УДК 004.85
DOI: 10.53015/18159958_2025_21_3_63

НОВАЯ АРХИТЕКТУРА НЕЙРОСЕТЕЙ KAN И ЕЕ СРАВНЕНИЕ С АРХИТЕКТУРОЙ MLP

THE NEW ARCHITECTURE OF KAN NEURAL NETWORKS AND ITS COMPARISON WITH THE MLP ARCHITECTURE

©2025
Седых Ирина Александровна, доктор технических наук, профессор кафедры автоматизированных систем управления
Струков Илья Владимирович, студент кафедры автоматизированных систем управления
Sedykh Irina Alexandrovna, Doctor of Technical Sciences, professor of the Department of Automated control systems
Strukov Ilya Vladimirovich, student of the Department of Automated control system

Липецкий государственный технический университет, Липецк (Россия)
Lipetsk State Technical University, Lipetsk (Russia)

E-mail: sedykh-irina@yandex.ru ORCID: https://orcid.org/0000-0003-0012-8103
E-mail: melee-chess@mail.ru

Аннотация: В статье рассматривается новая архитектура нейросетей KAN (Kolmogorov – Arnold Networks). Проводится сравнительный анализ с популярной архитектурой MLP (Multi-Layer Perceptron) по теоретическим основам, архитектурным различиям, эффективности и способах обучения.

Ключевые слова: нейронная сеть, многослойный перцептрон, нейросеть Колмогорова – Арнольда, обучение, аппроксимация функции, оптимизатор, сплайн.

Для цитирования: Седых И.А. Новая архитектура нейросетей KAN и её сравнение с архитектурой MLP / И.А. Седых, И.В. Струков // Вести высших учебных заведений Черноземья. – 2025. – Т. 21, № 3 (79). – С. 63–70. – DOI 10.53015/18159958_2025_21_3_63.

Abstract: The article discusses the new architecture of KAN neural networks (Kolmogorov – Arnold Networks). A comparative analysis is conducted with the popular ML (Multi-Layer Perceptron) architecture in terms of theoretical foundations, architectural differences, efficiency, and learning methods.

Keywords: neural network, Multi-Layer Perceptron, Kolmogorov – Arnold network, learning, approximation of function, optimizer, spline.

For citation: Sedykh I.A. The new architecture of KAN neural networks and its comprasion with the MLP architecture / I.A. Sedykh, I.A. Strukov // Vesti of Higher Educational Institutions of the Chernozem region. – 2025. – Vol. 21, № 3 (79). – P. 63–70. DOI 10.53015/18159958_2025_21_3_63.

ВВЕДЕНИЕ

В последние годы машинное обучение активно развивается, появляются новые архитектуры нейронных сетей, способные эффективно решать сложные задачи. Среди классических подходов особое место занимает многослойный перцептрон (англ.

Multilayer Perceptron, сокр. MLP), который благодаря своей универсальности и простоте широко применяется в различных областях. Однако в 2024 году появилась новая архитектура – нейросеть Колмогорова – Арнольда (англ. Kolmogorov – Arnold Network, сокр. KAN), основанная на теореме Колмогорова – Арнольда о представлении функций как сумму суперпозиций других функций [1]. KAN предлагает альтернативный подход к аппроксимации сложных зависимостей, потенциально превосходящий традиционные MLP по точности и интерпретируемости [1–3].

В статье рассмотрена архитектура KAN и проведено сравнение ее с MLP.

АРХИТЕКТУРА MLP

MLP – одна из базовых архитектур нейронных сетей, широко применяемая для задач регрессии, классификации и аппроксимации функций [4]. MLP относится к нейронным сетям с прямой связью, то есть соединения между узлами не образуют циклов [5]. Архитектура основана на теореме Цыбенко, которая доказывает, что перцептрон может аппроксимировать любую непрерывную функцию с любой точностью [6].

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ НЕЙРОНА В MLP

Каждый нейрон вычисляет сумму входов с добавлением смещения и применяет функцию активации [4]:

$$y = f(u), \quad (1)$$

где f – функция активации, u имеет вид:

$$u = \sum_{i=1}^{n} w_i x_i + w_0 x_0, \quad (2)$$

где x_i – сигналы на входах нейронов, w_i – веса входов, x_0 – дополнительный вход для инициализации нейрона, w_0 – вес дополнительного входа.

СТРУКТУРА MLP

MLP состоит из трех основных типов слоев: входного, скрытого и выходного. Входной слой принимает исходные данные. Количество нейронов соответствует размерности входных данных. Скрытые слои выполняют нелинейные преобразования. Выходной слой формирует итоговый результат.

На рис.1 представлена модель MLP.

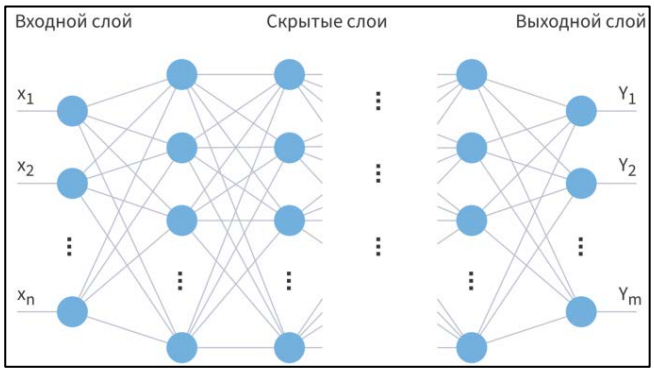

Рис. 1. Модель MLP
Fig. 1. Model of MLP

ДОСТОИНСТВА И НЕДОСТАТКИ АРХИТЕКТУРЫ MLP

У архитектуры MLP есть два основных достоинства: универсальность и простота реализации. Недостатками являются чувствительность к количеству слоёв и нейронов в них, склонность к переобучению [6]. Эти ограничения особенно заметны при работе с небольшим набором данных [7].

ОБУЧЕНИЕ MLP И АРХИТЕКТУРА KAN

Перцептрон обучается через минимизацию функции потерь с помощью трех ключевых процессов: прямого распространения (входные данные последовательно проходят через все слои), обратного распространения ошибки (вычисляются антиградиенты для каждого параметра) и обновления весов с помощью оптимизаторов [4].

Популярными оптимизаторами являются стохастический градиентный спуск (англ. Stochastic Gradient Descent, сокр. SGD) и Adam (англ. Adaptive Moment Estimation). Выбор оптимизатора существенно влияет на обучение [7]. В табл.1 приведено сравнение оптимизаторов SGD и Adam.

В 2024 году исследователи из MIT представили новую архитектуру – нейросеть

Колмогорова – Арнольда, в которой функции активации перемещены с нейронов на рёбра сети [1, 2, 8, 9]. Данная архитектура основана на теореме Колмогорова – Арнольда, доказанная советскими математиками А.Н. Колмогоровым и В.И. Арнольдом.

Таблица 1

Сравнение оптимизаторов

Table 1

Comparison of optimizers

Критерий	SGD	Adam
Скорость	Медленнее	Быстрее
Точность	Лучше для выпуклых функций	Лучше для невыпуклых функций
Ресурсы	Меньше памяти	Больше памяти

ТЕОРЕМА КОЛМОГОРОВА – АРНОЛЬДА

Теорема Колмогорова – Арнольда показывает, что единственная истинная функция многих переменных – это сложение, поскольку все другие функции можно записать с использованием функций одной переменной и сложения [1]. Данную теорему можно записать в виде формулы:

$$f(x)=f(x_1,\ldots,x_n)=\sum_{q=1}^{2n+1}\Phi_q\left(\sum_{p=1}^{n}\phi_{q,p}(x_p)\right),(3)$$

где Φ_q, $\phi_{q,p}$ — непрерывные одномерные функции, x_p – p-ая входная переменная.

По сравнению с теоремой Цыбенко, по теореме Колмогорова – Арнольда для создания одной большой функции связи между входами и выходами сети необходимы обычные одномерные функции, количество которых растет не экспоненциально, а полиномиально.

СТРУКТУРА KAN

KAN как и MLP состоит из трех основных типов слоев: входного, скрытого и выходного. Входной слой принимает исходные данные аналогично MLP. Скрытые слои реализуют декомпозицию функции по теореме Колмогорова – Арнольда. Выходной слой формирует итоговый результат через композицию базовых функций – сплайнов, обычно – это кубическая функция [1, 8].

Ключевое отличие заключается в перемещении функции активации на рёбра сети и организацией в нейронах операции суммирования [1].

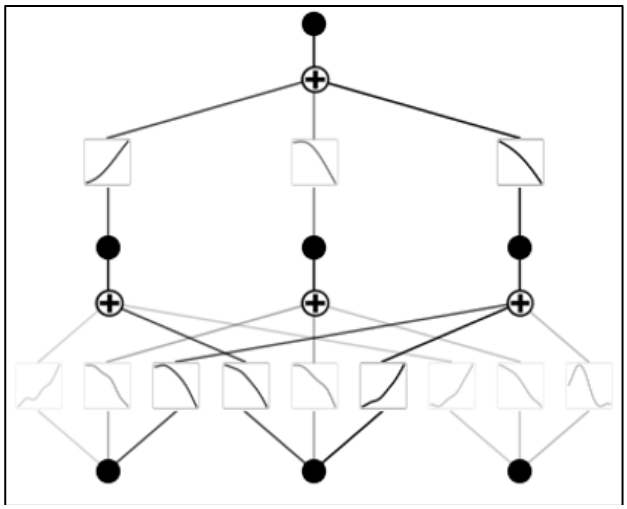

Рис. 2. Модель KAN

Fig. 2. Model of KAN

На рис. 2 представлена модель KAN с 3 входами, 1 скрытым слоем с 3 нейронами и с 1 выходом.

ДОСТОИНСТВА И НЕДОСТАТКИ KAN

Потенциально архитектура KAN имеет более высокую точность с MLP. Эксперименты показали, что KAN достигает более высокой точности на малых выборках (MSE на 15–20% ниже, чем у MLP) [3].

Недостатки: сложность реализации и большая вычислительная сложность (в сравнении с MLP в 3–5 раз больше потребляет оперативной памяти) [3].

Обучение KAN аналогично MLP, однако есть несколько особенностей: антиградиенты рассчитываются относительно контрольных точек сплайнов, учитывается гладкость, антиградиенты распространяются не только «в глубину», но и «по ширине» – между параллельными функциями.

ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

Для сравнительного анализа архитектур MLP и KAN на языке программирования Python [4, 5, 6, 10, 11] с использованием следующих библиотек: для MLP – TensorFlow и Keras [12, 13, 15], для KAN – PyTorch и PyKAN [9, 14].

Для обеих моделей использовалась одна и та же синтетическая функция:

$$f(x_0,x_1,x_2)= \\ = \sin\left(\sin(x_0)+x_1^2+\cos(x_2)\right).$$

Реализация MLP (для устойчивости была выбрана функция активации ELU [4, 5, 6]):

```
size_layer = 25
size_epochs = 100
count_layers = 12
# Создание модели (3 входа, 12 скрытых слоёв с 25 нейронами, 1 выход)
model = Sequential()
model.add(layers.Dense(units=size_layer, input_dim=3, activation="elu"))
for i in range(count_layers):
    model.add(layers.Dense(units=size_layer, activation="elu"))
model.add(layers.Dense(units=1))
# Выбор минимизируемого параметра и оптимизатора
model.compile(loss='mean_squared_error', optimiz-
er=keras.optimizers.Adam(), metrics=['mean_squared_error'])
# Обучение
history = model.fit(x_train, trainset, epochs=size_epochs, batch_size=32)
```

Реализация модели KAN (параметр grid, контролирующий детализацию сплайнов, был установлен экспериментально, оптимизатор LBFGS выбран из-за эффективности при оптимизации параметров сплайнов [8, 9, 3]):

```
# Возможность использовать ресурсы видеокарты
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
# Создание модели (3 входа, 15 нейронов в 1 скрытом слое, 1 выход)
model = KAN(width=[3,15,1], grid=5, k=3, seed=35, device=device)
# Обучение
history = model.fit(dataset, opt="LBFGS", steps=25, lamb=0.001)
```

СРАВНЕНИЕ MLP И KAN

Были проведены исследования при обучении KAN и MLP с поиском структуры MLP и KAN, которые дают в результате обучения минимальные ошибки MSE. Рассматривались следующие параметры для MLP: количество слоёв от 1 до 15, количество нейронов в слое, количество эпох и оптимизаторы SGD и Adam. Для KAN: количество слоёв от 1 до 2, количество нейронов в слое, количество эпох.

На рис. 3 для тестовой выборки представлены графики ошибок MSE, MAPE, времени обучения и RAM для MLP с оптимизатором Adam. Количество нейронов в слое: 12. Количество слоёв варьируется от 1 до 15. На рис. 4 представлены графики ошибок MSE, MAPE, времени обучения и RAM для KAN с оптимизатором LBFGS. Количество слоёв: 1. Количество нейронов в слое варьируется от 1 до 15.

Найдены следующие оптимальные параметры для моделей: для MLP – 12 слоёв по 25 нейронов в каждом при 100 эпохах обучения с оптимизатором Adam, для KAN – 1 слой с 15 нейронами при 25 эпохах с оптимизатором LBFGS.

На рис. 5 показаны результаты сравнения. Эксперименты показали, что KAN демонстрирует более высокую точность (снижение MSE на 15–20 %) по сравнению

с MLP при работе с малыми объёмами данных (до 500 примеров). Это особенно заметно при аппроксимации гладких и периодических функций, таких как тригонометрические зависимости.

Однако при увеличении объёма обучающей выборки свыше 500 примеров преимущество переходит к MLP, который показывает лучшую масштабируемость и стабильность результатов.

Анализ ресурсопотребления выявил существенные различия: нейросеть KAN требует до 1,4 Гб памяти с линейным ростом потребления при увеличении данных, в то время как MLP сохраняет стабильное потребление на уровне 0,5 ГБ независимо от размера выборки. Временные затраты на обучение KAN в 2-3 раза превышают аналогичные показатели MLP для одинаковых объёмов данных.

ЗАКЛЮЧЕНИЕ

В работе были рассмотрены архитектуры нейросетей MLP и KAN, и показана эффективность каждой из архитектур при аппроксимации многомерной функции. KAN демонстрирует преимущество на малых объёмах данных, тогда как MLP требует большего количества примеров для достижения сопоставимой точности.

Модель KAN удобна для работы с периодическими и гладкими функциями, такими как синус, косинус.

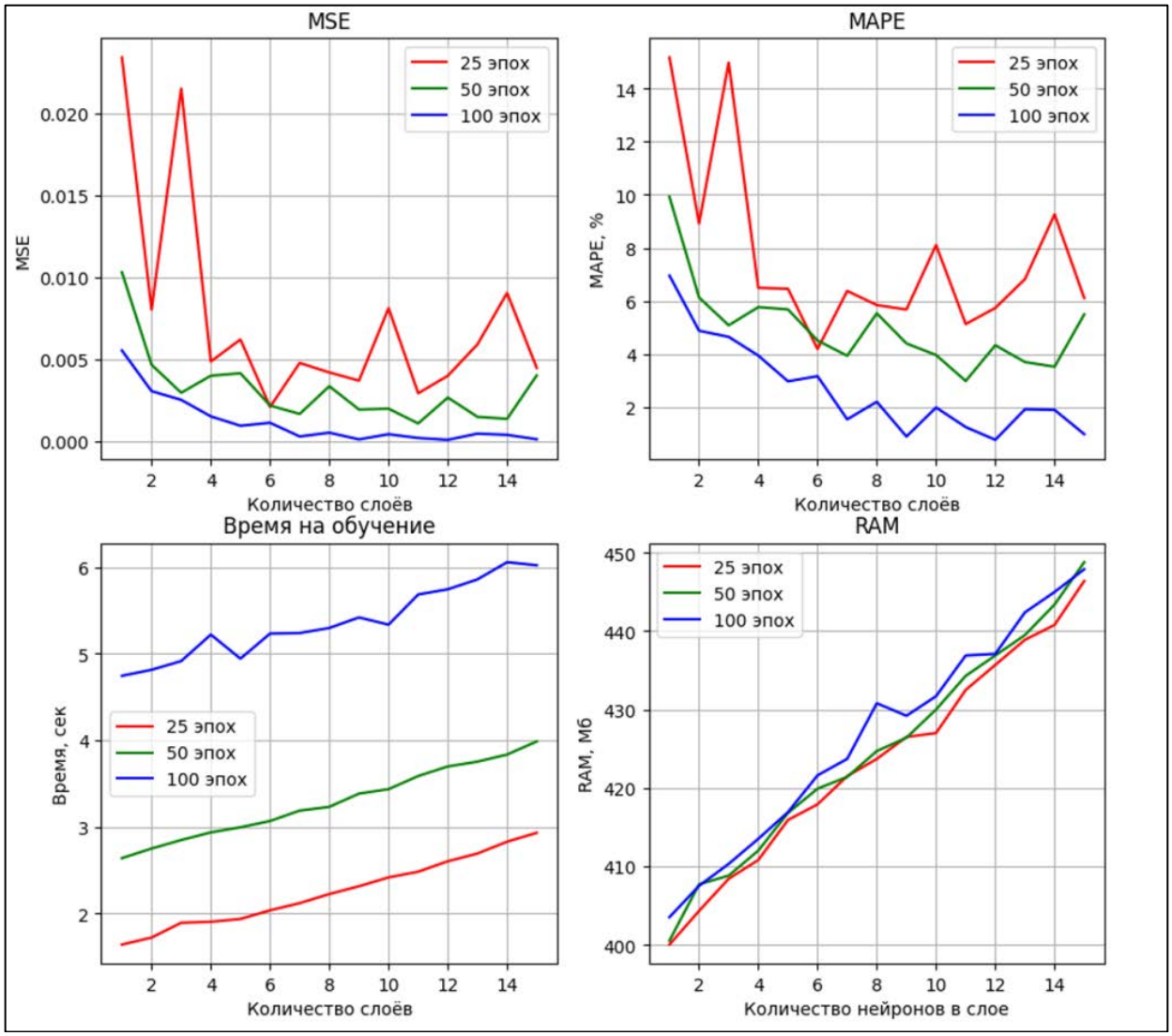

Рис. 3. Результаты тестов MLP Adam с 25 слоями
Fig. 3. Results of tests MLP Adam with 25 layers

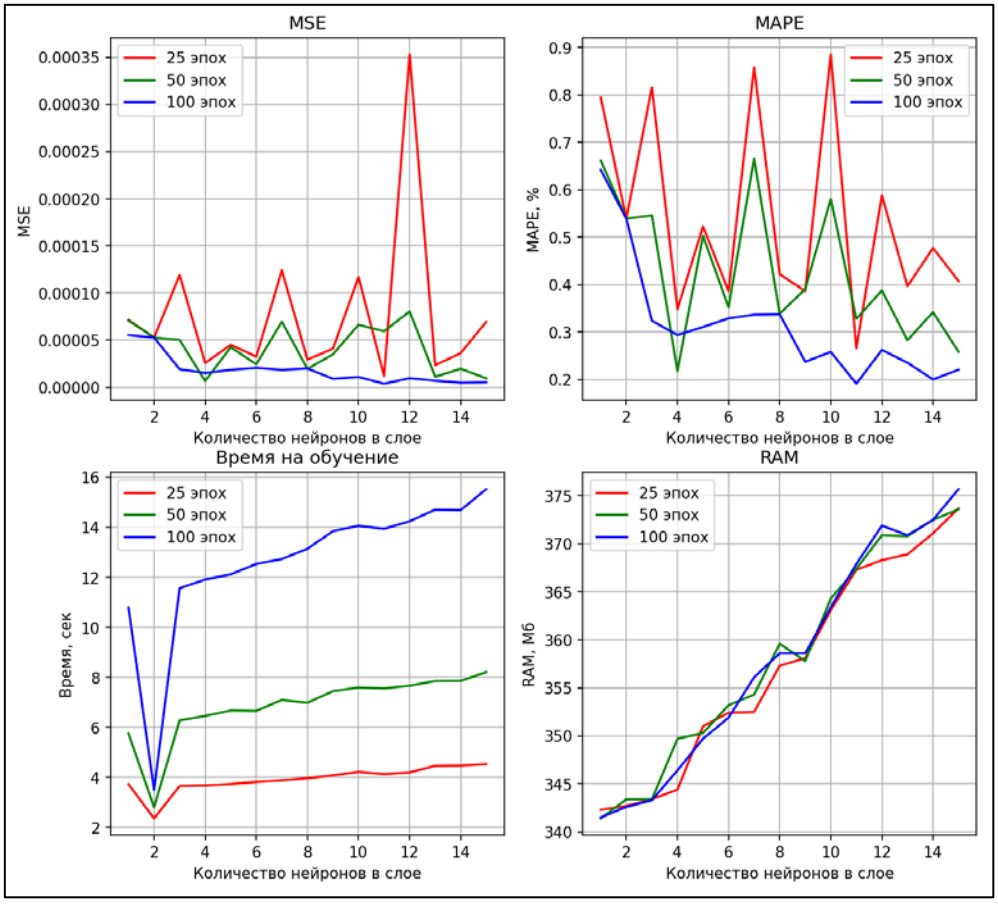

Рис. 4. Результаты тестов KAN
Fig. 4. Results of tests KAN

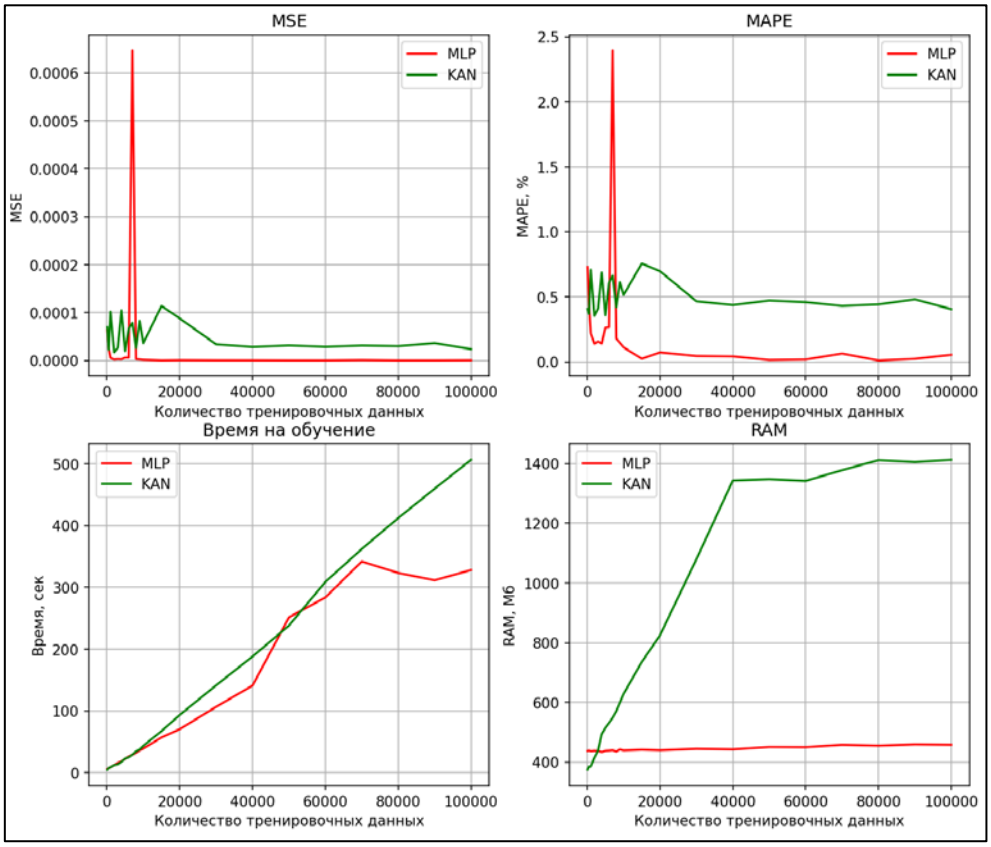

Рис. 5. Сравнение MLP и KAN
Fig. 5. Comparing MLP and KAN

СПИСОК ЛИТЕРАТУРЫ

1. KAN: Kolmogorov-Arnold Networks / Z. Liu, Y. Wang, S. Vaidya [et al.]. Tekst: electronic. URL: https://arxiv.org/abs/ 2404.19756 (date of access: 01.06.2024).
2. Разбор статьи про KAN – принципиально новую архитектуру нейросетей. Текст: электронный // DataSecrets : сайт. URL: https://datasecrets.ru/articles/9 (дата обращения: 10.06.2024).
3. EfficientKAN GitHub repository. Tekst: electronic // GitHub : website. URL: https://github.com/Blealtan/efficient-kan (дата обращения: 03.06.2024).
4. Рашид Т. Создаем нейронную сеть. Санкт-Петербург : Альфа-книга, 2017. 272 с.
5. Иванов А. Введение в нейронные сети: от перцептрона до трансформеров. Текст: электронный // ХАБР : сайт. URL: https://habr.com/ru/articles/815851/ (дата обращения: 10.09.2024).
6. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima / Keskar N.S., Mudigere D., Nocedal J. [et al.] // Proceedings of ICLR. 2017.
7. Allen-Zhu Z., Li Y., Song Z. A Convergence Theory for Deep Learning via Over-Parameterization // Proceedings of ICML. 2019.
8. PyKAN: Python Kolmogorov-Arnold Networks : website. URL: https:// kindxiaoming.github.io/pykan/ (дата обращения: 01.06.2024).
9. PyKAN GitHub repository : website. URL: https://github.com/KindXiaoming/ pykan (дата обращения: 01.06.2024).
10. Златопольский Д.М. Основы программирования на языке Python. Москва : ДМК Пресс, 2017. 284 с.
11. Федоров Д.Ю. Программирование на языке высокого уровня Python : учебное пособие для прикладного бакалавриата. 2-е изд., перераб. и доп. Москва : Юрайт, 2019. 161 с. ISBN 978-5-534-10971-9. URL: https://urait.ru/ bcode/437489 (дата обращения: 10.07.2024).
12. TensorFlow API Documentation : website. URL: https://www.tensorflow.org/ api_docs/python/tf (дата обращения: 10.09.2024).
13. Keras documentation : Examples : website. URL: https://keras.io/examples/ (дата обращения: 15.09.2024).
14. PyTorch Documentation : website. URL: https://docs.pytorch.org/docs/stable/index.html (дата обращения: 05.06.2024).
15. Полное руководство по Keras: от основ до продвинутых техник. Текст: электронный // Рег.ру : облачная платформа. URL: https://www.reg.ru/blog/keras/ amp/ (дата обращения: 15.09.2024).

REFERENCES

1. KAN: Kolmogorov-Arnold Networks / Z. Liu, Y. Wang, S. Vaidya [et al.]. Text: electronic. URL: https://arxiv.org/abs/ 2404.19756 (date of access: 01.06.2024).
2. Analysis of an article about KAN, a fundamentally new architecture of neural networks. Tekst: electronic // DataSecrets : website. URL: https://datasecrets.ru/ articles/9 (accessed: 10.06.2024).
3. EfficientKAN GitHub repository // GitHub : website. URL: https://github.com /Blealtan/efficient-kan (date of access: 03.06.2024).
4. Rashid T. Creating a neural network. Saint Petersburg : Alpha-kniga, 2017. 272 p.
5. Ivanov A. Introduction to neural networks: from perceptron to transformers/ Tekst: electronic // ХАБР : сайт. URL: https://habr.com/ru/articles/815851 / (date of access: 10.09.2024).
6. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima / Keskar N.S., Mudigere D., Nocedal J. [et al.] // Proceedings of ICLR. 2017.
7. Allen-Zhu Z., Li Y., Song Z. A Convergence Theory for Deep Learning via Over-Parameterization // Proceedings of ICML. 2019.
8. PyKAN: Python Kolmogorov-Arnold Networks : website. URL: https:// kindxiaoming.github.io/pykan/ (date of access: 01.06.2024).
9. PyKAN GitHub repository : website. URL: https://github.com/KindXiaoming/ pykan (date of reference: 01.06.2024).

10. Zlatopolsky D.M. Fundamentals of programming in Python. Moscow : DMK Press, 2017. 284 p.
11. Fedorov D.Y. Programming in the high-level Python language : a textbook for applied bachelor's degree. 2nd ed., revised and additional. Moscow : Yurait, 2019. 161 p. ISBN 978-5-534-10971-9. URL: https://urait.ru/bcode/437489 (date of request: 10.07.2024).
12. TensorFlow API Documentation : website. URL: https://www.tensorflow.org/api_docs/python/tf (date of access: 10.09.2024).
13. Keras documentation: Examples : website. URL: https://keras.io/examples / (date of access: 15.09.2024).
14. PyTorch Documentation : website. URL: https://docs.pytorch.org/docs/stable/index.html (date of access: 05.06.2024).
15. The complete guide to Keras: from basics to advanced techniques Tekst: electronic. // Per.py : cloufu platform. URL: https://www.reg.ru/blog/keras/amp / (date of access: 15.09.2024).