (NAS).

Поиск архитектур нейронной сети может быть формально определен как двухуровневая задача оптимизации. Пусть A обозначает пространство поиска архитектур, где каждая архитектура $a \in A$ представляет конкретную конфигурацию нейронной сети. Цель состоит в том, чтобы найти оптимальную архитектуру a^*, которая минимизирует значение функции потерь L_{val} на валидационной выборке:

$$a^* = \arg\min L_{\text{val}}\left(w^*(a), a\right);\ a \in A, \qquad (1)$$

$$w^*(a) = \arg\min L_{\text{train}}\left(w, a\right),$$

где $w^*(a)$ обозначает оптимальные веса для архитектуры a, полученные минимизацией функции потерь на обучении L_{train} [6].

Практическая интерпретация этого процесса выглядит следующим образом:

1. Для каждой кандидатной архитектуры $a_i \in A$:

– обучаем модель с нуля в результате чего получаем оптимальные веса $w^*(a_i)$;

– оцениваем качество обученной модели на валидационной выборке $L_{\text{val}}\left(w^*(a_i), a_i\right)$;

– сохраняем полученный результат валидации.

2. После оценки всех кандидатов (всех возможных вариантов архитектур):

– выбираем архитектуру a^* с наименьшей валидационной ошибкой, согласно (1);

– фиксируем соответствующую ей обученную модель $w(a)$.

Представленная практическая интерпретация свойственна базовой реализации поиска архитектуры нейронной сети.

Рис. 2. Схема алгоритма поиска архитектуры нейронной сети
Fig. 2. Flowchart of the neural network architecture search algorithm

На рис. 2 изображено схематичное представление алгоритма поиска архитектуры нейронной сети [7]: A – это все пространство поиска, т. е. множество всех возможных архитектур, которые может рассмотреть алгоритм, A – это конкретная архитектура, предложенная стратегией поиска.

Существующие подходы к NAS можно разделить на несколько направлений:

1. Методы на основе градиента – они трактуют дискретное пространство архитектур как непрерывное и применяют градиентную оптимизацию. К таким методам относятся DARTS и GDAS.

2. Природоподобные алгоритмы – моделируют естественный процесс для поиска оптимальных архитектур (генетический алгоритм, алгоритм роя частиц и др.).

3. Методы на основе Reinforcement Learning – используют обучение с подкреплением для генерации архитектур. Эти методы характеризуются сложностью реализации, нестабильностью обучения и высокими требованиями к вычислительной инфраструктуре.

4. Байесовская оптимизация – осуществляет поиск оптимальной архитектуры, применяя гауссовские процессы.

Несмотря на достигнутый прогресс в области NAS, большинство методов предназначены для использования на сбалансированных наборах данных. Применение этих методов к задачам с дисбалансом классов, характерным для IoT-трафика, остается малоизученным.

Рассмотренные научные исследования демонстрируют применение природоподобных алгоритмов оптимизации для решения частных задач – например, выбор признаков или настройка гиперпараметров предопределенных архитектур нейронной сети. Однако они не изучают возможность автоматизированного проектирования архитектуры нейронной сети (количество скрытых слоев, распределение нейронов по слоям и др.) с помощью алгоритмов оптимизации.

Рассмотрим работу подобного алгоритма оптимизации роя частиц более подробно.

Алгоритм роя частиц (PSO). Алгоритм роя частиц (Particle Swarm Optimization, PSO) впервые был предложен в 1995 г. Джеймсом Кеннеди и Расселом Эберхартом [8]. Этот метод оптимизации был вдохновлен моделями коллективного поведения птиц в стае и представляет собой многоагентную систему, где агенты – частицы, которые движутся к оптимальным решениям, обмениваясь информацией. Алгоритм роя частиц относится к классу природоподобных метаэвристических методов оптимизации.

Шаги работы алгоритма [9]:

1. Инициализация роя частиц. Алгоритм роя частиц берет свое начало со случайно сгенерированного роя N, D-мерных частиц. Положение каждой частицы x_i^0 на начальной итерации $t = 0$ инициализируется в случайную позицию в пространстве поиска

$$x_i^0 \sim U\left(x_{\min}, x_{\max}\right)^D,$$

где D – размерность пространства поиска или размер решаемой задачи; x_i – частица, которая имеет вектор начального положения x_i^0 со значениями, выбираемыми случайно из равномерно распределенного диапазона $U\left(x_{\min}, x_{\max}\right)^D$, где $x_{\min}$ и $x_{\max}$ – границы пространства поиска.

В общем случае для представления вектора положения i-й частицы на каждой последующей итерации $t \geq 0$ можно использовать обозначение следующего вида:

$$\mathbf{x}_i^t = \left(x_{i,1}^t, x_{i,2}^t, ..., x_{i,D}^t\right),$$

где $\mathbf{x}_i^t$ – вектор положения i-й частицы на итерации t.

Аналогично инициализации положения инициализируется скорость каждой частицы v_i как

$$\mathbf{v}_i^t = \left(v_{i,1}^t, v_{i,2}^t, ..., v_{i,D}^t\right),$$

где $\mathbf{v}_i^t$ – вектор скорости i-й частицы на итерации t.

2. Нахождение наилучшего решения для каждой частицы. Для каждого положения частицы на каждой итерации t вычисляется значение целевой функции $f\left(x_i^t\right)$. Алгоритм роя частиц запоминает индивидуальное лучшее решение, которое когда-либо встречалось для каждой частицы до текущей итерации t, как

$$\mathbf{p}_{\text{best}}^t = \left(p_{\text{best}_1}^t, p_{\text{best}_2}^t, ..., p_{\text{best}_N}^t\right),$$

где значение p_{best}^t представляет индивидуальные лучшие позиции всех частиц в рое на итерации t.

В зависимости от типа решаемой задачи оптимизации (минимизация или максимизация) индивидуальная лучшая позиция p_{best}^t, которую i-я частица посетила до текущей итерации, вычисляется как

$$p_{\text{best}}^t = x_i^l \mid f\left(x_i^l\right) = \min/\max_{k=0,1,...,t}\left(\left\{f\left(x_i^k\right)\right\}\right),$$

где l – тот же индекс, что и k-я итерация, в которой i-я частица нашла наилучшую позицию до текущей итерации t.

3. Нахождение лучшего решения среди всех частиц. По окончании каждой из итераций все решения-кандидаты в p_{best}^t сортируются и первое ранжированное решение выбирается как глобальное (или соседнее) наилучшее решение или позиция g_{best}^t, которую посетила i-я частица-сосед:

$$g_{\text{best}}^t = p_{\text{best}_m}^t \mid f\left(p_{\text{best}_m}^t\right) = \\ = \min/\max_{k=1,2,...,N}\left(\left\{f\left(p_{\text{best}_i}^t\right)\right\}\right),$$

где m – тот же индекс, что и i-я частица, занимающая в целом наилучшее положение среди частиц роя.

4. Коррекция скорости частиц. Скорость частицы обновляется с учетом текущей скорости, влечения к личному лучшему положению и влечения к глобальному лучшему положению:

$$v_{i,j}^{t+1} = \omega v_{i,j}^t + c_1 r_{i,j}^{t+1}\left(p_{\text{best}_{i,j}}^t - x_{i,j}^t\right) + \\ + c_2 r_{2_{i,j}}^{t+1}\left(g_{\text{best},j}^t - x_{i,j}^t\right), \qquad (3)$$

где $\omega v_{i,j}^t$ – инерциальный компонент, который обеспечивает частице импульс для перемещения по пространству поиска; $c_1 r_{1_{i,j}}^{t+1}\left(p_{\text{best}_{i,j}}^t - x_{i,j}^t\right)$ – когнитивный компонент, обозначающий память о предыдущем лучшем индивидуальном положении частицы; $c_2 r_{2_{i,j}}^{t+1}\left(g_{\text{best}_{i,j}}^t - x_{i,j}^t\right)$ – социальный компонент, определяющий текущую производительность, связанную с лучшими глобальными решениями, найденными до сих пор; $j \in 1, 2, ..., D$ – обозначает размеры (или компоненты) i-й частицы.

5. Перемещение частиц. Осуществляется посредством обновления позиции каждой частицы на основе ее текущей скорости после коррекции этой скорости (3) и может быть записано как

$$x_i^{t+1} = x_i^t + v_i^{t+1},$$

где x_i^{t+1} – позиция i-й частицы на следующей итерации; x_i^t – позиция на текущей итерации; v_i^{t+1} – обновленная скорость после учета индивидуального и глобального лучшего положения.

6. Проверка критерия останова. Работа алгоритма завершается при выполнении одного из установленных условий останова. Такими условиями могут стать достижение заданного числа итераций или достижение заданного значения целевой функции, когда значение функции оптимизации становится меньше определенного порога, либо когда выполнены другие критерии останова.

Иначе, происходит возврат к шагу 2.

Блок-схема, отображающая логику и последовательность операций для классического метода реализации алгоритма роя частиц, представлена на рис. 3.

Рассмотрим применимость алгоритма роя частиц для автоматизированного проектирования архитектуры нейронной сети в решении задачи классификации сетевых атак.

Использование алгоритма роя частиц для автоматизированного проектирования архитектуры нейронной сети в задаче классификации сетевых атак на IoT-устройства. Архитектура нейронной сети – это структура и организация сети, определяющая ее компоненты, их взаимосвязи и способ обработки информации. Она включает в себя: типы и количество слоев, способы их соединения, типы используемых функций активации. Архитектура определяет, как нейроны взаимодействуют друг с другом.

Рис. 3. Блок-схема реализации алгоритма роя частиц
Fig. 3. Flowchart of the particle swarm optimization algorithm implementation

В данной части статьи рассмотрим применение алгоритма роя частиц для оптимизации архитектуры нейронной сети на основе многослойного перцептрона для решения задачи классификации сетевых атак.

Перцептрон. Перцептрон Розенблатта – это математическая модель восприятия информации головным мозгом, предложенная Фрэнком Розенблаттом в 1957 г. [10]. Это самая ранняя и простая модель искусственной нейронной сети, применяемая для решения задач классификации, когда объекты классификации могут быть линейно разделимы.

Архитектура перцептрона, описанная Розенблаттом, характеризуется трехуровневой организацией. Прием входных сигналов осуществляют S-элементы (сенсоры). Каждый S-элемент соответствует одному признаку или компоненту входного вектора данных. Обработку и преобразование сигналов осуществляют A-элементы (ассоциативные). Каждый A-элемент связан со случайным подмножеством S-элементов. Он активируется, если взвешенная сумма его входных сигналов превышает определенный порог. Веса связей между S- и A- элементами фиксированы и случайны. Выходной слой из R-элементов (реагирующие) производит окончательную классификацию и отвечает за генерацию выходного отклика. Каждый R-элемент соответствует одному возможному классу. Он суммирует сигналы от всех A-элементов, умноженные на настраиваемые веса, после чего R-элемент с наибольшей взвешенной суммой объявляется победителем и его реакция, например класс 1, соответствует выходному решению системы. Обучение перцептрона заключается именно в итеративной настройке весовых коэффициентов между A- и R- элементами.

Самое известное ограничение перцептрона, описанное в 1969 г., состоит в том, что перцептрон не может решить задачу, если классы не относятся к линейно разделимым [11]. С этой проблемой может справиться многослойный перцептрон.

Многослойный перцептрон (MLP). Многослойный перцептрон (Multilayer Perceptron, MLP) – это класс искусственных нейронных сетей прямого распространения, который состоит как минимум из трех слоев: входного, одного или нескольких скрытых и выходного.

Входной слой многослойного перцептрона принимает исходные данные, скрытые слои выполняют обработку и преобразование информации, а выходной слой формирует результат.

Каждый нейрон многослойного перцептрона связан со всеми нейронами следующего слоя с помощью взвешенных связей, и веса этих связей корректируются в процессе обучения с помощью алгоритма обратного распространения ошибки.

Скрытые слои многослойного перцептрона нелинейно преобразуют входные данные, создавая новые представления, в которых данные уже могут стать линейно разделимыми на выходном уровне. Такой подход позволяет MLP аппроксимировать сложные нелинейные зависимости и справляться с задачами, где классы не имеют линейной границы.

Обзор набора данных CIC IoT-DIAD 2024. В качестве набора данных для обучения искусственной нейронной сети на основе многослойного перцептрона был использован CIC IoT-DIAD 2024 [12] – набор данных двойного назначения для идентификации устройств интернета вещей и обнаружения аномалий, полученный в результате проведенных исследований в Канадском институте кибербезопасности University of New Brunswick. Были проведены 33 отдельные атаки в топологии интернета вещей, включающей 105 устройств, в результате чего сформированы данные по 7 различным категориям атак:

1) DDoS (распределенный отказ в обслуживании);

2) DoS (отказ в обслуживании);

3) Recon (разведка);

4) Web-based (веб-атаки);

5) Brute Force (подбор пароля/ключа);

6) Spoofing (маскировка);

7) Mirai (ботнет).

Все проведенные атаки осуществлялись вредоносными IoT-устройствами и были нацелены на другие IoT-устройства.

Также стоит отметить, что некоторые категории атак из набора данных разделены еще на несколько отдельных атак. Исходное количество классов – 16.

Предварительная обработка набора данных. Перед проведением обучения набор данных был предварительно обработан, так как данные в CIC IoT-DIAD 2024 не размечены построчно и наблюдается серьезный дисбаланс классов. Также были удалены различные категориальные признаки, проведено масштабирование признаков, выполнена нормализация данных и произведена обработка бесконечных значений. Классы веб-атак были сгруппированы в один общий класс Web-based ввиду малого количества записей, после чего результирующее количество классов после группировки составило 14.

В процессе предварительной обработки набора данных целенаправленно не применялись методы борьбы с выбросами. Это обусловлено тем, что для данных сетевого трафика экстремальные значения – это характерные особенности атак и обработка выбросов в таком случае повлечет за собой снижения качества классификации.

Также было проведено кодирование целевой переменной с помощью LabelEncoder, который нормализует метки так, чтобы они содержали только числовые значения от 0 до n. Это необходимо для корректного восприятия данных, так как алгоритмы машинного обучения работают с числами и не понимают текстовые метки в явном виде.

На последнем этапе предварительной обработки набора данных было осуществлено стратифицированное разделение данных на обучающую и тестовую выборки в соотношении 80 к 20 % соответственно.

После предварительной обработки набора данных CIC IoT-DIAD 2024 распределение классов атак представлено в табл. 1.

Табл. 1. Распределение классов после предобработки данных
Tab. 1. Class distribution after data preprocessing

№	Класс	Количество записей
1	DoS_SYN_Flood	30 000
2	DoS_UDP_Flood	30 000
3	DoS_HTTP_Flood	30 000
4	DDoS_HTTP_Flood	30 000
5	Recon	30 000
6	DDoS_ACK_Fragmentation	30 000
7	BenignTraffic	30 000
8	DNS_Spoofing	15 000
9	ARP_Spoofing	15 000
10	DDoS_ICMP_Fragmentation	14 479
11	Web-based	11 328
12	DDoS_ICMP_Flood	10 415
13	Mirai	5172
14	BruteForce	3619

Таким образом, дисбаланс классов атак исходного набора данных был значительно снижен, но не устранен полностью. При дальнейшем анализе и обучении моделей необходимо будет учитывать оставшуюся неравномерность распределения, особенно для классов с минимальным количеством примеров.

Разработка программы. Для проведения обучения искусственной нейронной сети и выявления ее оптимальной архитектуры для решения задачи классификации сетевых атак была разработана программа на языке программирования Python, в которую был интегрирован алгоритм роя частиц для оптимизации архитектуры многослойного перцептрона.

Алгоритм реализации программы по классификации сетевых атак с помощью MLP, архитектура которого оптимизирована с помощью PSO, представлен в виде блок-схемы на рис. 4.

Программа может быть выполнена с использованием как вычислительных ресурсов центрального процессора системы (CPU), так и архитектуры параллельных вычислений CUDA, которая позволяет применять графический процессор (GPU) для выполнения вычислительных задач.

В результате работы программ по классификации сетевых атак в случае, где архитектура для MLP была подобрана эмпирически, и в случае, где архитектура для MLP была подобрана автоматизированно с использованием алгоритма оптимизации роем частиц, были получены результаты, представленные в табл. 2.

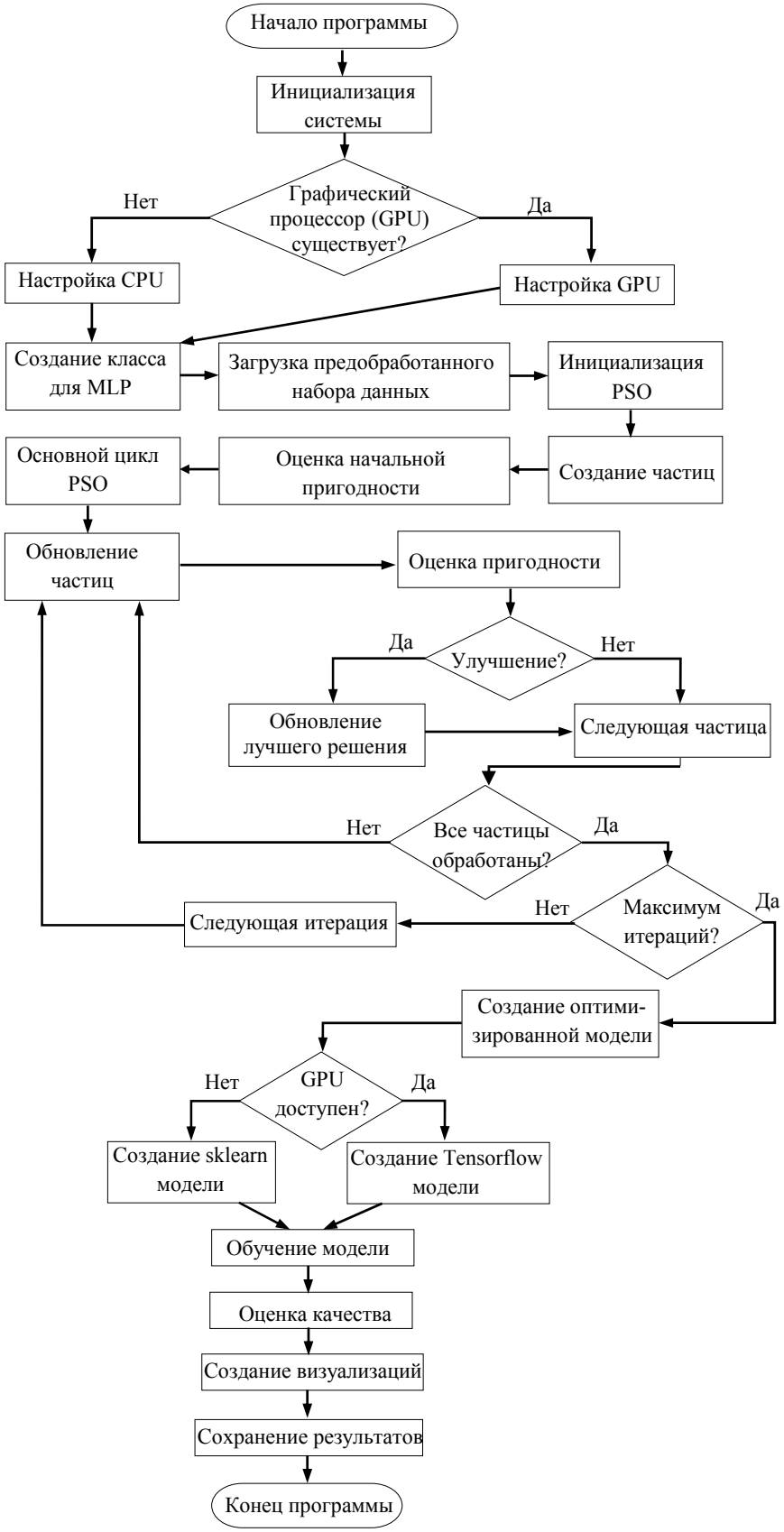

Рис. 4. Блок-схема программы классификации сетевых атак с оптимизацией архитектуры MLP с помощью PSO
Fig. 4. Flowchart of the network attack classification program with MLP architecture optimized using PSO

Табл. 2. Распределение результатов работы программ
Tab. 2. Comparison of program operation results

Метрики оценки	Архитектура MLP подобрана эмпирически	Архитектура MLP оптимизирована PSO
Accuracy	0.7100	0.7408
Precision	0.7128	0.7518
Recall	0.7109	0.7408
F1-Score	0.6878	0.7320
ROC-AUC	0.9600	0.9679

Для оценки качества программ-классификаторов были использованы следующие стандартные метрики:

– Accuracy – доля корректно классифицированных объектов от общего числа объектов;

– Precision – доля верно классифицированных объектов среди всех объектов, которые к этому классу отнес классификатор;

– Recall – отношение верно классифицированных объектов класса к общему числу элементов этого класса;

– F1-Score – это гармоническое среднее между Precision и Recall, используемое для оценки баланса между точностью и полнотой классификации;

– ROC-AUC – интегральная метрика, характеризующая способность модели различать классы при варьировании порога классификации. Метрика не зависит от выбранного порога классификации и устойчива к дисбалансу классов.

Также следует отметить, что реализованные программы решают задачу многоклассовой классификации, однако метрики Precision, Recall, F1-Score, ROC-AUC предназначены для оценки результатов бинарной классификации. В связи с этим для их расчета использовался подход взвешенного усреднения, при котором метрики вычисляются для каждого класса отдельно, а затем усредняются с учетом количества образцов в каждом классе. Для метрики ROC-AUC применялась стратегия One-vs-Rest, где каждый класс сравнивается со всеми остальными с последующим взвешенным усреднением результатов.

Оптимизация архитектуры многослойного перцептрона с помощью алгоритма роя частиц привела к улучшению качества модели по сравнению с эмпирически подобранной архитектурой. Общая доля правильных прогнозов Accuracy увеличилась на 3.08 %. Также модель стала совершать меньше ложноположительных ошибок, о чем свидетельствует метрика Precision. Рост метрики Recall для оптимизированной модели означает, что модель стала пропускать меньше ложноотрицательных результатов. Рост метрики F1-Score на 4.42 % говорит о том, что оптимизированная модель стала более сбалансированной. Согласно метрике ROC-AUC обе модели демонстрируют отличное качество разделения классов, так как значения метрики близки к 1 в обоих случаях.

Высокое значение ROC-AUC при меньших значениях других метрик для обоих программ-классификаторов объяснимо. ROC-AUC оценивает способность каждой модели ранжировать объекты по вероятностям и не зависит от выбранного порога классификации. Расхождение между метриками указывает на хорошую способность модели различать классы, но недостаточную уверенность в конкретных предсказаниях при стандартном пороге классификации. Данное наблюдение открывает перспективы для дальнейших исследований, в частности возможности оптимизации порогов классификации для каждого класса атак индивидуально.

Заключение. В настоящей статье был рассмотрен метод автоматизированного проектирования архитектуры искусственной нейронной сети с использованием алгоритма роя частиц для повышения эффективности обнаружения аномалий в сетевом трафике устройств интернета-вещей.

Экспериментально подтверждено превосходство архитектуры MLP, которая была спроектирована оптимизацией алгоритма роя частиц, над архитектурой MLP, спроектированной эмпирически. Значение метрики точности Accuracy повысилось на 3.08 %, рост Precision составил 3.9 %, Recall вырос с 71 до 74 %. Гармоническое среднее между Precision и Recall выросло на 4.42 %, что говорит о том, что модель стала более сбалансированной. При этом обе модели продемонстрировали отличные способности в разделении классов, о чем свидетельствует значение метрики ROC-AUC.

Таким образом, данные исследования вносят вклад в развитие теории автоматического проектирования нейронных сетей (Neural Architecture Search), демонстрируя эффективность применения алгоритма роя частиц для решения данной задачи. Полученные результаты открывают перспективы для дальнейших исследований, в частности для применения рассмотренного метода автоматизированного проектирования архитектуры MLP к другим типам нейронных сетей и решения более сложных задач классификации.

Список литературы

1. State of IoT 2025: Number of connected IoT devices growing 14 % to 21.1 billion globally. URL: https://iot-analytics.com/number-connected-iot-devices/ (дата обращения: 14.09.2025).

2. Фаткиева Р. Р., Судаков А. С, Нерсисян А. С. Ключевые характеристики сетевого трафика для идентификации DDoS-атаки // Изв. СПбГЭТУ «ЛЭТИ». 2024. Т. 17, № 8. С. 65–80. doi: 10.32603/2071-8985-2024-17-8-65-80.

3. Kuang Ch. Research on network traffic anomaly detection method based on deep learning // J. of Phys.: Conf. Series. 2021. Vol. 1861, no. 1. Art. 012007. P. 1–6. doi: 10.1088/1742-6596/1861/1/012007.

4. Enhancing IoT botnet detection through hybrid sampling techniques and machine learning / M. I. Mohammed, M. A. Ahmad, A. A. Aliyu, A. Saa, I. Mohammed // SLU J. of Sci. and Technol. 2025. Vol. 11, no. 1. P. 52–62.

5. Sur D. Performance evaluation of swarm intelligence-driven neural classifiers for IoT anomaly detection. 2025. URL: https://www.researchgate.net/publication/395715485_Performance_Evaluation_of_Swarm_Intelligence-Driven_Neural_Classifiers_for_IoT_Anomaly_Detection (дата обращения: 16.09.2025).

6. Laurier L. Neural architecture search: Automated machine learning for deep neural networks. 2025. P. 1–17. doi: 10.13140/RG.2.2.16605.93925.

7. Elsken Th., Metzen Ja. H., Hutter F. Neural architecture search. 2019. P. 1–15. doi: 10.1007/978-3-030-05318-5_3.

8. Kennedy J., Eberhart R. Particle swarm optimization // Proc. of ICNN'95 – Intern. Conf. on Neural Networks. Perth, WA, Australia: IEEE, 1995. Vol. 4. P. 1942–1948. doi: 10.1109/ICNN.1995.488968.

9. Казакова Е. М. Применение метода роя частиц в задачах оптимизации // Изв. Кабардино-Балкарского науч. центра РАН. 2022. № 5 (109). С. 48–57. doi: 10.35330/1991-6639-2022-5-109-48-57.

10. Митина О. А., Ломовцев П. П. Перцептрон в задачах бинарной классификации // НАУ. 2021. № 66–1. С. 41–46.

11. Minsky M., Papert S. A. Perceptrons: An introduction to computational geometry. Cambridge, MA, USA: MIT Press, 2017. 316 p. doi: 10.7551/mitpress/11301.001.0001.

12. CIC IoT-DIAD 2024 dataset. URL: https://www.unb.ca/cic/datasets/iot-diad-2024.html (дата обращения: 23.09.2025).

Информация об авторах

Добровольский Дмитрий Сергеевич – студент гр. 5132704/20502, Высшая школа управления киберфизическими системами, Институт компьютерных наук и кибербезопасности. Санкт-Петербургский политехнический университет Петра Великого, ул. Политехническая, д. 29, Санкт-Петербург, 195251, Россия.
E-mail: d.dobrovolskiy4137@gmail.com
http://orcid.org/0009-0008-3581-6612

Кожубаев Юрий Нургалиевич – канд. техн. наук, доцент, доцент Высшей школы управления киберфизическими системами, Институт компьютерных наук и кибербезопасности, Санкт-Петербургский политехнический университет Петра Великого, ул. Политехническая, д. 29, Санкт-Петербург, 195251, Россия.
E-mail: kozhubaev_yun@spbstu.ru
http://orcid.org/0009-0006-1822-7117

References

1. State of IoT 2025: Number of connected IoT devices growing 14 % to 21.1 billion globally. URL: https://iot-analytics.com/number-connected-iot-devices/ (data obrashhenija: 14.09.2025).

2. Fatkieva R. R., Sudakov A. S, Nersisjan A. S. Kljuchevye harakteristiki setevogo trafika dlja identifikacii DDoS-ataki // Izv. SPbGJeTU «LJeTI». 2024. T. 17, № 8. S. 65–80. doi: 10.32603/2071-8985-2024-17-8-65-80. (In Russ.).

3. Kuang Ch. Research on network traffic anomaly detection method based on deep learning // J. of Phys.: Conf. Series. 2021. Vol. 1861, no. 1. Art. 012007. P. 1–6. doi: 10.1088/1742-6596/1861/1/012007.

4. Enhancing IoT botnet detection through hybrid sampling techniques and machine learning / M. I. Mohammed, M. A. Ahmad, A. A. Aliyu, A. Saa, I. Mohammed // SLU J. of Sci. and Technol. 2025. Vol. 11, no. 1. P. 52–62.

5. Sur D. Performance evaluation of swarm intelligence-driven neural classifiers for IoT anomaly detection. 2025. URL: https://www.researchgate.net/publication/395715485_Performance_Evaluation_of_Swarm_Intelligence-Driven_Neural_Classifiers_for_IoT_Anomaly_Detection (data obrashhenija: 16.09.2025).

6. Laurier L. Neural architecture search: Automated machine learning for deep neural networks. 2025. P. 1–17. doi: 10.13140/RG.2.2.16605.93925.

7. Elsken Th., Metzen Ja. H., Hutter F. Neural architecture search. 2019. P. 1–15. doi: 10.1007/978-3-030-05318-5_3.

8. Kennedy J., Eberhart R. Particle swarm optimization // Proc. of ICNN'95 – Intern. Conf. on Neural Networks. Perth, WA, Australia: IEEE, 1995. Vol. 4. P. 1942–1948. doi: 10.1109/ICNN.1995.488968.

9. Kazakova E. M. Primenenie metoda roja chastic v zadachah optimizacii // Izv. Kabardino-Balkarskogo nauchnogo centra RAN. 2022. № 5 (109). S. 48–57. doi: 10.35330/1991-6639-2022-5-109-48-57. (In Russ.).

10. Mitina O. A., Lomovcev P. P. Perceptron v zadachah binarnoj klassifikacii // NAU. 2021. № 66–1. S. 41–46. (In Russ.).

11. Minsky M., Papert S. A. Perceptrons: An introduction to computational geometry. Cambridge, MA, USA: MIT Press, 2017. 316 p. doi: 10.7551/mitpress/11301.001.0001.

12. CIC IoT-DIAD 2024 dataset. URL: https://www.unb.ca/cic/datasets/iot-diad-2024.html (data obrashhenija: 23.09.2025).

Information about the authors

Dmitrii S. Dobrovolskii – student of gr. 5132704/20502, Higher School of Cyber-Physical Systems Management, Institute of Computer Science and Cybersecurity, Peter the Great St. Petersburg Polytechnic University (SPbPU), Polytechnicheskaya St., 29, St. Petersburg, 195251, Russia.
E-mail: d.dobrovolskiy4137@gmail.com
http://orcid.org/0009-0008-3581-6612

Yuriy N. Kozhubaev – Cand. Sci. (Eng.), Associate Professor, Associate Professor at the Higher School of Cyber-Physical Systems Management, Institute of Computer Science and Cybersecurity, Peter the Great St. Petersburg Polytechnic University (SPbPU), Polytechnicheskaya St., 29, St. Petersburg, 195251, Russia.
E-mail: kozhubaev_yun@spbstu.ru
http://orcid.org/0009-0006-1822-7117

Статья поступила в редакцию 02.11.2025; принята к публикации после рецензирования 26.12.2025; опубликована онлайн 26.02.2026.

Submitted 02.11.2025; accepted 26.12.2025; published online 26.02.2026.