# Оптимизация процесса обучения при ограниченном объеме вычислительных ресурсов

Н. С. Мокрецов

*Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)*

mokrecovnikita6374@gmail.com

Т. М. Татарникова

*Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)*

tm-tatarn@yandex.ru

***Аннотация.*** **Рассматривается проблема оптимизации моделей глубокого обучения для их использования в приложениях реального времени на устройствах с ограниченными ресурсами. Для решения этой проблемы предлагается использовать дистилляцию знаний, которая заключается в переносе знаний из точной, но громоздкой модели нейронной сети в более компактную. Описываются различные архитектуры такого подхода, их недостатки и преимущества. Приведён анализ эффективности дистилляции знаний с выявлением важных закономерностей для достижения желаемого результата при использовании такого подхода. Описывается реализация и результаты применения применительно к свёрточным нейронным сетям визуальной классификации.**

***Ключевые слова: нейронные сети, глубокое обучение, оптимизация модели, дистилляция знаний***

## I. Введение

Применение алгоритмов глубокого обучения стало основой многих успехов в области искусственного интеллекта, включая различные приложения в области компьютерного зрения, обучения с подкреплением и обработки естественного языка. Благодаря множеству новейших методов, таких как остаточные соединения и пакетная нормализация, стало легко тренировать модели с миллиардами параметров на мощных кластерах графических процессоров (Graphics Processing Unit, GPU) или тензорных процессоров (Tensor Processing Unit, TPU).

Например, требуется менее десяти минут, чтобы обучить модель ResNet – глубокую сверточную нейронную сеть распознавания образов на популярном датасете, содержащем миллионы изображений. Требуется не более полутора часов, чтобы обучить мощную модель BERT для решения задач, основанных на обработке естественных языков.

Несмотря на то, что крупномасштабные модели глубокого обучения достигли ошеломляющих успехов, их огромная вычислительная сложность и требования к размерам хранилищ делают их использование в приложениях реального времени большой проблемой, особенно на устройствах с ограниченными ресурсами, таких как системы видеонаблюдения и автономные автомобили.

Для решения этой проблемы сообществом было предложено большое количество методов, алгоритмов и архитектур моделей, оптимизирующих её размеры. Чаще всего используются различные модификации алгоритмов удаления лишних нейронных связей из слоёв сети или повторное использование параметров для разных задач. Такие подходы в общем случае помогают достичь желаемых результатов, но обладают рядом недостатков, таких как уменьшение устойчивости модели к шуму, непредсказуемому уровню потери информации, отсутствием универсальности, либо невозможности применения таких подходов в конкретном случае.

## II. Дистилляция знаний

В настоящее время, в качестве подхода для оптимизации модели нейронной сети, особое внимание уделяется идее дистилляции знаний. Суть данного подхода заключается в переносе знаний из точной, но громоздкой модели нейронной сети (учителя) в более компактную (ученика), с учётом ограничения вычислительных ресурсов или размеров чипа, на которых планируется запускать оптимизированную версию модели. На данный момент предложено большое количество различных подходов реализации дистилляции знаний. С точки зрения архитектуры модели, они делятся на два вида: основанные на выводе модели и основанные на внутренней структуре весов нейронной сети.

В случае архитектуры, основанной на выводе модели, сравниваются «логиты» учителя и ученика при одних и тех же входных данных. Схематично это представлено на рис. 1.

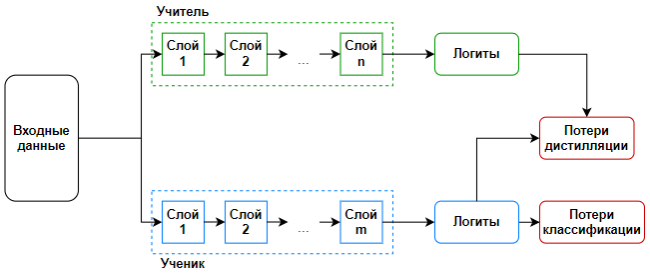


Рис. 1. Архитектура дистилляции знаний, основанная на выводе модели

Логит – это логарифмическая функция, используемая для преобразования вероятности в линейный интервал. В нейронах сети классификации, логиты представляют собой выходные значения последнего слоя сети, которые представляют собой неотличимые вероятности классов.

Эти значения используются для предсказания класса объекта и обычно преобразуются в вероятности с помощью функции активации softmax, которая гарантирует, что все вероятности суммируются до 1.

Одной из особенностей решений с данной архитектурой является то, что в большинстве случаев, для достижения желаемой точности модели ученика, обучение должно происходить в течении большого количества эпох, а логиты моделей должны быть основаны на одних и тех же изображениях. Однако, они отличаются универсальностью, так как не подразумевают изменений во внутренней структуре нейронной сети или анализ отдельных частей её скрытых слоёв, которые имеют уникальный характер для каждой модели или семейства моделей.

Архитектура, в которой дистилляция знаний реализуется на уровне внутренней структуры нейронной сети, использует передачу знаний в рамках весов, карт активаций и прочих свойств весов скрытых слоёв нейронной сети учителя и ученика (рис. 2).

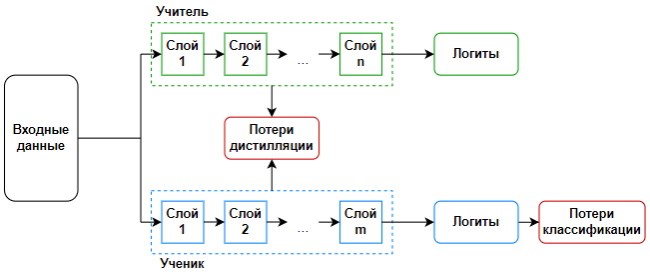


Рис. 2. Архитектура дистилляции знаний, основанная на внутренней структуре модели

Подходы дистилляции знаний, основанные на внутренней структуре модели, предоставляют ученикам дополнительную информацию путем оптимизации карт активации промежуточного слоя (intermediate activation maps) учеников, чтобы они были похожи на карты учителя. В данном случае, необходимо сделать конкретный выбор относительно того, какие слои модели учителя и ученика должны соответствовать друг другу, а, следовательно, эти подходы зависят от архитектуры используемых моделей, что усложняет достижение желаемого уровня универсальности данного подхода.

### *A. Достоверность*

Подробный анализ идеи дистилляция знаний выявил важные закономерности для достижения необходимого результата при использовании описываемого подхода [5]. Зачастую, происходит так, что несмотря на то, что дистилляция знаний может повысить качество обобщения (способность модели делать правильные предсказания на новых данных после завершения обучения) модели ученика, не достигается основная цель, заложенная в идею дистилляции знаний – остается большое расхождение между предсказательными распределениями учителя и ученика. Подобная ситуация наблюдается даже в тех случаях, когда ученик способен идеально соответствовать учителю.

Для обозначения свойства соответствия предсказаний ученика предсказаниям учителя используется термин достоверности (fidelity). Анализ моделей дистилляции знаний, предшествующих описываемому исследованию, показывает, что уровень достоверности ученика в них достаточно мал.

Авторами были описаны причины, по которым уровень достоверности ученика играет важную роль для достижения желаемого уровня метрик его модели. Повышение достоверности ученика является наиболее очевидным способом уменьшения уровня расхождения обобщений в модели учителя и модели ученика. Между достоверностью и обобщением присутствует явная корреляция, как показано на рис. 3, в следствии чего можно сделать вывод, что повышение достоверности играет ключевую роль даже в том случае, когда разработчика модели интересует исключительно качество её обобщений.

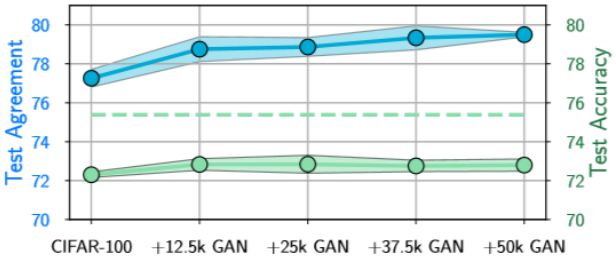


Рис. 3. Показатели достоверности и точности модели ученика [5]

Кроме того, повышение уровня достоверности улучшает модель с точки зрения интерпретируемости и надёжности. Способность выполнять перенос представлений данных из больших моделей, которые в меру своей громоздкости являются «черными ящиками», в более простые интерпретируемые модели могут способствовать выявлению закономерностей в данных, которые являются неочевидными.

### *B. Визуальные объяснения*

В настоящее время используется разновидность подходов дистилляции знаний с архитектурой, использующей выводы нейронной сети, построенных на идее визуальных объяснений. Такое решение повышает уровень достоверности ученика за счёт ввода в модель дополнительной информации о том, почему учителем было принято конкретное решение.

Одним из методов получения визуального объяснения является Gradient-weighted Class Activation Mapping (Grad-CAM) [6]. Суть метода заключается в использовании градиентов какого-либо целевого концепта, например, для модели классификации объекта на изображении – «кот», протекающих в последний свёрточный слой нейронной сети, для последующего построения локализирующей карты, которая подсвечивает область изображения, за счет которой объекту в области изображения присваивается определённый класс. За счет локализирующих карт мы получаем информацию о дискриминативных областях входного изображения – важные части изображения, которые используются для классификации или распознавания объектов.

Этот метод позволяет выявить важные нейроны, которые после присвоения им имён [7] предоставляют человекочитаемые текстовые объяснения решений модели, которые позволяют разработчику проанализировать её внутреннюю структуру.

Отличительной чертой Grad-CAM в сравнении со своими предшественниками является универсальность, с точки зрения применимости к любой свёрточной нейронной сети без внесения изменений в их архитектуру или повторного переобучения.

Для получения класс-дискриминационной локализирующей карты методом Grad-CAM для выбранного класса $L^c_{\text{Grad-CAM}}$, применяется следующая последовательность вычислений. Сначала вычисляется отношение градиента оценки для класса – вектор, который показывает, насколько изменение входных данных влияет на оценку для определенного класса, к *k*-той свёрточной карте признаков. Оценка для класса – это число, которое показывает, насколько вероятно, что входное изображение принадлежит определенному классу. По итогу, получается частичная линеаризация $\alpha_k^c$ – вес, показывающий важность выбранной карты признаков $A^k$ для класса *c*

$$\alpha_k^c = \frac{1}{HW}\sum_i^H\sum_j^W \frac{\partial y^c}{\partial A_{ij}^k},$$

где *c* – индекс класса; *k* – индекс карты признаков; *H* – высота карты признаков в пикселях; *W* – ширина карты признаков в пикселях; $y^c$ – оценка вероятности класса *c* (вывод модели до применения softmax); *A* – карта признаков.

После этого, по отношению ко всем картам признаков в нейронной сети, получается взвешенная комбинация с последующим применением к ней ReLU

$$L^c_{\text{Grad-CAM}} = \text{ReLu}\left(\sum_k \alpha_k^c A^k\right). \quad (1)$$

В результате получаем тепловую карту того же размера, что и карты активации. Применение ReLU важно для выделения только тех пикселей, значимость которых должна быть повышена для присваивания изображению правильного класса. Без применения ReLU локализирующие карты начинают выделять лишние части изображения, которые не относятся области, на котором находится объект нужного класса.

### *C. Улучшенная дистилляция знаний с объяснениями*

Для улучшения результатов дистилляции знаний, с учетом описанных выше проблем, связанных с достоверностью модели ученика, было предложено решение улучшенной дистилляции знаний, использующие визуальные объяснения для настройки более точного соответствия модели учителя – explanation-enhanced Knowledge Distillation – $\left(e^2KD\right)$ [8].

Для достижения этого, оптимизируется не только классическая потеря дистилляции знаний, но и схожесть в объяснениях моделей учителя и ученика. Применение такого подхода позволяет увеличить точность ученика и соответствие между его моделью и моделью учителя, убедиться, что ученик даёт схожие объяснения, то есть выдаёт правильные ответы по тем же причинам, что и учитель. Более того, этот метод независим от деталей конкретной архитектуры моделей, количества данных для обучения и позволяет использовать вычисленные заранее объяснения учителя, что ускоряет процесс обучения нейронной сети.

Для увеличения точности также используется модель объяснений GradCAM.

Для обучения модели используется следующая функция потерь:

$$\Im = \Im_{KD} + \lambda\Im_{\text{exp}},$$

где $\Im_{KD}$ – классическая функция потерь дистилляции знаний; $\Im_{\text{exp}}$ – потери схожести объяснений; $\lambda$ – весовой коэффициент потери схожести объяснений.

Функция потерь классической дистилляции знаний минимизирует KL-дивергенцию между учителем (*T*) и учеником (*S*):

$$\Im_{\text{KD}} = -\tau^2\sum_{j=1}^n \sigma_j\left(\frac{z_T}{\tau}\right)\log\sigma_j\left(\frac{z_S}{\tau}\right),$$

где $\tau$ – температура; $\sigma$ – функция softmax; *z* – результирующие логиты модели.

Температура – параметр, который используется для регулировки степени «мягкости» или «жёсткости» функции softmax, влияя на распределение вероятностей классов. Чем выше значение температуры, тем более равномерное распределение вероятностей классов, а при низких значениях температуры вероятности классов будут более сконцентрированы на одном классе.

Функция потери схожести объяснений имеет следующий вид:

$$\Im_{\text{exp}} = 1 - sim\left(Exp\left(T, x, \hat{y}_T\right), Exp\left(S, x, \hat{y}_T\right)\right),$$

где *sim* – функция схожести; $Exp\left(H, x, \hat{y}_T\right)$ – объяснения модели *H*; $\hat{y}_T$ – класс, определенный моделью учителя.

При дистилляции знаний применительно к свёрточным нейронным сетям, в качестве объяснений модели используется класс-дискриминационная карта, полученная методом Grad-CAM по формуле (1).

## III. Результаты эксперимента на разных архитектурах дистиляции знаний

В ходе исследования проведены эксперименты с использованием различных подходов дистилляции знаний. Тестировались как архитектура дистилляции знаний, основанная на выводе модели (рис. 1), так и архитектура, основанная на внутренней структуре модели (рис. 2) при различном проценте дистилляции

знаний (4%, 15% и 100%) из модели учителя в модель ученика. В экспериментах на архитектуре рис. 2 проверялась целесообразность применения модели $e^2KD$ в связке с объяснениями, полученными методикой Grad-CAM.

Дистилляции знаний выполнялась из модели ResNet-34 в ResNet-18.

В качестве функции схожести использовалось косинусное сходство – мера сходства между двумя векторами, которая вычисляется как косинус угла между ними. Мера показывает, насколько похожи два вектора по направлению, но не учитывает их длину. Косинусное сходство вычисляется как скалярного произведения двух векторов и деления его на произведение длин двух векторов. Результат вычисления косинусного сходства находится в диапазоне от -1 до 1, где 1 означает, что вектора идентичны, а -1 означает, что вектора противоположны.

Визуально результаты применения такого подхода показаны на рис. 4. На нем в первом столбце представлено исходное изображение, во втором объяснение учителя, в третьем объяснения ученика при использовании классического подхода дистилляции знаний и на последнем объяснение ученика при добавлении $\left(e^2KD\right)$. Если при классической дистилляции знаний объяснения ученика и учителя заметно отличались, то за счет такого подхода они максимально схожи.

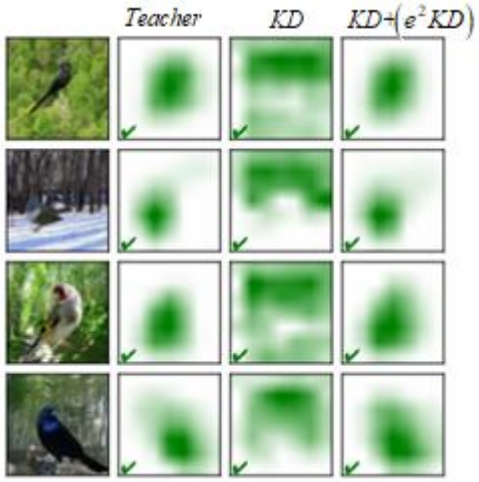


Рис. 4. Визуальные объяснения моделей [8]

Результаты, отражающие изменения в точности (Accuracy, Acc) и согласованности (Agreement, Agr) получаемой модели ученика, показаны в табл. I.

ТАБЛИЦА I. ИЗМЕНЕНИЕ В ТОЧНОСТИ И СОГЛАСОВАННОСТИ МОДЕЛИ УЧЕНИКА

| **Процент дистилляции** | **4%** | | **15%** | | **100%** | |
|---|---|---|---|---|---|---|
| Характеристика | Acc | Agr | Acc | Agr | Acc | Agr |
| ResNet-18 | 23,3 | 24,8 | 47,0 | 50,2 | 68,9 | 78,8 |
| *KD* | 49,8 | 55,5 | 63,1 | 71,9 | 71,8 | 81,2 |
| *KD+e²KD* | **54,9** | **61,7** | **64,1** | **73,2** | **71,8** | **81,6** |

Результаты эксперимента показали, что архитектуры дистилляции знаний, основанные на внутренней структуре модели являются более точными и согласованными в сравнении с архитектурой, основанной на выводе модели.

## IV. ЗАКЛЮЧЕНИЕ

Показано, что дистилляция знаний является эффективным решением проблемы оптимизации моделей нейронных сетей для использования их на устройствах с ограниченными ресурсами.

Эффективность дистилляции доказана в ходе эксперимента, проведенного на глубоких свёрточных нейронных сетях ResNet-34 в ResNet-18.

Результаты эксперимента показали важность достоверности модели ученика по отношению к модели учителя, а также создание моделей объяснений, использование которых приводит к большей точности и согласованности модели ученика.

В дальнейшем планируется изучить влияние дистилляции данных и для других моделей нейронных сетей, где используются другие архитектуры и обрабатываются другие типы данных (тексты на естественном языке, векторы данных, аудиозаписи).

## СПИСОК ЛИТЕРАТУРЫ

[1] He F., Liu T., Tao D. Why ResNet works? Residuals generalize // IEEE Transactions on Neural Networks and Learning Systems 31(12), 2020. С. 5349-5362.

[2] Wu B., Dai X., Zhang P., Wang Y., Sun F., Wu Y., Keutzer K. FBNet: Hardware-Aware Efficient ConvNet Design via Differentiable Neural Architecture Search // IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, США, 2019. С. 10726-10734. doi: 10.1109/CVPR.2019.01099.

[3] Devlin J., Chang M. W., Lee K., Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding, Minnesota, США // Proceedings of NAACL-HLT, 2019. С. 4171-4186.

[4] Hinton G.E., Vinyals O., Dean, J. Distilling the Knowledge in a Neural Network. 2015. 9 с. ArXiv, abs/1503.02531.

[5] Stanton S., Izmailov P., Kirichenko P., Alemi A.A., Wilson A.G. Does Knowledge Distillation Really Work? 2021. 21 с. ArXiv, abs/2106.05945.

[6] Selvaraju R.R., Cogswell M., Das A., Vedantam R., Parikh D., Batra D. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization // IEEE International Conference on Computer Vision (ICCV), Venice, Италия, 2017. С. 618-626. doi: 10.1109/ICCV.2017.74.

[7] Bau D., Zhou B., Khosla A., Oliva A., Torralba A. Network Dissection: Quantifying Interpretability of Deep Visual Representations // IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, США, 2017. С. 3319-3327. doi: 10.1109/CVPR.2017.354.

[8] Parchami-Araghi A., Bohle M., Rao S., Schiele B. Good Teachers Explain: Explanation-Enhanced Knowledge Distillation. 2024. 21 p.

[9] Bohle M, Fritz M., Schiele B. B-cos Networks: Alignment Is All We Need for Interpretability // IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022. С. 10319-10328. doi:10.1109/CVPR52688.2022.01008.

# Применение нейронной сети YOLOV8 для детекции средств индивидуальной защиты на опасных предприятиях

М. В. Алексюк, Д. А. Терешкин, М. Ю. Ушанкова, А. А. Шабуров

*Государственный университет «Дубна», г. Дубна Московской обл.*

shaburov8242@gmail.com

***Аннотация.*** **Исследование включает анализ применения искусственных нейронных сетей семейства архитектур *YOLO*, для мониторинга техники безопасности на опасных предприятиях. Разработан программный модуль, основанный на *YOLOv*8, для отслеживания в реальном времени обнаружения средств индивидуальной защиты на производстве. Проведено сравнительное исследование моделей *YOLO* на изображениях, полученных с камер видеонаблюдения на предприятии.**

**Выбор модели YOLOv8 обусловлен ее высокой точностью для задач детекции и сегментации изображений. (*IoU* ~90% на 25-30 *fps*). Обученная модель распознает пять классов нарушений техники безопасности. Обучение проведено на примерно 1000 изображений, обеспечивающем эффективное обучение и обобщение модели. Программный модуль имеет расширенный функционал авторизации и регистрации, а также функции просмотра видео и анализа нарушений с возможностью выбора типа нарушения.**

**Разработано комплексное решение для мониторинга соблюдения правил техники безопасности на промышленных объектах. Программное обеспечение включает в себя настольное приложение и чат бот для Telegram. Стек используемых технологий: Windows Presentation Foundation (WPF), языки C# и Python, .Net Framework, OpenCvSharp.**

***Ключевые слова: исследование; искусственные нейронные сети; YOLO; мониторинг; безопасность; программный модуль; YOLOv8; детекция; средства защиты; сравнительное исследование; обученная модель; датасет; расширенный функционал; авторизация; регистрация; просмотр видео; анализ нарушений; WPF; C#; интеграция; Telegram; Python-бот; гибкость; удобство; комплексное решение; высокая точность; коммуникационные платформы***

## I. Введение

В современном мире безопасность на опасных предприятиях является критически важной. Проблема текущего видеомониторинга: ограниченная или отсутствующая возможность отслеживания в реальном времени и нечеткое распознавание работников в форме.

Использование YOLOv8 для быстрой обработки видеопотока в реальном времени, точной детекции объектов (включая работников и их форму), минимизации ложных срабатываний и адаптивности к различным условиям освещения. Это обеспечивает эффективный мониторинг, повышает безопасность и оперативность реагирования на угрозы.

### *A. Цель*

Оценить применимость модели *YOLOv*8 для мониторинга безопасности на опасных предприятиях.

### *B. Задачи*

- анализ существующих решений;
- поиск и подготовка (разметка) данных;
- выбор архитектуры и настройка *YOLOv*8;
- обучение модели;
- оценка производительности;
- обработка данных в реальном времени;
- модуль уведомлений;
- тестирование на сценариях;
- реализация и интеграция приложения.

## II. Теоретическая справка

В настоящее время распознавание объектов с помощью машинного обучения в компьютерном зрении – ключевое направление. Для этого используются различные подходы и алгоритмы, например, обучение с учителем.

*YOLO* (*You Only Look Once*) – семейство нейронных сетей, предназначенное для обнаружения объектов в изображениях и видео. *YOLOv*8 – последняя версия, разработанная *Ultralytics*, с улучшенной точностью и функциональностью. Эта модель использует одношаговую детекцию, полностью сверточные слои, многомасштабную детекцию, что позволяет ей достигать компромисса между скоростью и точностью. Она разработана в 2012 году Джозефом Редмондом и Сантьяго Скарлетт. Оригинальная архитектура отличается способностью проводить детекцию в реальном времени [1].

Особенности *YOLO*:

- Одношаговая детекция: обеспечивает быструю детекцию в реальном времени.
- Сеть полностью сверточная: эффективна при обработке изображений различных размеров.
- Многомасштабная детекция: разбивает изображение на сетку и предсказывает боксы с объектами в каждой ячейке.
- Эффективность: компромисс между скоростью и точностью, подходит для реального времени.

*YOLOv*8:

- *Backbone* (основа): использует сверточные слои для извлечения признаков из изображений.
- *Head* (голова): формирует финальные предсказания, включая координаты ограничивающего прямоугольника, вероятность объекта и класс объекта.
- *Loss Function* (функция потерь): определяет соответствие предсказаний действительным данным и корректирует веса.

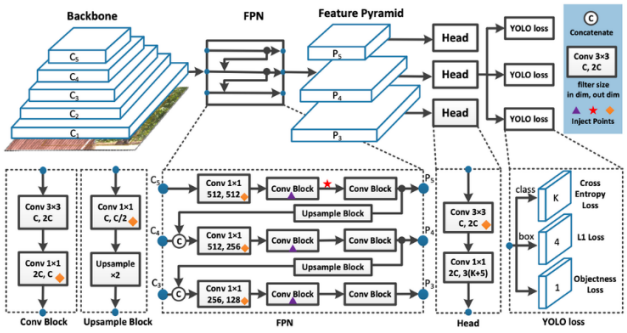


Рис. 1. Архитектура *YOLO[5]*

В работе [3, *с*. 7] представлены экспериментальные данные, демонстрирующие производительность системы средней точности 97,7 %, *F*1-меры 92,7 % и скорости кадров 63 кадра в секунду, превосходящей другие передовые методы обнаружения шлемов на мотоциклах.

## III. Методология

Датасет [6] для обучения модели был собран самостоятельно из изображений, находящихся в открытых источниках. Поскольку не существовало подходящего размеченного датасета, необходимо было разметить его вручную. Обучение модели проводилось путем использования предобученной модели *YOLOv*8 с замороженными слоями, за исключением последних слоев, которые были дообучены на подготовленном датасете. В результате обучения было получено около 5 параметров. Обучение заняло примерно 45 минут на мошностях *Google Colabaratories* с *GPU Tesla T*4, 15102*MiB*.

Модель прошла 117 эпох обучения, направленные на оптимизацию ее производительности.

Достигнутые показатели:

- *F*1-мера (1): 0,704, что свидетельствует о высокой точности модели в классификации объектов.

$$F1=2*\frac{Precision*Recall}{Precision+Recall} \quad (1)$$

- *Precision* (2): 0,696, демонстрируя низкий уровень ложных срабатываний, что особенно важно для минимизации ошибок в процессе работы.

$$Precision=\frac{True\ positive}{True\ positive+\ False\ positive} \quad (2)$$

- *Recall* (3): 0,587, указывая на способность модели находить большую часть релевантных объектов, что является ключевым фактором для полноты и достоверности результатов.

$$Recall=\frac{True\ positive}{True\ positive+\ False\ negative} \quad (3)$$

Анализ результатов:

Полученные результаты обучения модели позиционируют ее как эффективный инструмент для решения задач, связанных с классификацией и прогнозированием. Высокие значения *F*1-меры, *precision* и *recall* подтверждают точность и надежность модели. Существует потенциал для дальнейшего совершенствования модели, особенно в области *recall*, для достижения максимально возможных показателей.

### A. *Преимущества*

- высокая точность детекции;
- удобство использования;
- интеграция с *Telegram*.

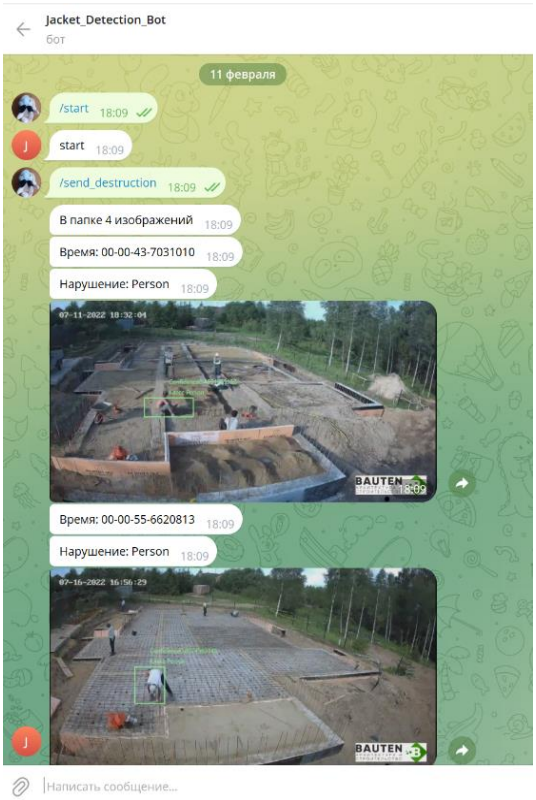


Рис. 2. Интеграция с *Telegram*

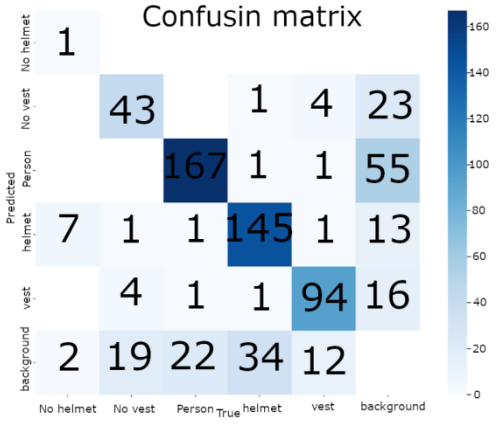


Рис. 3. Матрица ошибок

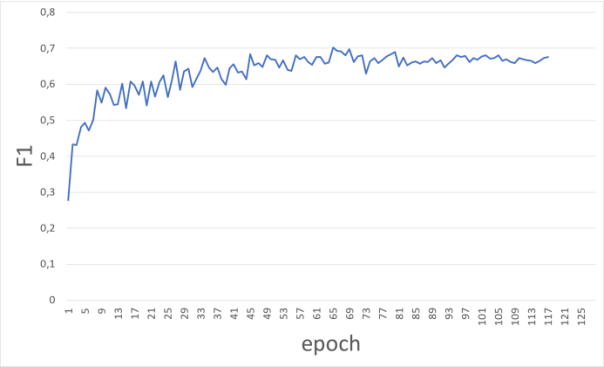


Рис. 4. Гармоническое среднее между точностью и полнотой

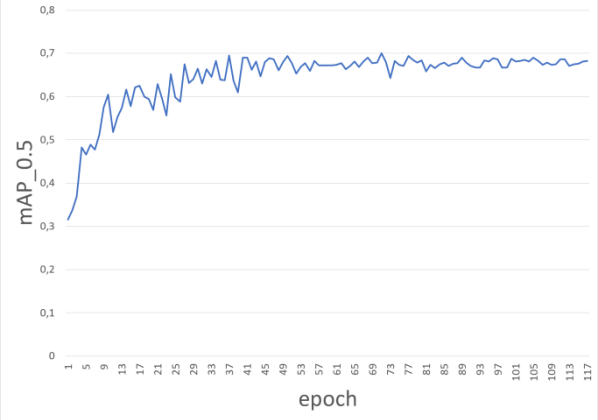


Рис. 5. Средняя точность

## IV. Результаты и обсуждение

### A. *Видеомониторинг*

- Загружает видео из файла или веб-камеры;
- Воспроизводит, приостанавливает и останавливает воспроизведение видео;
- Отображает видео в пользовательском интерфейсе.

### B. *Обнаружение объекта*

- Использует модель обнаружения объектов *YOLOv8* для идентификации объектов в видеокадрах;
- Фильтрует обнаруженные объекты на основе выбранного класса и/или области обнаружения.
- Модель очень хорошо делает прогнозы относительно того, носят ли люди каску. Из 155 примеров данных, которые на самом деле имели каску, модель машинного обучения правильно классифицировала 145 из них.
- Модель также очень хорошо делает прогнозы относительно того, носят ли люди жилет. Из 100 примеров данных, которые на самом деле имели жилет, модель машинного обучения правильно классифицировала 94 из них.
- Модель машинного обучения реже делает ошибки при прогнозировании того, является ли найденный объект человеком. Из 169 примеров данных, которые на самом деле были людьми, модель машинного обучения неправильно классифицировала только 2 из них.
- Гармоническое среднее между точностью и полнотой достигает максимума при частоте эпох 73.
- Средняя точность модели машинного обучения увеличивается со временем. На 117 эпохе точность достигла 0,68206


Рис. 6. Работа программы с ограничивающей рамкой

### C. *Ограничивающая рамка и визуализация уверенности*

- Рисует ограничивающие рамки вокруг обнаруженных объектов с регулируемой толщиной и цветом;
- Отображает класс объекта и оценку достоверности в ограничивающей рамке.


Рис. 7. Обнаружение объекта по классу с определенной точностью

### D. *Покадровая обработка*

- эффективно обрабатывает видеокадры со скоростью 30 *FPS*;
- отправляет каждый кадр [3] интерпретатору *Python* [2] для обнаружения объектов.

```python
def main():
    # Инициализация
    video_source=... # Укажите источник видео (камера, файл)
    model = ... # Загрузите модель машинного обучения
    ui = ... # Инициализируйте интерфейс пользователя
    # Цикл по кадрам
    while True:
        # Захват кадра
        frame = capture_frame(video_source)
        # Преобразование в формат OpenCV
        image = convert_to_opencv(frame)
        # Обработка кадра моделью
        results - send_to_python(image, model)
        # Визуализация
        ui.clear_display() # Очистить экран
        # Отображение результатов
        for object in results:
            draw_bounding_box(object) # Нарисовать рамку вокруг объекта
            display_class_and_confidence(object)# Отобразить класс и уверенность
        # Взаимодействие
        handle_user_input(ui) # Обработать пользовательский ввод
```

Рис. 8. Алгоритм обработки видео

### *E. Пользовательский интерфейс*

- предоставляет кнопки для управления воспроизведением видео и выбора класса объекта;
- отображает ползунок прогресса для прокрутки видео;
- представляет видео и обнаруженные объекты в четкой и информативной форме.

### *F. Вывод*

Разработанный программный модуль представляет собой комплексное решение с высокой точностью, интеграцией с коммуникационными платформами и удобством в использовании.

## V. Заключение

В ходе исследования разработан программный модуль, основанный на *YOLOv*8, обеспечивающий высокоточный мониторинг техники безопасности на опасных предприятиях. Модель *YOLOv*8, выбранная за высокую точность, распознает пять классов нарушений на реальных изображениях. Эффективное обучение на датасете из 1000 изображений обеспечивает обобщение модели. Программный модуль с расширенным функционалом авторизации, регистрации, просмотра видео и анализа нарушений обеспечивает удобство использования. Используемые технологии *WPF* и *C#* гарантируют эффективное и надежное функционирование, а интеграция с *Telegram* через *Python*-бот добавляет гибкость.

## Список литературы

[1] YOLO: Краткая история. [Электронный ресурс]. Режим доступа: https://docs.ultralytics.com/ru#yolo-a-brief-history (дата обращения14.03.2024)

[2] Python для анализа данных [Электронный ресурс]. Режим доступа: https://www.python.org/ (дата обращения: 10.01.2024).

[3] OpenCV [Электронный ресурс]. Режим доступа: https://opencv.org/ (дата обращения: 10.01.2024).

[4] YOLOv8-Based Automated Helmet Detection System for Enhanced Workplace Safety in Industrial Environments. [Электронный ресурс]. Режим доступа: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4727525 (дата обращения: 10.01.2024).

[5] What is YOLOv7? A Complete Guide [Электронный ресурс]. Режим доступа: https://blog.roboflow.com/yolov7-breakdown/ (дата обращения: 10.01.2024).

[6] Objectdetected [Электронный ресурс]. Режим доступа: https://universe.roboflow.com/gloves/objectdetected/browse?queryText=&pageSize=50&startingIndex=0&browseQuery=true (дата обращения: 10.01.2024).

# Цифровая реализация спайкового нейрона на программируемой логической интегральной схеме

И. А. Зайцев

*Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)*

ivan-za-2000@mail.ru

***Аннотация.*** **Цель работы – разработка синхронной цифровой схемы сегментной спайковой модели нейрона как элемента искусственной нейронной сети. В работе применены методы численного моделирования в средах Quartus Prime, MatLab, Simulink и ModelSim. Результатом работы является синтезируемое описание цифровой схемы искусственного нейрона.**

***Ключевые слова: спайковый нейрон; импульсная нейронная сеть; модель Ижикевича; цифровая обработка сигналов; программируемые логические интегральные схемы; программируемые вентильные матрицы***

## I. Введение

В современном мире активно развиваются нейросетевые технологии. Разработчики наращивают размеры нейросетей в угоду увеличения точности нейровычислений. Одновременно с этим растут требования к скорости вычислений, т.к. многие прикладные задачи должны решаться в реальном времени. Наиболее перспективными принято считать импульсные (спайковые) нейронные сети, как наиболее приближенные по протекающим процессам к живым нейронам [3].

Так же в современной вычислительной технике появилась тенденция внедрять в бытовые вычислители и в ЭВМ общего назначения небольшие по размеру модули программируемой логики. Данные программируемые логические матрицы могут представлять собой как отдельные микросхемы, так и блоки внутри монокристальных ЭВМ и микропроцессоров. Такие блоки, как правило, содержат исключительно базовые ячейки (пара LUT-таблица перекодировки и D-триггер), блоки статической памяти, магистральные шины и банки ввода-вывода.

Как предмет доклада предлагается адаптированная под программируемые логические матрицы цифровая схема сегментного спайкового (импульсного) нейрона [2], реализованная исключительно на базовых логических элементах без использования матричных умножителей или иных специфических блоков цифровой обработки сигналов.

## II. Теоретические сведения

Рассмотрим биологический нейрон, проиллюстрированный на рис. 1. Можно выделить несколько ключевых элементов.

- Синапс – «вводит» импульсы от других нейронов или от рецепторов в нейрон.
- Дендрит – проводит электрический импульс от синапса к телу нейрона.
- Сома – тело нейрона. От изменения мембранного потенциала сомы зависит выходной импульс нейрона.
- Аксон – выводной отросток нейрона. Проводит сгенерированный сомой нейрона импульс к другим нейронам.

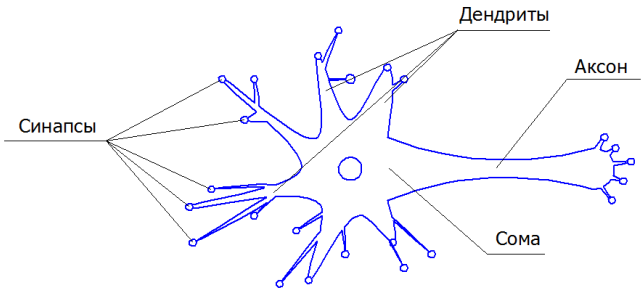


Рис. 1. Схема биологического нейрона

Биологические нейроны обмениваются импульсами.

Каждый синапс, в зависимости от концентрации ионов, наличия нейромедиаторов и прочих факторов ослабляет или наоборот усиливает импульс, приходящий от соседнего нейрона или рецептора. Нейрон, от которого получено воздействие, принято называть пресинаптическим, а нейрон, получающий воздействие – постсинаптическим. Каждый синапс имеет сразу несколько типов связи с аксоном соседнего нейрона – несколько ионных каналов. Типы связей отличаются ионами, отвечающими за перенос энергии от пресинаптического нейрона к постсинаптическому. Для обобщения и упрощения вычислений ионные каналы делят на те, что увеличивают потенциал мембраны нейрона – возбуждающий или деполяризационный, и те, что уменьшают потенциал мембраны – тормозящий или гиперполяризационный [1]. Согласно кабельной теории дендритов [1], дендриты можно представить как фильтры нижних частот. Т. е. короткие импульсы, полученные по ионным каналам от синапсов, проходят фильтрацию, прежде чем начнут влиять на потенциал мембраны.

Поведение мембраны описывается достаточно сложными законами. Наиболее точная модель – модель нейрона Ходжкина–Хаксли [2], однако она обладает значительной вычислительной сложностью. Самая простая в описании модель – модель интегратора с утечкой [3]. Согласно наиболее простой модели, мембрана копит с потерями заряд от синаптических воздействий, и во время достижения порогового значения весь накопленный потенциал сбрасывается, а на аксоне формируется выходной импульс. В предлагаемой реализации использована модель нейрона Ижикевича, как компромисс между точностью описания и вычислительной сложностью [4].

## III. Практическая реализация

На рис. 2 представлена структурная схема цифровой модели сегментного импульсного нейрона.

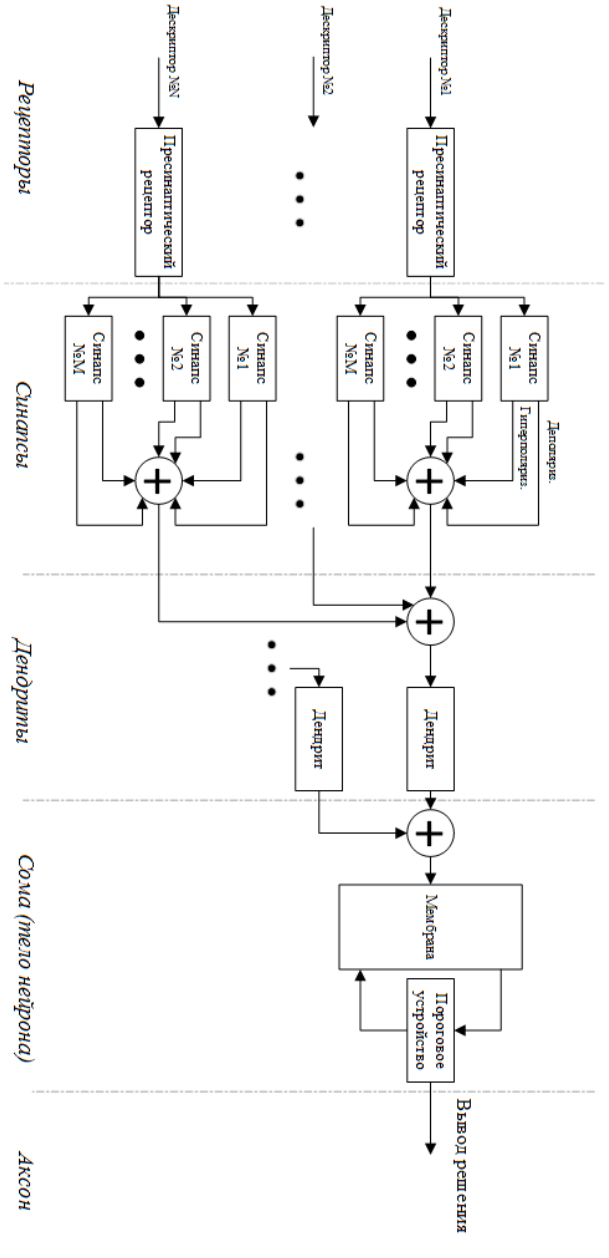


Рис. 2. Структурная схема аппаратной реализации нейрона

### A. *Рецептор*

Первым делом в нейросеть необходимо ввести значения – дескрипторы, операции над которыми будет производить нейросеть. Для ввода числовых данных часто используют метод разреженных гауссовских рецептивных полей [3]. В данной разработке метод заключается в следующем:

- Примем, что на один рецептор приходится $m$=32 синапса. Рецептор кодирует значения

  от $I_{min}$ = -128 до $I_{max}$ =127

- Рассчитываем значения:

$$\sigma = \frac{1}{1{,}5} \cdot e^{\frac{I_{max} - I_{min}}{m-2}}$$ – ширина гауссиана,

$$c_i = I_{min} + \frac{2 \cdot i - 3}{2} \cdot \frac{I_{max} - I_{min}}{m-2}$$ – центры гауссиан,

где $i$ – номер гауссиана.

- Строим 32 функции Гаусса по формуле

$$y = 100 - \frac{100}{\max(x)} \cdot \left( \frac{1}{\sigma \cdot \sqrt{2\pi}} \cdot e^{-\frac{(x-c_i)^2}{2 \cdot \sigma^2}} \right)$$, округляя до ближайшего целого,

где $y$ – ось времени (100 дискретов), а x – ось дескрипторов (256 отсчётов) см. рисунок 3.

Рецептивный кодировщик работает следующим образом: выбираем дескриптор (кодируемое значение), таким образом значение каждого гауссиана для этого значения равняется времени испускания импульса на конкретном выводе. К каждому выводу подключается свой синапс. Если величина гауссиана больше 90, то импульс считается не сработавшим.

Например (рис. 3) для кодирования значения «-121» на 2-м выводе импульс появится на 11-м временном отсчёте, на 3-м выводе – на 40м временном отсчёте, на 1-м – на 86-м временном отсчёте.

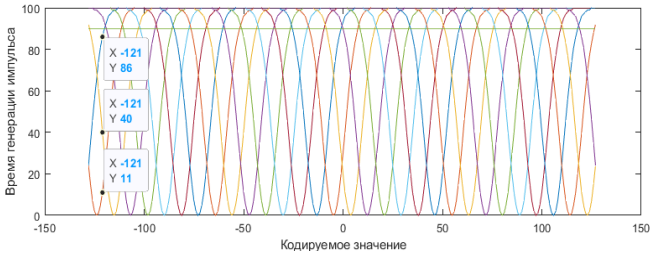


Рис. 3. Рецептивные Гауссовы поля

Идея практической реализации рецептора заключается в записи функций Гаусса в ПЗУ таким образом, чтобы при подаче адреса на ПЗУ на выходе выдавалось значение гауссиана для данного числа. Стоит учитывать, что при кодировании знаковых переменных, адрес будет подаваться как беззнаковое число. Соответственно необходимо будет развернуть функции относительно нуля. Для перехода от пространственного формата представления данных к временному

используется возрастающий счётчик и компаратор, как показано на рис. 4.

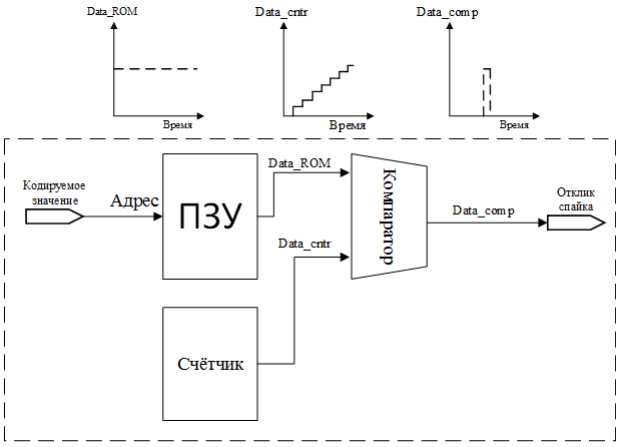


Рис. 4. Фрагмент рецептора

## *B. Синапс*

Синапс в данной схеме переводит одноразрядные импульсы в шестнадцатиразрядные импульсы с определённой амплитудой, равной «весу» синапса.

Несмотря на то, что модель нейрона использует отдельно деполяризационный и гиперполяризационный ионные каналы, в аппаратной реализации для экономии аппаратуры целесообразно использовать только один канал, вес которого равен разнице между деполяризационным и гиперполяризационным «весами».

Наличие двух каналов в программной модели обусловлено тем, что разные каналы обучаются по разным правилам.

Структурная схема приведена на рисунке 5. «Spike_in» – вход импульса от одного из выводов рецептора. «W» – вес синапса.

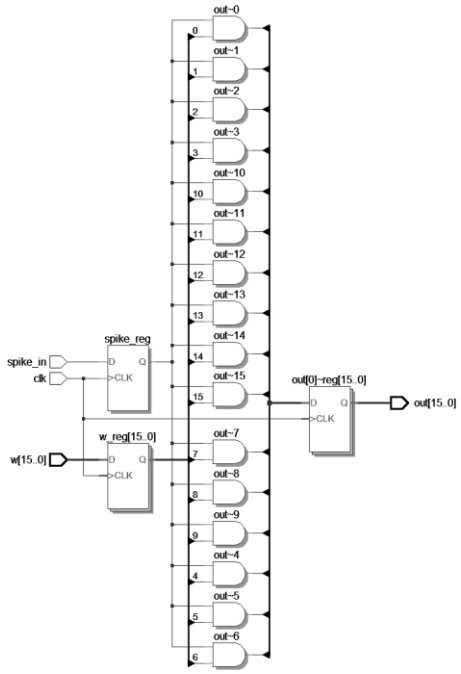


Рис. 5. Структурная схема синапса

## *C. Дендрит*

Дендрит представляет собой емкостной фильтр нижних частот. Аналоговый фильтр нижних частот можно заменить на цифровой, как показано на рисунке 6.

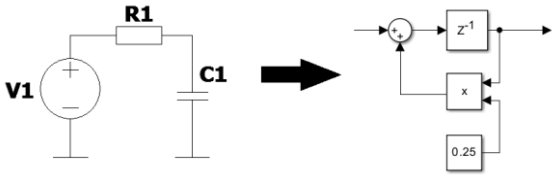


Рис. 6. Преобразование фильтра

В данном примере терминация заряда на резисторе реализована как коэффициент 1/4 положительной обратной связи. Знаковое деление на 4 в двоичном виде можно представить как сдвиг на два двоичных разряда вправо (в сторону младших разрядов) и дублирование старшего бита на место вакантных битов слева, т. е. [15..0] ← [15, 15, 15..2].

## *D. Мембрана и пороговое устройство*

В данной работе применена модель мембраны нейрона Ижикевича, как наиболее приближенная к биологическому нейрону при невысокой вычислительной сложности [3] [4]. Конкретный вид уравнений мембраны Ижикевича взят следующего вида:

$$\begin{cases} \dfrac{dV}{dt} = \dfrac{1}{32}\cdot V^2 + 5\cdot V + 140 - U + I \\ \dfrac{dU}{dt} = a\cdot\left(b\cdot V - U\right) \end{cases}, V \le V_{th}$$

$$\begin{cases} V \leftarrow c \\ U \leftarrow U + d \end{cases}, U > 30$$

где V – потенциал мембраны; U – вспомогательная переменная; I – токовое воздействие от синапсов; a = 0.625, b = 0.5, c = -40, d = 50 – константы; $V_{th}$ = 30 – порог срабатывания.

Данные уравнения численно решаются по методу Эйлера следующим образом, где h = 0.25 – время интегрирования.

$$\begin{cases} V(t) = V(t-1) + \left[\dfrac{1}{32}\cdot V^2(t-1) + 5\cdot V(t-1) + 140 - U(t-1) + I(t-1)\right]\cdot h \\ U(t) = U(t-1) + a\cdot\left(b\cdot V(t-1) - U(t-1)\right)\cdot h \end{cases}, V \le V_{th}$$

$$\begin{cases} V(t) = c \\ U(t) = U(t-1) + d \end{cases}, U > 30$$

Рассмотрим уравнения для условия V ≤ 30. За исключением возведения в квадрат и умножения на 5, все прочие коэффициенты-множители выбраны как степени двойки. Умножение и деление в двоичной системе на множители или делители, представляющие из себя степени двойки, выглядит как сдвиг влево или вправо на число разрядов, равное этой самой степени. Как пример, умножение на 4 выглядит как [15..0] ← [13..0, $0_{bin}$, $0_{bin}$]. Деление на 4 выглядит как [15..0] ← [15, 15, 15..2]. Для упрощения вычислений, представим 5·V(t-1) = 4·V(t-1) + V(t-1).

Возведение в квадрат в данном случае выполняется аппаратно алгоритмом быстрого возведения в квадрат [5]. Сущность этого алгоритма заключается в следующем: любое машинное слово можно представить как бинарный вектор. В примере это вектор X.

$$X = \begin{bmatrix} x_7 & x_6 & x_5 & x_4 & x_3 & x_2 & x_1 & x_0 \end{bmatrix} =$$
$$= x_7 \cdot 2^7 + x_6 \cdot 2^6 + x_5 \cdot 2^5 + x_4 \cdot 2^4 + x_3 \cdot 2^3 + x_2 \cdot 2^2 + x_1 \cdot 2^1 + x_0 \cdot 2^0$$

Возведение в квадрат соответственно представляется как X·X.

$$X^2 = X \cdot X =$$
$$= \left(x_7 \cdot 2^7 + x_6 \cdot 2^6 + x_5 \cdot 2^5 + x_4 \cdot 2^4 + x_3 \cdot 2^3 + x_2 \cdot 2^2 + x_1 \cdot 2^1 + x_0 \cdot 2^0\right) \cdot$$
$$\cdot \left(x_7 \cdot 2^7 + x_6 \cdot 2^6 + x_5 \cdot 2^5 + x_4 \cdot 2^4 + x_3 \cdot 2^3 + x_2 \cdot 2^2 + x_1 \cdot 2^1 + x_0 \cdot 2^0\right) =$$
$$= \left[\left(x_7^2 \cdot 2^{14}\right) + \left(x_7 \cdot x_6 \cdot 2^{13}\right) + \left(x_7 \cdot x_5 \cdot 2^{12}\right) + \left(x_7 \cdot x_4 \cdot 2^{11}\right) + \right.$$
$$+ \left(x_7 \cdot x_3 \cdot 2^{10}\right) + \left(x_7 \cdot x_2 \cdot 2^9\right) + \left(x_7 \cdot x_1 \cdot 2^8\right) + \left(x_7 \cdot x_0 \cdot 2^7\right)\Big] +$$
$$+ \left[\left(x_6 \cdot x_7 \cdot 2^{13}\right) + \left(x_6^2 \cdot 2^{12}\right) + \left(x_6 \cdot x_5 \cdot 2^{11}\right) + \left(x_6 \cdot x_4 \cdot 2^{10}\right) + \right.$$
$$+ \left(x_6 \cdot x_3 \cdot 2^9\right) + \left(x_6 \cdot x_2 \cdot 2^8\right) + \left(x_6 \cdot x_1 \cdot 2^7\right) + \left(x_6 \cdot x_0 \cdot 2^6\right)\Big] +$$
$$+ \bullet\bullet\bullet +$$
$$+ \left[\left(x_0 \cdot x_7 \cdot 2^7\right) + \left(x_0 \cdot x_6 \cdot 2^6\right) + \left(x_0 \cdot x_5 \cdot 2^5\right) + \left(x_0 \cdot x_4 \cdot 2^4\right) + \right.$$
$$+ \left(x_0 \cdot x_3 \cdot 2^3\right) + \left(x_0 \cdot x_2 \cdot 2^2\right) + \left(x_0 \cdot x_1 \cdot 2^1\right) + \left(x_0^2 \cdot 2^0\right)\Big]$$

Побитное умножение не даёт расширения разряда и может быть заменено логической операцией конъюнкции, остается только сложить 8 слагаемых, внутри которых происходит побитное «И».

Важным замечанием является то, что данная операция происходит над беззнаковыми числами. Для операции над знакопеременными числами необходимо сначала взять модуль числа, и только потом выполнить возведение в квадрат.

Модуль числа можно представить как операцию «исключающего ИЛИ» старшего разряда над каждым более младшим разрядом и последующее сложение полученного машинного слова непосредственно с этим самым старшим разрядом. Пример:

$$Y = \begin{bmatrix} y_{15} & y_{14} & \dots & y_0 \end{bmatrix} = \left|X\right|$$
$$X = \begin{bmatrix} x_{15} & x_{14} & \dots & x_0 \end{bmatrix}$$
$$\begin{bmatrix} y_{15} & y_{14} & \dots & y_0 \end{bmatrix} = \begin{bmatrix} x_{15} \oplus x_{15} & x_{15} \oplus x_{14} & \dots & x_{15} \oplus x_0 \end{bmatrix} +$$
$$+ \begin{bmatrix} 0_{bin} & 0_{bin} & \dots & x_{15} \end{bmatrix}$$

Общая схема мембраны представлена на рис. 7.

ПУ – пороговое устройство. $Z^{-1}$ – элемент памяти на D-триггерах, работающей на тактовой частоте «CLK_X1» = 50 МГц. Остальная часть схемы синхронна и полностью выполняется за 4 такта на частоте «CLK_X4» = 250 МГц.

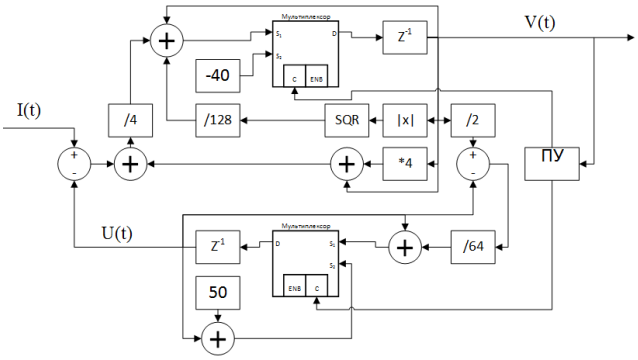


Рис. 7. Структурная схема мембраны

## IV. Заключение

Данный аппаратный комплекс портирован на ПЛИС Altera Cyclone IV GX. Рассматриваются варианты портирования на малые ПЛИС китайских фирм, а так же на монокристальные ЭВМ (SoC) с блоками программируемой синхронной логики. Затраченные ресурсы приведены в табл. 1. Под ячейками подразумевается пара таблица перекодировки-регистр. В примере использовался только один рецептор.

ТАБЛИЦА I.

| **Элемент** | **Число затраченных ячеек** | **Число затраченных ячеек всего** |
|---|---|---|
| Рецептор | 214 | 1318 |
| Синапс (32 штуки) | 513 | |
| Дендрит | 51 | |
| Мембрана | 268 | |
| Пороговое уст-во | 16 | |
| Межсегментные сумматоры | 512 | |

Для обучения применялось правило STDP (Spike-time-dependable Plasticity) [4]. Для примера бралась простейшая задача различения положительных чисел от отрицательных. После обучения на полной группе, нейрон различал положительные числа от отрицательных с вероятностью 1. Успешное выполнение задачи классификации чисел наглядно показывает работоспособность данной аппаратной модели нейрона.

## Список литературы

[1] Позин Н.В. Моделирование нейронных структур. М.: Гл. ред. физ.-мат. лит-ры изд-ва «Наука», 1970, 264 стр.

[2] Бахшиев А.В., Демчева А.А. Сегментная спайковая модель нейрона CSNM // Известия высших учебных заведений. Прикладная нелинейная динамика. 2022. Т. 30, № 3.

[3] Евграфов В.А., Ильюшин Е.А. Спайковые нейронные сети // IInternational Journal of Open Information Technologies ISSN: 2307-8162 vol. 9, no.7, 2021, 21. 32 с.

[4] Humaidic A.J., Kadhim T.M., Hasan S., Ibraheem I.K., Azar A.T. A Generic Izhikevich-Modelled FPGA-Realized Architecture: A Case Study of Printed English Letter Recognition // 24th International Conference on System Theory, Control and Computing (ICSTCC) – 2020, 825–830 p.

[5] Панкратова И.А. Теоретико-числовые методы криптографии: учебное пособие. Томск: Томский государственный университет, 2009. 120 с.

# Разработка структуры нейронной сети для управления системами кондиционирования воздуха

М. Дык Нгуен, М. П. Белов, А. М. Белов

*Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)*

haduna.hv@gmail.com, milesa58@mail.ru, sana199706@mail.ru

***Аннотация.*** **Система кондиционирования воздуха является стохастической, нестационарной, многосвязной, нелинейной промышленной системой. Использование традиционных методов управления для такого объекта не обеспечивает требуемого качества регулирования, так как поддержание параметров микроклимата (температуры, влажности в определенном помещении) с высокой точностью. Для управления многосвязными выходными параметрами системы кондиционирования предлагается искусственная нейронная сеть (ИНС). Тип нейронной сети — многослойный персептрон (MLP). Алгоритм управления нейронной сетью основан на коррекции весовых коэффициентов с использованием алгоритма градиентного спуска и обратного распространения ошибки.**

***Ключевые слова: искусственная нейронная сеть (ИНС); многослойный персептрон (MLP); градиентный спуск; Отопление, вентиляция и кондиционирование воздуха (ОВиК).***

## I. Введение

Первоначально наиболее важной проблемой для предприятий, использующих системы ОВиК (отопление, вентиляция и кондиционирование воздуха), было поддержание зональных условий на заданных значениях, связанных с комфортом в отношении тепла и влажности для пассажиров. Однако, когда энергетический кризис начался, проблема системного энергопотребления стала актуальной. Чтобы контролировать оба этих фактора, многие проектировщики начали моделировать и проектировать новые типы компонентов систем ОВиК.

В этой статье несколько исследований были сосредоточены на точном построении модели ОВиК. Например, Кларк [1] представил предварительную модель системы HVAC для некоторых компонентов, таких как увлажнитель и смесительный блок, но не представил конкретную модель охлаждающих/нагревательных змеевиков. Втанг и др. [2] использовали принципы энергетического баланса и теплопередачи для определения линейной модели, представляющей нелинейную модель охлаждающего змеевика. Многие другие исследователи изучали динамические модели ОВиК, используя экспериментальные или теоретические методы. Нассиф и др. [3] представили валидацию охлаждающего змеевика, смесительной камеры и вентилятора для системы VAV (система вентиляции с переменным расходом воздуха).

Однако видно, что моделирование этих отдельных процессов не может точно разрешить внутренние ограничения процессов теплообмена и энергообмена, происходящих в системах ОВиК. Это затрудняет процесс прогнозирования и нахождения взаимосвязи между внешними возмущениями. В то же время процесс построения и калибровки точной модели по сравнению с реальностью очень сложен и требует много времени.

Современный подход к управлению процессами, происходящими в системах ОВиК, основан на искусственных нейронных сетях. Этот метод основан на достаточно большой базе реальных данных и использует современные вычислительные технологии (машинное обучение, глубокое обучение – это быстрорастущие области исследований в области искусственного интеллекта), чтобы найти и смоделировать практически точные нелинейные взаимосвязи между элементами системы. Таким образом, исследователям не требуется слишком глубокое понимание системы, чтобы иметь возможность построить модель управления системой.

## II. Структура системы автоматического управления

Общий принцип организации системы автоматического управления кондиционером (рис. 1) включает 3 уровня: уровень управления, уровень управляющих устройств, уровень полевых устройств. В частности, устройств на верхнем уровне меньше, чем устройств на нижнем уровне. Устройства, расположенные на верхнем этаже, имеют более четкие функции управления.

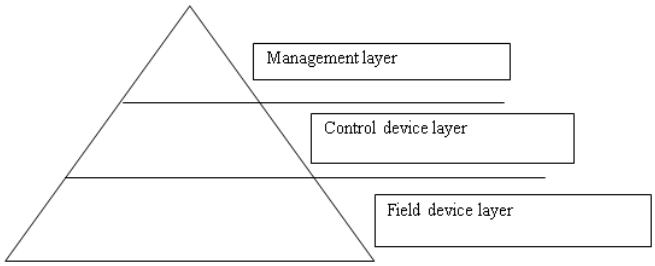


Рис. 1. Общий организационный принцип работы САУ

### *A. Уровень управления*

На рис. 2 показаны устройства на уровне управления. С точки зрения физического подключения серверы, рабочие станции и BNA равны. Однако функции рабочей станции и сервера различны. Сервер может иметь полную функциональность рабочей станции, но рабочая станция не может иметь функции сервера.

- Сервер: система управления кондиционированием воздуха состоит из 2 серверов, работающих одновременно и резервирующих друг друга. Серверы являются одновременно серверами приложений и серверами данных. Для выполнения этой функции на серверах устанавливается программное обеспечение, помогающее управлять, хранить данные и при необходимости распределять их по рабочим станциям.

- Рабочие станции: это прикладные рабочие станции, на которых установлено программное обеспечение для мониторинга и управления системой посредством обработки данных с сервера. Рабочие станции в местах расположения имеют эквивалентные роли и могут взаимозаменяться в процессе работы системы кондиционирования воздуха.

- BNA выполняет функцию преобразования сигналов из сети C-bus в Ethernet для серверов, собирающих данные с контроллеров. BNA является связующим звеном между уровнем управляющих устройств и уровнем управления.

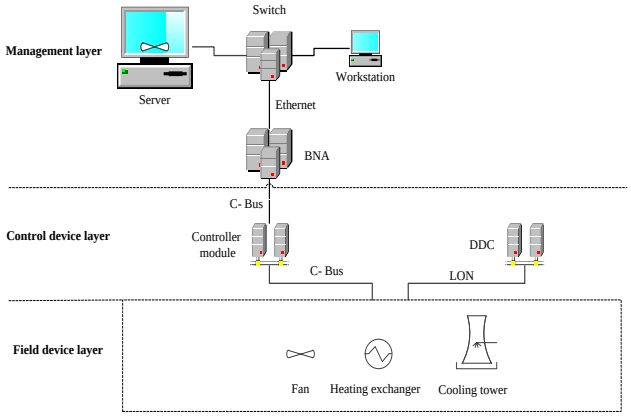


Рис. 2. Устройства слоев

### *B. Уровень управляющих устройств*

Включает в себя прямые цифровые контроллеры (DDC) в шкафах управления, сенсорные экраны в шкафах управления, модули ввода/вывода, стандартные блоки измерения холодопроизводительности Lonworks и устройства измерения электрической мощности.

Контроллеры взаимодействуют с сенсорным экраном и общаются друг с другом по сети C-bus.

Внутри шкафа управления контроллеры, модули ввода/вывода и блоки измерения холодопроизводительности, блоки измерения электрической мощности взаимодействуют данными друг с другом по протоколу Lonworks или сети Panel-bus.

### *C. Уровень полевых устройств*

Включает датчики температуры и влажности, датчики расхода воздуха, датчики давления воды, переключатели перепада давления, переключатели сигнализации ветра, сигнализаторы перегрева... и исполнительные механизмы, такие как воздушные клапаны, VAV, трехходовой клапан, двухпозиционный водяной клапан, водяной насос, инвертор, электрическая сушилка, электростатический фильтр, увлажнитель воздуха...

Полевые устройства связаны с модулями ввода/вывода через 4 типа сигналов, разделенных на соответствующие типы точек передачи данных: цифровой вход, аналоговый вход, цифровой выход, аналоговый выход.

## III. Разработка системы управления ОВиК на основе нейтронной сети

### *A. Принцип управления системой кондиционирования воздуха*

Общие принципы работы систем автоматического управления:

1. Степень открытия трехходового клапана чиллеров и мощность увлажнителя регулируются влажностью в помещении. Датчик влажности измеряет влажность в помещении. Контроллер сравнивает это значение с заданным значением, при наличии отклонения между этими двумя значениями ( $\Delta\varphi \neq 0$) контроллер выдает 2 сигнала управления (0 ÷ 10)В соответственно (0 ÷ 100) % на 2 выхода для управления трехходовой клапан чиллера или мощность увлажнителя. Когда влажность в помещении выше заданной, мощность увлажнителя снизится или трехходовой клапан охладителя откроется больше. И наоборот, если влажность в помещении ниже, мощность увлажнителя увеличится или клапан чиллера уменьшится. Диапазон регулирования трехходового клапана составляет от 10 % до 100 %.

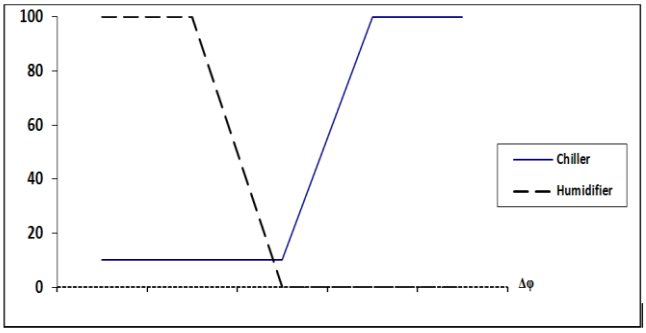


Рис. 3. Зависимость между сигналами управления в зависимости от влажности

2. Мощность теплообменника и степень открытия VAV контролируются температурой помещения. Датчик температуры измеряет температуру воздуха в помещении. Контроллер сравнивает это значение с заданным значением, при наличии отклонения между этими двумя значениями ( $\Delta t \neq 0$) контроллер выдает 2 сигнала управления (0 ÷ 10)В соответственно (0 ÷ 100) % на 2 выхода для управления мощности теплообменника или степени открытия VAV. Когда температура в помещении выше заданной температуры, мощность теплообменника снизится или степень открытия VAV откроется больше. И наоборот, если температура в помещении ниже, мощность теплообменника увеличится или степень открытия VAV уменьшится. Сигнал управления мощностью теплообменника находится в диапазоне от 5 % до 100 %. Степень открытия VAV колеблется от 10 % до 60 %.

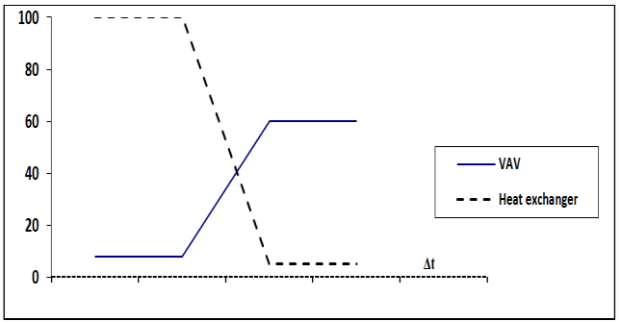


Рис. 4. Зависимость между сигналами управления в зависимости от температуры

3. Мощность вентилятора контролируется статическим давлением за вентилятором. Датчик измерения давления измеряет статическое давление на трубе за вентилятором. Контроллер сравнивает это значение с заданным значением (ΔP = статическое давление за вентилятором – заданное значение), если появляется отклонение между этими двумя значениями, контроллер генерирует на выходе управляющий сигнал (0÷10)В, эквивалентный (0÷100) % для управления инвертором вентилятора. Если измеренное статическое давление превышает заданное значение, мощность инвертора вентилятора уменьшится; если измеренное статическое давление ниже заданного значения, мощность инвертора увеличится. Сигнал управления вентилятором находится в диапазоне от 20 % до 100 % (после периода запуска сигнал управления не может упасть ниже 20 % мощности). Мощность инвертора вентилятора также изменяется инвертором во время работы VAV, чтобы обеспечить воздухообмен.

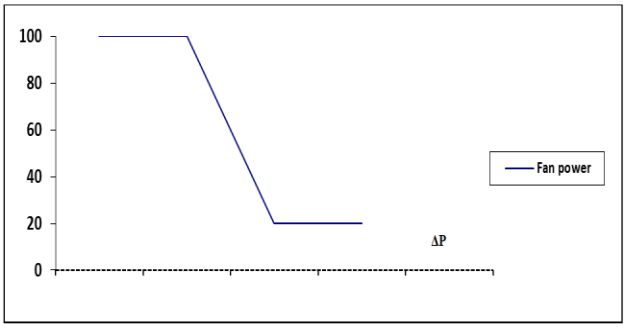


Рис. 5. Зависимость сигналов управления вентилятором от давления

### *B. Применение нейронных сетей для управления системами кондиционирования воздуха*

Система кондиционирования представляет собой многосвязный, стохастический, нестационарный, нелинейный объект. Следует отметить, что существуют требования по высокоточному поддержанию параметров микроклимата, а именно температурного и влажностного режимов.

Искусственные нейронные сети (ИНС) начинают играть заметную роль в создании систем автоматического управления (САУ) сложными динамическими объектами. К таким объектам относятся современные самолеты, энергетические комплексы, мобильные роботы и другие. Они характеризуются отсутствием точных математических моделей или их чрезмерной сложностью, высокой размерностью пространства состояний и управляющих решений, иерархичностью, разнообразием критериев качества, высоким уровнем шума и внешних возмущений.

В настоящее время проблеме управления параметрами окружающей среды в системах ОВиК посвящено множество работ. Представленное решение основано на градиентном методе управления киберфизическими системами. Преимуществом разработанного метода является его способность к самонастройке в условиях неконтролируемых возмущений и отсутствие необходимости обучения нейронной сети на основе данных о динамике объекта.

Обобщенную структурную схему АСУ можно представить в виде, показанном на рис. 6.

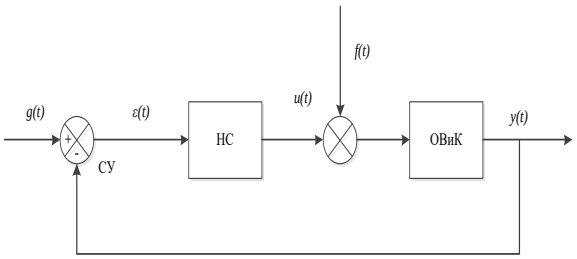


Рис. 6. Обобщенная структурная схема САУ

где ОВиК – объект управления; НС – регулятор; g(t) – опорное влияние; ε(t) – ошибка управления; u(t) – управляющее воздействие; f(t) – возмущающее воздействие; y(t) – управляемая переменная.

Существуют различные способы применения НС в системах управления. Такие применение зависят от положения подключения НС в сетях. Нейронная сеть может использоваться для получения нелинейной или инверсной математической модели объекта управления. Другой вариант использования НС – ее применение в качестве блока подстройки параметров основного (линейного) регулятора. В этой задаче, НС играют роль регулятора в замкнутый контур управления объектом. В этом случае на вход нейронной сети подается сигнал ошибки управления, выход из нейронной сети u(t) также является входом объекта; Цель обучения НС — уменьшить величину ошибки управления ε(t) = y(t) – g(t) между выходами объекта и заданным значением.

Для построения блока нейронной сети используется нейронная сеть прямого распространения, такая как многослойный перцептрон (Multi-Layer Perceptron – MLP).

Алгоритм обучения искусственной нейронной сети основан на коррекции весовых коэффициентов с использованием метода градиентного управления и метода обратного распространения ошибки. Он состоит из следующих этапов:

1. Запуск нейросетевого алгоритма управления. Каждый нейрон вычисляет свою функцию активации на основе входных данных и передает результат следующему нейрону.

2. Вычислите ошибку. После прямого распространения сигнала вычисляется разница между

полученными и ожидаемыми выходными значениями сети. Эта разница является ошибкой, которую необходимо свести к минимуму.

3. Расчет локальных градиентов (метод обратного распространения). Сетевая ошибка распространяется в противоположном направлении, от выходных нейронов к входным нейронам. Каждый нейрон вычисляет градиент ошибки относительно своих входов и весов соединений.

4. Используя градиент ошибки, веса связей между нейронами корректируются с использованием заданного обновления веса. Цель состоит в том, чтобы минимизировать ошибки и улучшить результаты работы сети.

Алгоритм обучения сети можно записать как среднеквадратическую ошибку по обучающим наборам N:

$$E(w) = \frac{1}{N}\sum_{i=1}^{N}(g_i(k) - y_i(k))^2 = \frac{1}{N}\sum_{i=1}^{N}\varepsilon_i^2(k), \quad (1)$$

где $y_i(k)$ – реальный выходной сигнал нейрона выходного i -слоя; $g_i(k)$ – желаемый выходной сигнал этого нейрона; $\varepsilon_i(k)$ – вектор ошибки на k-итерации.

Структура системы управления представлена в виде динамической системы (рис. 7) и включает нейросетевой блок и систему кондиционирования воздуха.

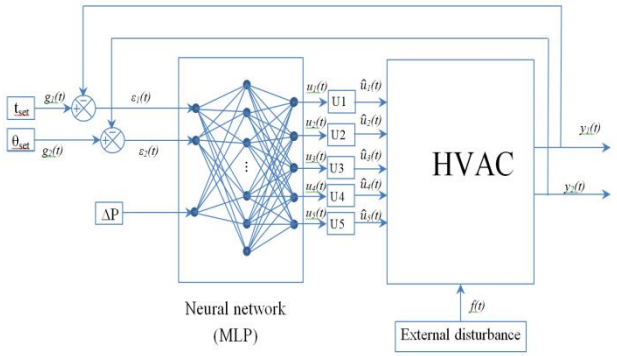


Рис. 7. Структура динамической системы нейронной сети

Входными параметрами управления (рис. 7), обеспечивающими высокоточное поддержание заданных значений температуры $g_1(t)$ и влажности $g_2(t)$, могут быть мощность оросительной камеры (увлажнителя) $u_1(t)$, степень открытия трехходового клапана холодильных машин (чиллеров) $u_2(t)$, мощность теплообменника (рекуператоров) u3(t), степень открытия систем с переменным объемом воздуха (VAV) u4(t) ) и мощность вентилятора u5(t).

Динамическая система нейронной сети описывается дискретной моделью:

$$y(k+1) = F(y(k), W(k)), \quad (2)$$

где y(k)=($y_1$(k),$y_2$(k)) – вектор выходного состояния динамической системы на k-шаге процесса; W(k) – набор управляющих координат на k-шаге процесса, k=0, 1, 2,... (весовые коэффициенты нейронной сети). Дискретная модель задается набором отображений системы F(k).

На основе значений множества W(k) и входных ε(k), блок нейронной сети генерирует вектор выходных параметров u(k)=($u_1$(k),..., $u_5$(k)), где каждый элемент вектора u(k) связан с объектом управления через масштабирующий блок (U).

Цель управления задается в виде неотрицательной функции E(y(k+1)) ≤ Δ при k>k*, где Δ > 0 – заданное значение порога точности задачи; k* – номер шага, на котором достигается цель управления.

Неизвестно оптимальное значение w*(t) весовых коэффициентов, соответствующее условию:

$$w^*(t) = \arg\min E(t), w \in W, \quad (3)$$

$$E(t) = \frac{1}{2}\sum_{i=1}^{M}\varepsilon_i^2(t), \quad (4)$$

$$\varepsilon_i(t) = g_i(t) - y_i(t), \quad (5)$$

Установка весовых коэффициентов w(t) осуществляется градиентным методом по уравнению:

$$w(k+1) = w(k) - \eta\frac{\partial E(k)}{\partial w(k)} = w(k) - \eta\nabla E(w), \quad (6)$$

где η – коэффициент шага градиентного метода.

Система управления с нейрорегулятором на основе (6) со временем начинает терять свою устойчивость из-за накопления ошибок округления. Задача (4) относится к классу некорректных задач. Для решения этой проблемы используется метод регуляризации. Для преобразования некорректной задачи в корректную критерий (4) был переписан в виде:

$$\tilde{E}(w(t)) = E(w(t)) + \Omega(w(t)), \quad (7)$$

где Ω(w(t)) – равномерно выпуклая функция, которая определяется как:

$$\Omega(w(t)) = \frac{1}{2}\lambda[(\sum_{n=1}^{3}\sum_{s=1}^{S}w_{ns}^1(k))^2 + (\sum_{m=1}^{5}\sum_{s=1}^{S}w_{sm}^2(k))^2 + (\sum_{n=1}^{3}\sum_{s=1}^{S}b_{ns}^1(k))^2 + (\sum_{m=1}^{5}\sum_{s=1}^{S}b_{sm}^2(k))^2], \quad (8)$$

где $\lambda$ – коэффициент регуляризации; $w_{ns}^1(k), w_{sm}^2(k), b_{ns}^1(k), b_{sm}^2(k)$ – нейронная сеть весовых коэффициентов с одним внутренним слоем (настройка параметров).

Использование параметра λ ограничит рост весов нейронной сети, что обеспечит устойчивость системы управления. Параметр λ определяется экспериментально для каждого конкретного случая [5].

Таким образом, решается робастная задача нейронного управления, которая включает в себя нахождение архитектуры сети MLP (количество слоев сети, количество нейронов в скрытых слоях, функцию активации в каждом слое), системное количество шагов градиентного метода η, определяет скорость установки параметра W, коэффициента регуляризации λ.

## IV. Заключение

Были представлены общие сведения о структуре системы управления кондиционером и контролируемых параметрах.

Предполагается, что применение нейронной сети в системе управления ОВиК возможно при наличии случайных возмущений. Это может принести много преимуществ по сравнению с традиционными методами управления.

Предлагается план определения параметров нейронной управляющей сети. Используется метод регуляризации для повышения стабильности процесса нейронного управления.

## Список литературы

[1] Кларк Дж.А., 2011, «Энергетическое моделирование при проектировании зданий». 2-е издание, Баттерворт Хайнеманн.

[2] Ван Ю., Цай В., Ли С., Се Л., Со Ю., 2012 г. «Разработка модели охлаждающего змеевика для управления и оптимизации системы», IEEE CCA, Китай.

[3] Нассиф Н., Кайл С., Сабурен Р., 2010. «Моделирование композиций» существующая система CVCA». Ви Кол. Межуниверситетский Франко-Квебекский университет, Канада.

[4] Муравьева Е.А., Шарипов М.И. Системы искусственного интеллекта: Учебное пособие. К.: Уфа, 2018. 168 с.

[5] Фролов С.В., Синдеев С.В., Коробов А.А., Потлов А.Ю. Комбинированный метод нейроуправления нелинейным нестационарным объектом // 2-я Международная конференция по системам управления, математическому моделированию, автоматизации и энергоэффективности (SUMMA), 2020, с. 582–585, doi: 10.1109/SUMMA50634.2020.9280705

# Классификация движений руки на основе данных электромиограмм с использованием частотно-временных признаков

И. В. Кузнецов

*Дальневосточный государственный университет путей сообщения*

kyzaiv1@yandex.ru

***Аннотация*. При классификации движений на основе данных электромиографии (пЭМГ) важную роль играет выбор признаков классов движений. Ставится предположение о повышении эффективности классификации движений при совместном использовании признаков временного (Time Domain, TD) и частотно-временного (Time-Frequency Domain, TFD) пространств. Для проверки предположения были построены классификаторы на основе метода опорных векторов (Support Vector Machines, SVM), метода K-ближайших соседей (K-Nearest Neighbors, KNN), Случайного леса (Random Forest, RF). В качестве набора данных использован набор NinaPro DB5. Была достигнута эффективность классификации 91.3 % и 92.8 % для метрик Recall и Precision соответственно, при использовании KNN. Предлагаемые методы могут использоваться для создания человеко-машинных интерфейсов, использующих данные мышечной активности.**

***Ключевые слова: электромиография; метод опорных векторов; Random Forest; KNN; классификация движений; частотно-временная область; временная область***

## I. Введение

Одной из актуальных проблем, стоящих при решении задачи классификации движений на основе электромиографии (ЭМГ), является выбор признаков движений, которые можно извлечь из ЭМГ-сигнала для построения классификаторов. Исследователями построено большое количество классификаторов, использующих для своей работы различные комбинации признаков. Признаки, которые могут быть извлечены из ЭМГ-сигнала для классификации движений, относятся к временной (Time Domain, TD), частотной (Frequency Domain, FD) или частотно-временной областям (Time-Frequency Domain, TFD).

Широкое распространение получило применение признаков временной области, что обусловлено простотой их извлечения. Так, авторы [1] отмечают, что признаки временной области используются для классификации движений чаще всего, что обусловлено простотой их получения, в то время как признаки частотной области, как правило, требуют больше вычислительных ресурсов. В [2] авторы приводят результаты разработки экзоскелета для людей, перенесших инсульт, для которой разработали систему адаптивного контроля на основе методов машинного обучения, использующих признаки TD. Авторы [3] добились точности классификации 98 % для движений, описанных с помощью двух признаков ЭМГ-сигнала и двух признаков, получаемых на основе данных инерционных датчиков (IMU) и смогли применить классификатор для управления промышленным роботом с шестью степенями свободы. В [4] при сравнении эффективности методов машинного обучения автору удалось достичь точности 98.34 % при использовании признаков временной области для пяти движений. Авторы [5] предложили архитектуру глубокой нейронной сети, достигшей точности 98 % для трех движений с использованием трех признаков временной области. В [6] приводятся результаты распознавания десяти движений на основе искусственной нейронной сети, метода опорных векторов, метода случайного леса и логической регрессии. В работе было использовано шесть TD-признаков, и была получена точность 94 % для искусственной нейронной сети. Авторы отмечают, что используя только функции TD, удалось достичь более высокого отношения жестов к каналам, чем другие подобные исследования, и предполагают, что предлагаемый ими метод может улучшить удобство использования системы и снизить вычислительную нагрузку.

Однако применение признаков TD имеет и ограничения. Phinyomark в [7] указывает, что главным недостатком признаков TD является нестационарность ЭМГ-сигнала, изменяющая статистические свойства с течением времени, в то время функции TD принимают данные как стационарный сигнал. Существует большое число факторов внешней среды, влияющих на характеристики ЭМГ-сигнала. Помимо этого, важную роль играет и состояние субъекта, что может привести к ситуации, когда одно и то же движение приводит к разным формам электромиограммы. Это в свою очередь приводит к увеличению числа ошибок классификации. В частности, авторы [8], рассматривая возможности использования ЭМГ-сигнала для контроля электроники автомобиля, указывают, что в силу нестационарности ЭМГ-сигнала, для его использования в качестве управляющего сигнала требуется собирать как временную, так и частотную информацию.

Одним из перспективных направлений повышения точности классификации является применение признаков TFD. Авторы [9, 10] указывают, что вычисление признаков TFD требует проведения требовательных к вычислительным ресурсам вычислений. Часто для извлечения признаков TFD применяются дискретные вейвлет-преобразования (ДВТ)

и оконное преобразование Фурье (ОПФ). В [11] было проведено сравнение эффективности ОПФ и непрерывного вейвлет-преобразования (НВП). Автор указывает, что значения, полученные в результате использования НВП были более репрезентативны, нежели после применения ОПФ для оценки мышечной усталости при сверхмаксимальной динамической нагрузке с постоянной нагрузкой. Авторы [1] при сравнении точности классификации движений на основе спектрограмм указывают, что вейвлет-преобразования вычислительно более сложны, нежели преобразования Фурье, однако последние не предоставляют информации из временной области. В работе [12] авторы добились точности классификации 90 % для SVM и 100 % для KNN и классификатора Boosted Trees при решении задачи классификации движений пальца. Авторы указывают, что в ряде случаев использование признаков TFD позволяет преодолеть ограничения временной области. В [13] автор приводит результаты сравнения эффективности KNN при использовании признаков TFD и FD, приводятся результаты в виде 95.5 % для TFD и 89 % для FD.

Несмотря на большое число исследований, посвященных классификации движений на основе TD и TFD, а также их совместном использовании, остается открытым вопрос эффективности применения различных комбинаций признаков. Целью данной работы является сравнение эффективности классификации движений при использовании и признаков TD и при совместном использовании признаков TD и TFD.

## II. Материалы и методы исследования

Для проверки гипотезы был проведен эксперимент по оценке точности работы классификаторов, построенных на основе следующих методов:

- SVM
- RF
- KNN

Было построено по два варианта классификатора: с использованием TD-признаков и с использованием признаков как TD, так и TFD-признаков.

Для проведения расчетов использована библиотека scikit-learn и язык программирования Python.

### *A. Используемый набор данных*

В качестве используемого набора данных был выбран набор NinaPro DB5. Это набор данных ЭМГ-сигнала, полученный от 10 субъектов с помощью двух браслетов Thalmic Labs Myo. ЭМГ считывалась с помощью шестнадцати поверхностных многоразовых электродов. В рамках эксперимента для обучения классификаторов не использовалась иная информация (такая как данные кинематики движений), представленная в наборе.

Набор данных содержит информацию о пятидесяти двух движениях, из них для классификации выбрано 12. Данные движения показаны на рис. 1. Помимо представленных движений, в наборе данных содержится информация о периодах, когда не выполнялось никакого движения.

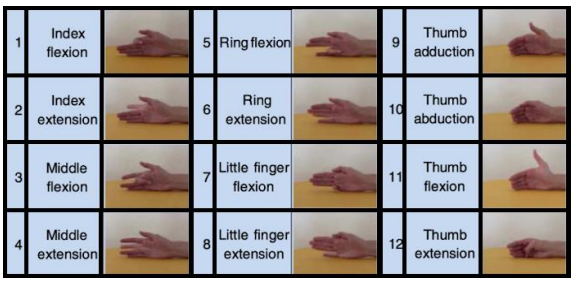


Рис. 1. Движения, выбранные для эксперимента

Движение в наборе описано в виде набора векторов, каждый из которых представляет собой запись определенного канала записи ЭМГ-сигнала.

Следует отметить, что количество объектов различных классов не одинаково: количество объектов, соответствующих периодам состояния покоя, значительно больше, нежели объектов, соответствующих классам движений. Это требуется учитывать при оценке эффективности работы классификаторов, использующих данный набор.

При проведении эксперимента принято, что набор данных не подвергнут влиянию шумов и других факторов, требующих его дополнительной обработки.

### *B. Используемые признаки временной области*

В рамках эксперимента были извлечены следующие TD-признаки: Root Men Square (RMS), Mean Absolute Value (MAV), Standard Deviation (SD).

MAV – это среднее значение абсолютных величин EMG-сигнала. Spiewak в [9] описывает MAV как метод обнаружения и измерения сокращения мышц. В [10] авторы, рассмотрев возможности применения SVM для анализа ЭМГ-сигнала, пришли к выводу, что MAV относится к одному из наиболее часто используемых признаков. MAV вычисляется следующим образом:

$$MAV = \frac{1}{N}\sum_{i=1}^{N}|x_i|$$

С помощью RMS описывается сила мышцы при неустающем сокращении [14]. Авторы [15] отмечают, что признаки RMS и iEMG (integrated, сумма значений ЭМГ-сигнала) не только отражают изменения амплитудных значений сигнала, но и характеризуют биомеханические свойства и изменения мышечной энергии в процессе выполнения движения. RMS вычисляется по согласно следующему выражению:

$$RMS = \sqrt{\frac{1}{N}\sum_{i=1}^{N}x_i^2}$$

Стандартное отклонение – еще один из широко используемых признаков. Согласно [16], он может использоваться для определения порогового уровня активности сокращения мышц. SD вычисляется следующим образом ($\bar{x}$ – среднее значение ЭМГ-сигнала):

$$SD = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i-\bar{x})^2}$$

Каждый признак рассчитывался для каждого канала считывания ЭМГ-сигнала. Совокупность полученных значений использовалась в качестве набора данных для обучения классификаторов.

### *C. Используемые признаки частотно- временной области*

Для извлечения признаков частотно-временной области записи ЭМГ-сигналов были подвергнуты дискретному вейвлет-преобразованию. В качестве материнского был использован вейвлет Добеши пятого порядка (DB5). Для каждой группы полученных коэффициентов было рассчитано значение RMS.

Полученные значения использованы в качестве признаков различных классов движений совместно с признаками временной области для второй группы классификаторов. В рамках эксперимента не применялись методы снижения размерности пространства признаков.

## III. Результаты эксперимента

Для оценки эффективности работы были рассчитаны метрики эффективности Accuracy, Recall и Precision:

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

где $TP$ – доля объектов, корректно отнесенных к заданному классу, $TN$ – доля объектов, корректно не отнесенных к классу, $FN$ – доля объектов, некорректно не отнесенных к заданному классу, $FP$ – доля объектов, некорректно отнесенных к заданному классу.

Показатели эффективности работы классификаторов представлены в табл. 1. При расчете метрик использовано взвешенное среднее по классам. Наилучшие показатели были получены при использовании классификатора на основе метода KNN.

ТАБЛИЦА I. Метрики эффективности работы классификаторов

| | **SVM** | **KNN** | **RF** |
|---|---|---|---|
| **Accuracy (TD)** | 0.791 | 0.843 | 0.739 |
| **Precision (TD)** | 0.884 | 0.893 | 0.857 |
| **Recall (TD)** | 0.878 | 0.878 | 0.861 |
| **Accuracy (TD + TFD)** | 0.793 | 0.868 | 0.754 |
| **Precision (TD + TFD)** | 0.904 | 0.928 | 0.880 |
| **Recall (TD + TFD)** | 0.885 | 0.913 | 0.868 |

Матрица ошибок классификатора на основе KNN, использующего только TD-признаки, представлена на рис. 2. Класс, соответствующий состоянию покоя обозначен нулем, классы движений – их номерами на рис. 1.

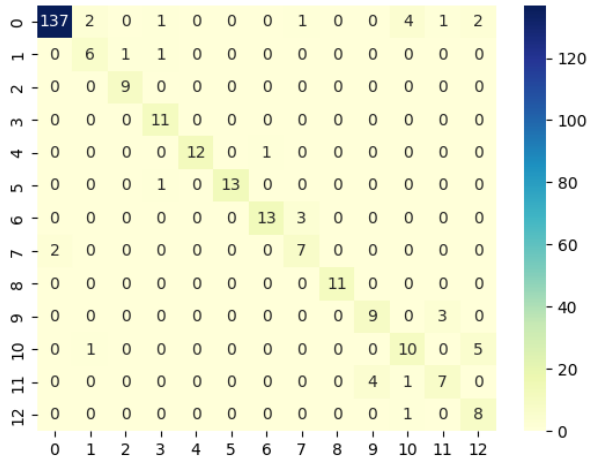


Рис. 2. Матрица ошибок KNN при использовании признаков временной области

Матрица ошибок позволяет сделать вывод о том, классификаторы часто ошибаются при классификации движений большого пальца, безымянного пальца и мизинца. Ставится предположение о том, что для правильной классификации движений большого пальца требуется располагать электроды специальным образом для повышения точности считывания ЭМГ-сигнала.

При использовании признаков частотно-временной и временной областей точность классификаторов растет. Лучший результат также показывает KNN и составляет 91.3 % для метрики Recall и 92.8 % для Precision. Матрица ошибок KNN при использовании признаков частотно-временной области представлена на рис. 3.

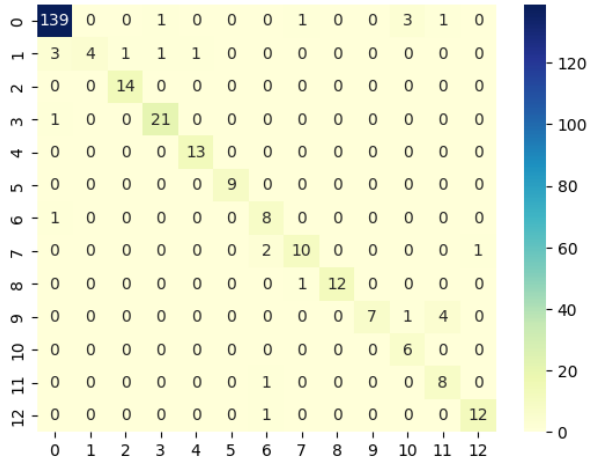


Рис. 3. Матрица ошибок KNN при использовании признаков временной и частотно-временной областей

Совместное использование признаков TD и TFD позволило увеличить долю успешной классификации движений большого пальца. Однако эффективность классификации движений мизинца и безымянного пальца изменяется незначительно, что ставит вопрос о поиске более качественных методов обработки и классификации движений.

## IV. Заключение

В работе рассмотрено применение признаков временной и частотно-временной областей для классификации движений руки на основе ЭМГ-сигнала. Рассмотрена эффективность работы классификаторов на основе метода опорных векторов, K ближайших соседей и Random Forest.

Наилучшие результаты классификации показал метод K-ближайших соседей с метриками классификации 91.3 % (Recall) и 92.8 % (Precision). Высокая точность классификации позволяет сделать вывод о возможности применения методов машинного обучения при построении человеко-машинных интерфейсов, где управляющим сигналом служит биоэлектрическая активность мышц.

Однако данная задача требует дополнительных исследований по поиску методов, способных обеспечить высокую точность классификации в режиме реального времени с минимальными затратами вычислительных ресурсов. Ожидается, что рассмотренные классификаторы могут оказаться достаточно эффективными при работе в режиме реального времени.

Перспективным направлением поиска методов классификации кажется использование методов глубокого обучения, в частности LSTM и сверточных нейронных сетей. Ожидается, что использование методов глубокого обучения позитивно скажется на эффективности классификации.

Кроме того, точность работы рассмотренных классификаторов может быть повышена за счет дальнейшего поиска эффективных комбинаций признаков. Одним из направлений дальнейших исследований является применение признаков FD для классификации движений, а также комбинаций TD, TFD и FD для достижения наибольшей точности классификации.

Еще одним актуальным направлением исследований является поиск эффективных методов снижения размерности пространства признаков. Дополнительное извлечение признаков TFD к признакам TD кратно увеличивает итоговое число признаков, однако нет гарантий того, что все они необходимы для проведения качественной классификации. Полученные в работе результаты достаточно высоки для создания человеко-машинных интерфейсов, использующих данные мышечной активности, однако дальнейшее уменьшение количества ошибок является актуальной задачей. Ожидается, что снижение размерности пространства признаков приведет к уменьшению количества потребляемых классификаторами вычислительных ресурсов, а также повысит точность классификации за счет избавления от малоинформативных признаков. Часто для этого применяются метод главных компонент и линейный дискриминантный анализ, однако вопрос их эффективности, в том числе относительно друг друга и других методов требует отдельного рассмотрения.

## Список литературы

[1] Zawawi T.N.S.T., et al. Electromyography signal analysis using spectrogram // 2013 IEEE Student Conference on Research and Developement. IEEE, 2013. P. 319-324.

[2] Bonilla D. et al. Progressive Rehabilitation Based on EMG Gesture Classification and an MPC-Driven Exoskeleton // Bioengineering. MDPI AG, 2023. Vol. 10, № 7. P. 770.

[3] Ali H., Yaneh W. SVM Classification for Novel Time Domain IMU and EMG fused features for control of 6-DOF industrial robot // 2020 IEEE International Conference on Mechatronics and Automation (ICMA). IEEE, 2020. P. 18-22.

[4] Ari A. EMG Signal Classification Using Deep Learning and Time Domain Descriptors-Based Feature Extraction for Hand Grip Movement Recognition // Traitement du Signal. International Information and Engineering Technology Association, 2023. Vol. 40, № 3. P. 949–960.

[5] Gaso M.S., Cankurt S., Subasi A. Electromyography Signal Classification Using Deep Learning // 2021 16th International Conference on Electronics Computer and Computation (ICECCO). IEEE, 2021. P. 1-6.

[6] Lee K.H., Min J.Y., Byun S. Electromyogram-Based Classification of Hand and Finger Gestures Using Artificial Neural Networks // Sensors. MDPI AG, 2021. Vol. 22, № 1. P. 225.

[7] Phinyomark A., Phukpattaranont P., Limsakul C. Feature reduction and selection for EMG signal classification // Expert Systems with Applications. Elsevier BV, 2012. Vol. 39, № 8. P. 7420–7431.

[8] Shair E. et al. EMG Pattern Recognition Using TFD for Future Control of In-Car Electronic Equipment // INTERNATIONAL JOURNAL of FUZZY LOGIC and INTELLIGENT SYSTEMS. Korean Institute of Intelligent Systems, 2022. Vol. 22, № 1. P. 11–22.

[9] Spiewak C. A. et al. Comprehensive Study on EMG Feature Extraction and Classifiers // Open Access Journal of Biomedical Engineering and Biosciences. Lupine Publishers LLC, 2018. Vol. 1, № 1.

[10] Toledo-Pérez D.C. et al. Support Vector Machine-Based EMG Signal Classification Techniques: A Review // Applied Sciences. MDPI AG, 2019. Vol. 9, № 20. P. 4402.

[11] Camata T.V. et al. Fourier and wavelet spectral analysis of EMG signals in supramaximal constant load dynamic exercise // 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology. IEEE, 2010. P. 1364-1367

[12] Shair E.F., Jamaluddin N.A., Abdullah A.R. Finger Movement Discrimination of EMG Signals Towards Improved Prosthetic Control using TFD // International Journal of Advanced Computer Science and Applications. The Science and Information Organization, 2020. Vol. 11, № 9. P. 244-251.

[13] Narayan Y. SEMG signal classification using KNN classifier with FD and TFD features // Materials Today: Proceedings. Elsevier BV, 2021. Vol. 37. P. 3219–3225.

[14] Too J. et al. EMG Feature Selection and Classification Using a Pbest-Guide Binary Particle Swarm Optimization // Computation. MDPI AG, 2019. Vol. 7, № 1. P. 12.

[15] Sun J. et al. Application of Surface Electromyography in Exercise Fatigue: A Review // Frontiers in Systems Neuroscience. Frontiers Media SA, 2022. Vol. 16.

[16] Saikia A. et al. Combination of EMG Features and Stability Index for Finger Movements Recognition // Procedia Computer Science. Elsevier BV, 2018. Vol. 133. P. 92–98.

# Определение степени заполненности мусорных баков при помощи компьютерного зрения

Я. О. Сениченкова, М. Д. Поляк

*Санкт-Петербургский государственный университет аэрокосмического приборостроения*

markpolyak@gmail.com

***Аннотация.*** **В работе рассматривается решение проблемы несвоевременного вывоза мусора с помощью машинного зрения. Предлагается осуществлять фото- и видеофиксацию состояния контейнерных площадок, на которых расположены мусорные баки, после чего, с помощью нейросетевых моделей, определять количество скопившегося мусора. Для решения данной задачи была подготовлена обширная выборка из изображений мусорных баков разных типов и форм. На основе архитектуры ResNet обучена нейронная сеть для классификации мусорных баков. С помощью дообученной модели Yolo осуществляется детекция отдельных баков на изображении.**

***Ключевые слова: ResNet, Yolo, классификация, обработка изображений, экология***

## I. Введение

В последнее десятилетие технологии компьютерного зрения вышли на новый виток развития. Все началось с представления нейронной сети AlexNet [1] на конкурсе ImageNet в 2012 году. Это событие является знаменательным, т. к. в предыдущие года в данном конкурсе побеждали классические алгоритмы машинного обучения, а не нейронные сети. Так, решение, представленное командой NEC-UIUC в 2010 году, является примером подхода к классификации изображений без использования нейронных сетей [2]. Активное использование сверточных нейронных сетей [3] началось после 2015 года, т. к. в этом году нейронная сеть ResNet [4] смогла превзойти человека по качеству классификации изображений на конкурсе ImageNet [5]. В данной работе предлагается использовать технологии компьютерного зрения и глубокого обучения для распознавания уровня заполненности мусорных баков. В качестве решения данной задачи авторы работы [6] использовали сверточную нейронную сеть, которая смогла достичь точности 93 %. Также для решения данной задачи применяются датчики, определяющие уровень заполненности мусорного бака [7]. Однако данное решение является дорогостоящим при внедрении в крупных населенных пунктах. Это связано с тем, что датчики необходимо поместить во все мусорные баки, также есть риски, связанные с вандализмом (оборудование могут воровать) и возможным повреждением датчиков острыми и твердыми отходами.

Автоматизация распознавания наполненности мусорного бака позволит службам, ответственным за вывоз мусора, оперативно получать информацию о том, какие контейнеры переполнены. Информация о наполненности мусорных баков позволит адаптировать график вывоза мусора к ситуативным изменениям, как единоразовым, так и сезонным. Коммунальные службы смогут оптимизировать свои расходы и повысить качество предоставляемых населению услуг.

Для решения поставленной задачи используются YOLOv8 для детекции бака и ResNet18 для классификации наполненности. На вход модели поступает изображение размером 224x224. Далее, часть изображения, попавшая в bounding box, поступает в классификатор, реализованный нейронной сетью ResNet18. Данный классификатор вычисляет вероятности принадлежности изображения к классам «полный бак», «полуполный бак», «пустой бак», «неопределенный бак». Примеры изображений данных классов приведены на рис. 1–3.

Рис. 1. Пример изображения, относящегося к классу «Пустой бак»

Рис. 2. Пример изображения, относящегося к классу «Полный бак»

Рис. 3. Пример изображения, относящегося к классу «Полупустой бак»

Последний класс необходим на случай, если по изображению невозможно определить наполненность контейнера из-за различных факторов (например: угла съемки, освещения). Пример подобных фотографий приведен на рис. 4.

Рис. 4. Примеры фотографий контейнеров, относящихся к классу «Неопределенный бак»

## II. Подготовка данных

Для обучения и тестирования классификатора были собраны фотографии и видео, содержащие мусорные баки. Видео были покадрово нарезаны на изображения. Всего в датасете 8065 фотографий. 933 изображения относятся к классу «Пустой бак», 2863 изображения – к классу «Полный бак», 4165 изображений – к классу «Полупустой бак», 104 изображения – к классу «Неопределенный бак». Выборка была разделена на обучающую, валидационную и тестовую часть. В тестовой выборке содержится 906 изображений. В обучающей выборке – 7258 фотографий, в валидационной – 807.

## III. Обучение классификатора

Для дообучения нейронной сети ResNet18 использовалась техника Fine tuning, при которой последний слой классификатора заменяется на слой, в котором количество нейронов совпадает с количеством классов изображений. В нашем случае датасет содержит 4 класса, следовательно, последний слой классификатора ResNet18 заменяется слоем, содержащим 4 нейрона. В качестве функции потерь была выбрана Cross Entropy Loss [8]. После дообучения, состоящего из 5 эпох, получились следующие результаты на тестовой выборке: Точность – 0.9837, F1-score – 0.9834, ROC-AUC score – 0,9997. На рис. 5 приведен график изменения значения функции потерь на обучающей выборке, на рис. 6 – график изменения точности.

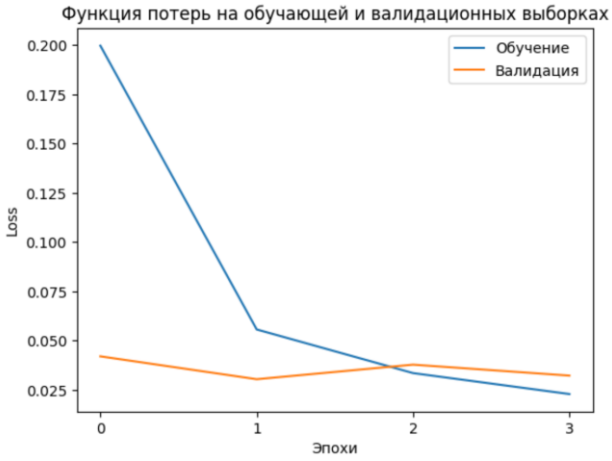


Рис. 5. График изменения значения функции потерь на обучающей и валидационных выборках в процессе обучения классификатора

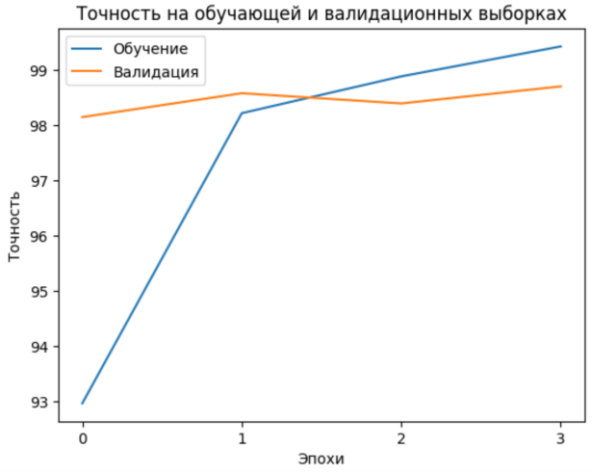


Рис. 6. График изменения значения точности на обучающей и валидационных выборках в процессе обучения классификатора

ТАБЛИЦА I. Матрица ошибок классификатора на тестовой выборке

| **Истинный класс** | **Предсказанный класс** | | | |
|---|---|---|---|---|
| | **Пустой бак** | **Полный бак** | **Полупустой бак** | **Неизвестно** |
| Пустой бак | 102 | 1 | 1 | 0 |
| Полный бак | 0 | 316 | 1 | 1 |
| Полупустой бак | 5 | 4 | 453 | 1 |
| Неизвестно | 0 | 1 | 0 | 11 |

## IV. Обучение детектора

В качестве детектора была взята нейронная сеть YOLOv8 [9]. По прошествии 10 эпох были получены следующие графики метрик, приведенные на рис. 7.

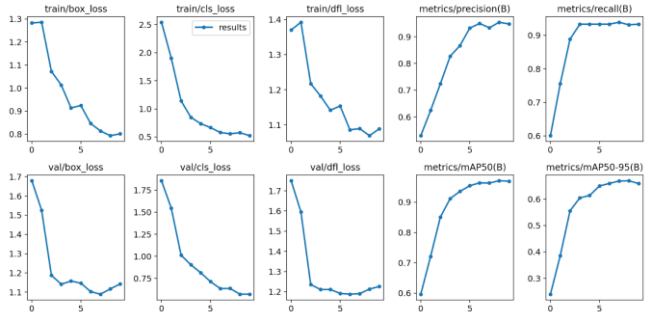


Рис. 7. Графики метрик, полученных в результате обучения YOLOv8

Были получены следующие метрики на тестовой выборке: box loss: 1.142, cls loss: 0.570, dfl loss: 1.23, mAP50: 0.969, mAP50-95: 0.658

На рис. 8 приведены пример работы полученного детектора на изображениях из тестовой выборки. Сверху ограничительной рамки указана вероятность, с которой модель отнесла объект к мусорному баку.


Рис. 8. Примеры изображений, полученных в результате работы обученной

На рис. 9 приведена нормализованная матрица ошибок для каждого класса из тестовой выборки. На диагонали можно увидеть процент верно классифицированных объектов.

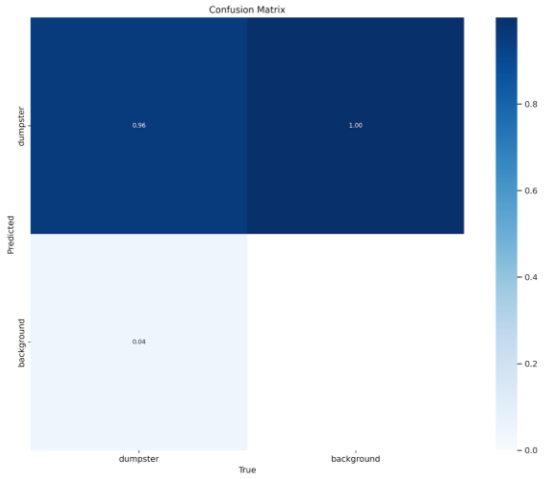


Рис. 9. Нормализованная матрица ошибок для тестовой выборки

## V. Результаты

Визуальный анализ неверно классифицированных изображений позволил выделить следующую закономерность: если внутри мусорного бака есть пятна, то нейронная сеть посчитает, что бак является полупустым. Возможно, это вызвано тем, что пятна ошибочно принимаются за объекты внутри контейнера. Пример данных изображений приведен на рис. 10.

Рис. 10. Примеры изображений, ошибочно отнесенных нейронной сетью к классу «Полупустой бак»

## VI. Заключение

В ходе исследования был построен пайплайн, который позволяет определить уровень заполненности мусорного бака. Детектор мусорных баков работает с полнотой 0.969 при значении порога IoU равным 0.5, классификатор определяет уровень наполнения контейнера для мусора с точностью 0.983. В ходе данного исследования было показано, что задача определения наполненности мусорного бака может быть достоверно решена с помощью нейросетевых алгоритмов машинного зрения. Автоматизация данной проблемы позволит коммунальным службам и экологическим сервисам более оперативно вывозить мусор из переполненных контейнеров.

## Благодарность

Авторы выражают благодарность компании ООО «АСТЕРА» за предоставление вычислительных мощностей, и ведущего инженера Бояркина Михаила Игоревича за ценные советы.

## Список литературы

[1] Krizhevsky A., Sutskever I., Hinton G.E. ImageNet classification with deep convolutional neural networks //Communications of the ACM. 2017. Т. 60. №. 6. С. 84-90.

[2] Zhou X. et al. Image classification using super-vector coding of local image descriptors // Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part V 11. Springer Berlin Heidelberg, 2010. С. 141-154.

[3] LeCun Y., Bengio Y. (1995). Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks, 3361(10), 1995.conference on computer vision and pattern recognition. 2016. С. 770-778.

[4] He K. et al. Deep residual learning for image recognition // Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. С. 770-778.

[5] ImageNet 2015. Дата обращения: https://image-net.org/challenges/LSVRC/2015/ (accessed 01 April 2024)

[6] Ramirez I., Cuesta-Infante A., Pantrigo J.J., Montemayor A.S., Moreno J.L., Alonso V., Anguita G., Palombarani L. (2020). Convolutional neural networks for computer vision-based detection and recognition of dumpsters // Neural Computing and Applications, 32(17), 13203-13211.

[7] Draz U., Ali T., Khan J. A., Majid M., Yasin S. (2017, November). A real-time smart dumpsters monitoring and garbage collection system. // In 2017 Fifth International Conference on Aerospace Science & Engineering (ICASE) (pp. 1-8). IEEE.

[8] Cox D.R. The regression analysis of binary sequences // Journal of the Royal Statistical Society Series B: Statistical Methodology. 1958. Т. 20. №. 2. С. 215-232.

[9] Han X., Chang J., Wang, K. (2021). You only look once: unified, real-time object detection // Procedia Computer Science, 183(1).

# Архитектура динамической децентрализованной большой языковой модели на основе блокчейна

Н. В. Козгунов, М. Халаши
*Санкт-Петербургский государственный университет*
st090866@student.spbu.ru

В. Д. Олисеенко
*Санкт-Петербургский Федеральный исследовательский центр РАН*
vdo@dscs.pro

Т. В. Тулупьева
*Санкт-Петербургский Федеральный исследовательский центр РАН*
*Северо-Западный институт управления РАНХиГС*
tvt@dscs.pro

***Аннотация.*** **В работе представлена новая архитектура большой языковой модели на принципе блокчейн. В основе архитектуры лежит концепция федеративного обучения через сеть узлов, призванного улучшить процесс обучения и использования LLM с применением вычислительной мощности и данными для обучения, которые предоставляют сами пользователи блокчейна. Представленная архитектура направлена на устранение проблем централизации, цензурирования и предвзятости в существующих LLM, делая доступнее их использование пользователям.**

***Ключевые слова: дифференциальная конфиденциальность; обработка естественного языка; децентрализованное федеративное обучение; блокчейн; Web 3.0***

## I. Введение

Начало 2010-х годов ознаменовалось значительным прогрессом в области обработки естественного языка, начиная с появления Word2Vec [1], ELMo [2] и Transformer [3], что привело к созданию больших языковых моделей (LLM), таких как семейства BERT [4] и GPT [5]. Развитие области обработки естественного языка помогло повысить понимание синтаксиса и семантики моделями машинного обучения [4–5].

Одновременно с этим появление концепции блокчейна, начавшегося с Bitcoin [6] и Ethereum [7], заложило основу для создания нового класса распределенных вычислений и концепции Web 3.0.

Совместное использование блокчейна и LLM может объединить преимущества технологий децентрализации блокчейна, повысив конфиденциальность, создав условия для оптимизации использования ресурсов, повысив защиту от сбоев централизованного сервера, а также снизив затраты, связанные с дообучением и использованием LLM [6–7, 14, 17, 19–20]. Стоит отметить, что такие теоретические решения находятся на начальном этапе развития, что подтверждается в некоторых публикациях [14, 19, 21].

Таким образом, цель данной работы заключается в разработке и описании архитектуры большой языковой модели на основе технологии блокчейн.

Исследование выполнено в рамках проекта по государственному заданию СПб ФИЦ РАН № FFZF-2022-0003

## II. Обзор литературы

Системы федеративного машинного обучения за счёт использования различных устройств для хранения данных и обучения моделей имеют существенные преимущества над централизованными системами машинного обучения благодаря обеспечению конфиденциальности данных и оптимизации процессов масштабируемости [8]. Однако они подвержены определенным недостаткам, включая коммуникационные издержки и риски единой точки сбоя из-за зависимости от локальных серверов [9]. В качестве альтернативы для решения указанных недостатков может выступать система децентрализованного федеративного обучения, которая использует механизмы гомоморфного шифрования для возможности обеспечения безопасного и эффективного обучения [10]. Также система предлагает структуру, основанную на ориентированных ациклических графах, для обеспечения надежности децентрализованных операций, стимулируя более широкое участие и обеспечивая справедливые выплаты участникам в соответствии с вкладом их вычислительных мощностей в обучение и вывод модели [11].

Потенциал интеграции федеративного обучения и децентрализованных методов с LLM значителен [12–16]. Исследование FusionAI демонстрирует, что потребительские графические процессоры могут эффективно использоваться для обучения и LLM, предлагая экономически выгодную альтернативу относительно существующих языковых моделей за счет сокращения расходов на мощности графических процессоров для дообучение и хранения существующей модели, расширяя возможности вывода (*англ. inference*) LLM на мобильные устройства [12, 14], при этом обеспечивая высокопроизводительный генеративный вывод в условиях ограниченных ресурсов [15]. Исследования показывают, что улучшение функциональности LLM на индивидуальных узлах, особенно в распределенных конфигурациях, способствует повышению общей эффективности системы [14–16].

Интеграция технологий консенсусов Proof-of-Work (*PoW*), Proof-of-Work (*PoS*), Proof-of-Engagement (*PoE*) [18] и Proof-of-Reputation (*PoR*) [19] в LLM представляет

собой перспективное направление развития области языковых моделей. Современный подход к децентрализации алгоритмов машинного обучения и данных через использование консенсусов направлен на усиление безопасности, приватности и коллаборативности. В контексте этих разработок сочетание больших языковых моделей и блокчейн-технологий открывает новые возможности, совмещая контекстуальное понимание и творческие способности языковых моделей с безопасностью и прозрачностью блокчейна, предлагая эффективное синергетическое взаимодействие [20].

Несмотря на ряд работ, посвященных децентрализованным большим языковым моделям [12–14], в них не использовались технологии блокчейна, которые могли бы усовершенствовать дообучение модели, защиту от сбоев и хранение информации распределенно, благодаря надежным механизмам стимулирования (криптовалютами) и протоколам консенсуса. Такая интеграция может значительно расширить процессы обучения и вывода, сделав их более эффективными и быстрыми. Кроме того, блокчейн может способствовать динамическому дообучению LLM посредством обновления версий на основе сетевого консенсуса, где версия модели при подключении к сети обновляется до последней. Хотя в некоторых исследованиях [16, 21–22] интеграция уже изучалась, в основном они были посвящены использованию LLM в блокчейне специально для улучшения децентрализованных приложений. В работе предлагается новая архитектура, которая способствует динамическому дообучению и локальным выводам LLM с системой поощрений, основанных на консенсусах. Эта среда поощряет участие пользователей, предлагая стимулы (криптовалюту) для предоставления данных, памяти и вычислительных ресурсов, используя механизмы консенсуса для обеспечения целостности сети, динамического обновления моделей и облегчения версионирования моделей.

## III. Архитектура динамической децентрализованной большой языковой модели

### A. *Обзор*

Веса в предлагаемой архитектуре первоначально устанавливаются с помощью методов трансферного обучения, которые затем распространяются по сети в одноранговом режиме. За этим шагом следует локальное обновление модели с использованием подходов федеративного обучения для уточнения модели на децентрализованных узлах. Затем локальные модели объединяются в новую, расширенную версию LLM, которая снова распространяется по сети. В основе всего процесса лежит блокчейн в качестве механизма для обновления и контроля версий LLM, включающий криптовалютную схему поощрения для мотивации участников сети.

Упрощенная схема предлагаемой архитектуры представлена на рис. 1.

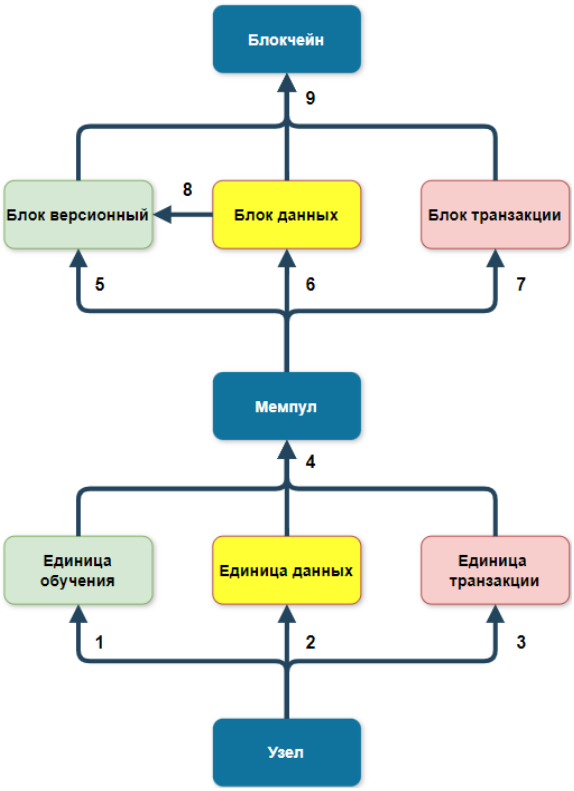


Рис. 1. Архитектура динамической децентрализованной большой языковой модели на основе блокчейна

### B. *Компоненты ахритектура*

Предлагаемая архитектура модели состоит из следующих элементов, представленных на рис. 1:

- **Узел** (*англ. Node*) – конечные устройства сети, предоставляющие вычислительные ресурсы и данные.
- **Единица обучения** (*англ. Learning unit*) – обновленные веса модели, полученные узлами в результате локального обучения на своих наборах данных.
- **Единица данных** (*англ. Data unit*) – фрагмент набора данных, который узлы передают по сети.
- **Единица транзакции** (*англ. Transaction unit*) – транзакции монет между узлами в сети.
- **Мемпул** (*англ. Mempool*) – место для хранения единиц данных, результатов единиц обучения, записи единиц транзакции перед их обновлением в новые блоки.
- **Блок транзакции** (*англ. Transaction block*) – список из нескольких транзакций, выбранных из пула памяти.
- **Блок данных** (*англ. Data block*) – блок, включающий транзакцию coinbase для получения вознаграждения и компиляцию блоков данных из Мемпула, формирующих тестовый набор данных.
- **Версионный Блок** (*англ. Version block*) – блок, содержащий транзакцию coinbase за вознаграждение и агрегированные веса, формирующие глобальную модель, полученную из комбинации единиц обучения Мемпула.

### *C. Описание работы*

Работа модели основана на представленной архитектуре (рис.1) начинается с установки начальных весов на выбранном узле путем трансферного обучения, используя веса из предварительно обученной модели, а не по принципу случайного выбора. Затем эти веса распространяются по одноранговой сети (P2P), позволяя узлам индивидуально обучать модель на своих данных. После обучения узлы обновляют веса модели и переводят эти обновления в блок обучения (рис. 1, 1). Этот блок защищается с помощью шифрования с закрытым ключом [22], создавая уникальную цифровую подпись, и впоследствии отправляется в Мемпул [23]. Для поддержки последующего комбинирования моделей применяется гомоморфное шифрование [22].

Одновременно узлы совершают криптовалютные транзакции в блокчейне, и эти транзакции (рис. 1, 3) также направляются в Мемпул (рис. 1, 4). Более того, некоторые узлы предоставляют тестовые наборы данных, шифруя и подписывая единицы данных в процессе (рис. 1, 2), схожем с процессом обучения образом, перед тем как отправить их в Мемпул (рис. 1, 4), где гомоморфное шифрование позволяет выполнять последующие операции.

После этого этапа механизм PoS выбирает узлы на основе размера их криптовалютных аткивов для создания новых блоков, добавляя определенную степень случайности для обеспечения разнообразного выбора создателей блоков. Эти узлы извлекают, проверяют и собирают транзакции из Мемпул в новый блок, который включает специальную транзакцию coinbase [23] в качестве вознаграждения для создателя блока (рис. 1, 7). Затем этот блок распространяется по сети для проверки целостности. Если большинство узлов подтверждают транзакции, блок интегрируется в блокчейн (рис. 1, 9); в противном случае недействительные транзакции приводят к потере доли создателя блока, что стимулирует тщательную проверку.

Система блокчейна устанавливает ограничение на частоту блоков транзакций, требуя добавления блока данных (рис. 1, 5) и версионного блока (рис. 1, 6) перед любыми последующими блоками транзакций. Эта процедура использует комбинацию механизмов PoS, Proof-of-Time (PoT) и PoW. Для генерации блока данных PoS определяет выбор узла на основе заданных узлами ставок, при назначении случайного вызова (номера), который обрабатывается с помощью функции верифицируемой задержки (VDF). Узлы выбирают и улучшают блоки данных из Мемпула (рис. 1, 5) для повышения качества набора данных. Создатель набора данных наивысшего качества формирует новый блок данных, который затем аутентифицируется сетью с помощью выходных данных VDF. Создатели проверенных блоков данных получают вознаграждение через транзакцию coinbase, отражающее их вклад в качество набора данных. Узлы, искажающие качество данных или манипулирующие VDF, рискуют лишиться вознаграждения.

В соответствии с установленной политикой частоты блоков, блокчейн останавливает включение блоков, не относящихся к транзакциям, до тех пор, пока не будет добавлен блок новой версии (рис. 1, 8). Используя механизм PoS, узлы выбираются для создания нового версионного блока. Этот процесс отбора, подобный тому, что используется при создании блока данных, направлен на объединение обучаемых блоков для повышения производительности модели. Уникальным аспектом этого этапа является разделение тестового набора данных, полученного из предыдущего блока данных, на различные партии (*англ. batches*). Затем эти партии распределяются между выбранными узлами. Выдача вызова вместе с этими разнообразными партиями данных побуждает узлы применять усреднение с помощью федеративного усреднения [24] или аналогичными методами для уточнения агрегированной модели на основе полученных данных.

После завершения процесса верификации с помощью функции проверяемой задержки узла, достигшему наивысшей производительности по всему набору тестовых данных, поручается создание блока новой версии. Далее блок проходит верификацию во всей сети. Успешный процесс проверки не только подтверждает правильность версионного блока, но и вызывает вознаграждение как для создателя новой версии блока, так и для узлов, предоставивших единицы обучения, использованные при его разработке. И наоборот, любой узел, уличенный в злоупотреблениях во время этого процесса, рискует потерять свою долю, что обеспечивает целостность и поощряет честное участие в работе блокчейна.

После успешной агрегации и проверки обновленной модели, заключенной в версионном блоке блокчейна, узлы получают возможность использовать эту улучшенную модель для решения задач вывода. У них есть возможность либо использовать свои локальные вычислительные ресурсы, что может привести к менее мощным выводам по сравнению с использованием коллективных вычислительных возможностей всей сети, либо получить доступ к более широкой вычислительной мощности и ресурсам сети через механизм обмена монетами с другими узлами. Система поощрений подразумевает сбалансированность вклада и выгоды для всей сети: предполагается, что средний узел будет зарабатывать на предоставлении единиц данных и единиц обучения примерно столько же, сколько он потратит на доступ к вычислительным ресурсам для выводов. Этот подход направлен на минимизацию стоимости использования системы для среднего узла. В отличие от этого, крупные узлы, требующие более обширных выводов и высокой точности, могут понести дополнительные расходы на заимствование необходимых вычислительных мощностей из сети.

## IV. Обсуждение

Представленная архитектура сталкивается с рядом проблем, главная из которых – получение вредоносных данных, влияющих на результаты моделирования. Также использование неидентичных и неидентично распределенные данные (non-IID) и перекос данных могут повлиять на эффективность дообучения модели. Для решения проблемы качества наборов данных, необходимых для создания блоков данных и версионных блоков, может быть использован комбинированный

подход, сочетающий различные метрики качества текста, например, статистические показатели, такие как показатели качества содержания; индекс Ганнинга-Фога; показатели разнообразия и сложности; мера текстового лексического разнообразия; а также показатели новизны и избыточности, например, коэффициент Жаккара. Оценка эффективности модели может включать в себя совокупность таких метрик, как accuracy, F1 Score, BLEU Score и ROUGE Score. Эти оценки должны быть вычислительно эффективными, чтобы обеспечить возможность проверки узлами в рамках протокола консенсуса архитектуры.

Еще одна проблема — управление большим объемом данных в блокчейне, сохраняя масштабируемость. Решения могут включать в себя стратегии вне цепочки (*англ. off-chain*), такие, как децентрализованные сети хранения (DSN) или якорение данных, а также шардинг данных и решения для масштабирования второго уровня. Кроме того, задержка, часто встречающаяся в сетях блокчейн, создает проблемы и в нашей архитектуре. Для ее снижения можно оптимизировать механизмы консенсуса, использовать передовые сетевые протоколы, такие как усовершенствованные протоколы Gossip, применять методы шардинга и исследовать решения второго уровня. Наконец, сложность реализации этой архитектуры и протоколов, обеспечивающих целостность блокчейна и, сохраняющих конфиденциальность и доступность системы с использованием локальной памяти со снижением предвзятости, весьма значительна. Успешная реализация требует пристального внимания к деталям и следования принципам архитектуры для решения проблем перенаправления вычислительных мощностей с вычисления криптографических задач блокчейна на процессы использования и дообучения LLM.

## III. Заключение

В данной статье описывается теоретическая архитектура, объединяющая технологии LLM и блокчейна для создания масштабируемого и распределенного искусственного интеллекта, одновременно решая проблемы конфиденциальности и централизованного контроля. Потенциал такого слияния огромен, оно может повысить эффективность LLM и расширить доступность для пользователей.

Тем не менее, архитектура сталкивается с такими проблемами, как обеспечение конфиденциальности данных, целостности модели и соответствия нормативным требованиям. Ключевые шаги для дальнейшего развития архитектуры включают детальную доработку архитектуры, протоколов консенсуса и разработку пилота программной реализации для тестирования.

В дальнейших работах также предстоит синтезировать эффективные метрики и алгоритмы для оценки данных и производительности, а также решить проблему загруженности блокчейна данными.

## Список литературы

[1] Caselles-Dupré H., Lesaint F., Royo-Letelier J. Word2vec applied to Recommendation: Hyperparameters Matter // arXiv. Aug 2018. URL: https://arxiv.org/abs/1804.04212 (дата обращения: 08.01.2024).

[2] Karampatsis R., Sutton C. SCELMO: SOURCE CODE EMBEDDINGS FROM LANGUAGE MODELS // arXiv. Apr 2020. URL: https://arxiv.org/abs/2004.13214 (дата обращения: 08.01.2024).

[3] Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A., Kaiser Ł., Polosukhin I. Attention is all you need In Advances in neural information processing systems // arXiv. Jun 2017. URL: https://arxiv.org/abs/1706.03762 (дата обращения: 08.01.2024).

[4] Alaparthi S., Mishra M. Bidirectional Encoder Representations from Transformers (BERT): A sentiment analysis odyssey // arXiv. Jul 2020. URL: https://arxiv.org/abs/2007.01127 (дата обращения: 08.01.2024).

[5] Yenduri G., Ramalingam M., Chemmalar G., Supriya Y., Srivastava, Maddikunta G., Raj G., Jhaveri H., Prabadevi B., Wang W., Vasilakos V., Gadekallu T. GPT (Generative Pre-trained Transformer) – A Comprehensive Review on Enabling Technologies, Potential Applications, Emerging Challenges, and Future Directions // arXiv. May 2023. URL: URL: https://arxiv.org/abs/2305.10435 (дата обращения: 08.01.2024).

[6] Reid F., Harrigan M. An Analysis of Anonymity in the Bitcoin System // arXiv. May 2012. URL: https://arxiv.org/abs/1107.4524 (дата обращения: 29.01.2024).

[7] Pavloff U., Amoussou-Guenou Y., Tucci-Piergiovanni S. Ethereum Proof-of-Stake and the Probabilistic Bouncing Attack // arXiv. Oct 2022. URL: https://arxiv.org/abs/2210.16070 (дата обращения: 30.01.2024).

[8] Garst S., Dekker J., Reinders M. A comprehensive experimental comparison between federated and centralized learning in bioRxiv // DOI: 10.1101/2023.07.26.550615. Jul 2023. URL: https://www.researchgate.net/publication/372759901_A_comprehensive_experimental_comparison_between_federated_and_centralized_learning (дата обращения: 07.02.2002).

[9] You X., Liu X., Lin X., Cai J., Chen S. Accuracy Degrading: Toward Participation-Fair Federated Learning // IEEE Internet of Things Journal. DOI: 10.1109/JIOT.2023.3238038. Jun 2023. URL: https://ieeexplore.ieee.org/document/10021580 (дата обращения: 07.02.2024).

[10] Bose A., Bai L. A Fully Decentralized Homomorphic Federated Learning Framework // IEEE 20th International Conference on Mobile Ad Hoc and Smart Systems (MASS). DOI: 10.1109/MASS58611.2023.00029. Sep 2023. URL: https://ieeexplore.ieee.org/document/10298305 (дата обращения: 14.02.2024).

[11] Yu G., Wang X., Sun C., Wang Q., Yu P., Ni W., Liu R., Xu X. IronForge: An Open, Secure, Fair, Decentralized Federated Learning // arXiv. Jan 2023. URL: https://arxiv.org/abs/2301.04006 (дата обращения: 14.02.2024).

[12] Tang Z., Wang Y., He X., Zhang L., Pan X., Wang Q., Zeng R., Zhao K., Shi S., He B., Chu X. FusionAI: Decentralized Training and Deploying LLMs with Massive Consumer-Level GPUs in Symposium on Large Language Models (LLM 2023) with IJCAI 2023, Macao, China // arXiv. Aug 2023. URL: https://arxiv.org/abs/2309.01172 (дата обращения: 28.02.2024).

[13] Chen C., Feng X., Zhou J., Yin J., Zheng X. Federated Large Language Model: A Position Paper, Zhejiang University, Hangzhou, China // arXiv. 18 Jul 2023. URL: https://arxiv.org/abs/2307.08925 (дата обращения: 29.02.2024).

[14] Zhao J., Song Y., Liu S., Harris I., Jyothi S. LinguaLinked: A Distributed Large Language Model // arXiv. Dec 2023. URL: https://arxiv.org/abs/2312.00388 (дата обращения: 29.02.2024).

[15] Sheng Y., Zheng L., Yuan B., Li Z., Ryabinin M., Fu D., Xie Z., Chen B., Barrett C., Gonzalez J., Liang P., Re´ C., Stoica I., Zhang C. FlexGen: High-Throughput Generative Inference of Large Language Models with a Single GPU // arXiv. Jun 2023. URL: https://arxiv.org/abs/2303.06865 (дата обращения: 06.03.2024).

[16] Alizadeh K., Mirzadeh I., Belenko D., Khatamifard S., Cho M., Mundo C., Rastegari M., Farajtabar M. LLM in a flash: Efficient Large Language Model Inference with Limited Memory // arXiv. Jan 2024. URL: https://arxiv.org/html/2312.11514v2 (дата обращения: 06.03.2024).

[17] Nguyen C., Thai H., Nyato D., Nguyen N., Dutkiewicz E. Proof-of-Stake Consensus Mechanisms for Future Blockchain Networks // IEEE Access. DOI: 10.1109/ACCESS.2019.2925010. Jun 2019. URL: https://ieeexplore.ieee.org/document/8746079?ref=hackernoon.com (дата обращения: 13.03.2024).

[18] Xu Y., Yang X., Zhang J., Zhu J., Sun M., Chen B. Proof of Engagement: A Flexible Blockchain Consensus Mechanism in Wireless Communications and Mobile Computing, Hindawi // DOI: 10.1155/2021/6185910. Aug 2021. URL: https://www.researchgate.net/publication/354037180_Proof_of_Engagement_A_Flexible_Blockchain_Consensus_Mechanism (дата обращения: 13.03.2024).

[19] Do T., Nguyen T., Pham H. Delegated Proof of Reputation: a novel Blockchain consensus, December 2019, arXiv:1912.04065v1 (дата обращения: 14.03.2024).

[20] Liu X. Decentralized Machine Learning on a Blockchain: Case Studies // 2023 IEEE International Conference on Omni-layer Intelligent Systems (COINS). DOI: 10.1109/COINS57856.2023.10189230. Jul 2023. URL: https://ieeexplore.ieee.org/document/10189252 (дата обращения: 21.03.2024).

[21] Mboma J., Tshipata O., Kyamakya K., Kambale W. Assessing How Large Language Models Can Be Integrated with or Used for Blockchain Technology: Overview and Illustrative Case Study in 2023 27th International Conference on Circuits, Systems, Communications and Computers (CSCC) // DOI: 10.1109/CSCC58962.2023.00018. Apr 2023. URL: https://www.researchgate.net/publication/376826315_Assessing_How_Large_Language_Models_Can_Be_Integrated_with_or_Used_for_Blockchain_Technology_Overview_and_Illustrative_Case_Study (дата обращения: 21.03.2024).

[22] Liang W., Zhang D., Lei X., Tang M., Li K., Zomaya A. Circuit Copyright Blockchain: Blockchain-Based Homomorphic Encryption for IP Circuit Protection // IEEE Transactions on Emerging Topics in Computing. DOI. 10.1109/TETC.2020.2993032. Jul.-Sep. 2021. (дата обращения: 27.03.2024).

[23] Saad M., Njilla L., Kamhoua C., Kim J., Nyang D., Mohaisen A. Mempool optimization for Defending Against DDoS Attacks in PoW-based Blockchain Systems // 2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). DOI: 10.1109/BLOC.2019.8751476. May 2019. URL: https://ieeexplore.ieee.org/abstract/document/8751402 (дата обращения: 28.03.2002).

[24] Plassier V., Durmus A., Moulines É. Federated Averaging Langevin Dynamics: Toward a unified theory and new algorithms // arXiv. Oct 2022. URL: https://arxiv.org/abs/2211.00100v1 (дата обращения: 01.04.2024).

# Реализация преобразования Карунена–Лоэва в классе многослойных самоподобных нейронных сетей

А. Ю. Дорогов

*Санкт-Петербургский электротехнический университет*
*«ЛЭТИ» им. В.И. Ульянова (Ленина)*

vaksa2006@yandex.ru

***Аннотация.*** **В работе представлен метод реализации произвольных одномерных и двумерных линейных преобразований в классе самоподобных нейронных сетей, акцент сделан на построение преобразования Карунена–Лоэва. Возможность нейросетевой реализации позволяет организовать высокоскоростную конвейерную обработку на специализированных процессорах и таким образом снять ограничения по размерности при использовании данного преобразования. В статье факторизованная форма линейного преобразования рассматривается как многослойная нейронная сеть, а вычисление коэффициентов факторизованной формы как обучение нейронных ядер сети. Алгоритм обучения не имеет обратных связей по ошибке, является абсолютно сходящимся и время его выполнения сопоставимо с временем обработки данных в сети.**

***Ключевые слова: преобразование Карунена–Лоэва; факторизация; быстрая нейронная сеть; топологические модели***

## I. Введение

Для задач классификации и распознавания сигналов существенное значение имеют процедуры предварительной обработки, ориентированные на устранение избыточности и выделение информативных признаков. Использование ортогональных преобразований для этих целей позволяет представить информацию, содержащуюся в исходном сигнале в виде взаимно-независимых спектральных составляющих. Поскольку энергия сигнала определяется суммой квадратов спектральных коэффициентов, то по модулю спектрального коэффициента можно непосредственно судить о значимости информативного признака.

Известно, что максимальное сокращение избыточности данных обеспечивается ортогональным преобразованием Карунена–Лоэва, которое образуется собственными векторами ковариационной матрицы сигналов. Преобразование впервые было предложено Хотеллингом [1] позже переоткрыто Каруненом и Лоэвом [2, 3]. Однако практическое использование данного преобразования сопряжено со значительными вычислительными затратами, которые квадратично растут с увеличением размерности преобразования. По этой причине метод Карунена–Лоэва в настоящее время не применяется для обработки данных высокой размерности.

Традиционно для обработки сигналов используются ортогональные преобразования, обладающие быстрыми алгоритмами выполнения. Быстрые алгоритмы основаны на возможности факторизации преобразования в произведение слабозаполненных матриц, каждую из которых можно интерпретировать как слой нейронной сети. Факторизация преобразований решает две задачи, во-первых сокращает общее число вычислительных операций и во-вторых позволяет организовать высокоскоростную конвейерную обработку на специализированных процессорах. Общеизвестно, что преобразование Карунена–Лоэва не имеет быстрого алгоритма. Это утверждение действительно справедливо для процессоров последовательного типа, но этот факт не связан с факторизацией. В этой статье будет показано, что факторизация преобразования Карунена–Лоэва возможна, и, следовательно, возможна его быстрое выполнение на специализированных процессорах конвейерного типа.

## II. Базовая схема факторизации быстрых преобразований

В основе построения схемы факторизации лежат классические алгоритмы быстрых преобразований. В работе [4] показано, что они являются частным случаем многослойных самоподобных нейронных сетей. На рис. 1 представлен граф быстрого преобразования Фурье размерности 8 с топологией Кули–Тьюки «с прореживанием по времени». В каждом слое выделены четыре базовых операции типа «Бабочка». Для преобразования Фурье параметры базовых операций полностью определены, однако, если полагать, что их параметры можно изменять, то мы приходим к варианту быстрых нейронных сетей (БНС) [4]. В этом контексте базовые операции уместно назвать нейронными ядрами.

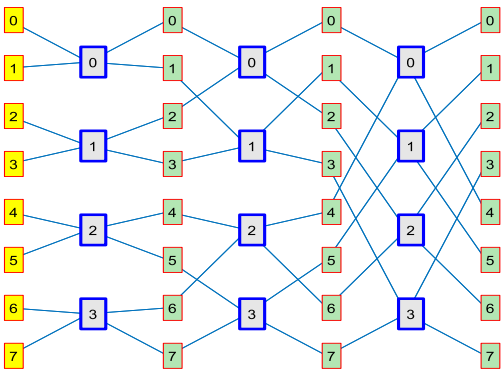


Рис. 1. Граф быстрого преобразования Фурье

Сетевая модель данной топологии и описывается набором кортежей [4]:

$$\begin{aligned} U^m &= \langle u_{n-1}u_{n-2}\cdots u_{m+1}u_m v_{m-1}v_{m-2}\cdots v_1 v_0\rangle, \\ V^m &= \langle u_{n-1}u_{n-2}\cdots u_{m+1}v_m v_{m-1}v_{m-2}\cdots v_1 v_0\rangle, \\ z^m &= \langle u_{n-1}u_{n-2}\cdots u_{m+1}v_{m-1}v_{m-2}\cdots v_1 v_0\rangle. \end{aligned} \quad (1)$$

где $z^m$ порядковый номер нейронного ядра, $U^m, V^m$ – порядковые номера рецепторов и аксонов в слое $m$, $u_m, v_m$ – локальные номера рецепторов и аксонов в пределах ядра слоя $m$. Каждый кортеж представляет собой поразрядную форму представления порядкового номера через разрядные переменные $u_m, v_m$. Основания разрядных переменных $u_m, v_m$ определяются целыми положительными числами, и могут быть заданы соответствиями:

$$\begin{pmatrix} u_0 & u_1 & \cdots & u_{n-2} & u_{n-1} \\ p_0 & p_1 & \cdots & p_{n-2} & p_{n-1} \end{pmatrix}, \quad \begin{pmatrix} v_0 & v_1 & \cdots & v_{n-2} & v_{n-1} \\ g_0 & g_1 & \cdots & g_{n-2} & g_{n-1} \end{pmatrix}.$$

Для ядер размерности 2×2 все разрядные переменные принимают значения {0,1}. Координатные направления $U^m$ и $V^m$ в дальнейшем будем называть входной и выходной плоскостью нейронного слоя. Для терминальных плоскостей (относящихся к начальному и конечному слою сети) будем использовать обозначения $U$ и $V$. Для построения быстрых алгоритмов размерность преобразования должна быть составным числом и чем больше множителей в разложении размерности, тем выше вычислительная эффективность быстрого алгоритма. Размерности быстрой нейронной сети по входу и выходу вычисляются через произведения оснований разрядных переменных:

$$N = p_{n-1}\ldots p_1 p_0, \quad M = g_{n-1}\ldots g_1 g_0.$$

Число слоёв в быстрых преобразованиях равно число сомножителей в этих произведениях. Несмотря на большое разнообразие быстрых алгоритмов, конфигурации их структур удовлетворяют системному инварианту самоподобия [4]. Как известно таким же свойством самоподобия обладают фракталы. Поэтому быстрые алгоритмы можно интерпретировать как квазифракталы. Свойство структурной фрактальности позволяет решить одновременно две задачи: реализовать быструю обработку данных и выполнить быстрое обучение преобразования. Обучение быстрого преобразования заключается в выборе значений элементов нейронных ядер, так чтобы в столбцах матрицы преобразования содержался заданный набор опорных функций, это могут быть, например, функции базиса Карунена–Лоэва или другого линейного преобразования. В [4] показано, что для самоподобных нейронных сетей элементы матрицы быстрого преобразования могут быть выражены через произведения элементов нейронных ядер:

$$h(U,V) = w_{z^{n-1}}^{n-1}(u_{n-1}, v_{n-1})\, w_{z^{n-2}}^{n-2}(u_{n-2}, v_{n-2})\cdots w_{z^0}^{0}(u_0, v_0).$$

Там же доказано, что произвольная функция, заданная на дискретном интервале длиной $N = p_{n-1}\ldots p_1 p_0$ также может быть представлена в мультипликативной форме:

$$f(u) = \phi_{i^0}(u_0)\,\phi_{i^1}(u_1)\ldots\phi_{i^{n-2}}(u_{n-2})\,\phi_{i^{n-1}}(u_{n-1}).$$

где $i^m = \langle u_{n-1}u_{n-2}\ldots u_{m+1}\rangle$. Отсюда следует правило настройки нейронных ядер

$$w_{z^m}^m(u_m, v_m) = \phi_{i^m}^k(u_m) \quad (2)$$

Здесь $k$ – номер опорной функции. Зададим точку привязки функции в выходной плоскости числом, представленным в поразрядной форме:

$$x = \langle x_{n-1}x_{n-2}\cdots x_0\rangle,$$

тогда номер настраиваемых ядер по слоям будет определяться выражением:

$$z^m = \langle u_{n-1}u_{n-2}\cdots u_{m+1}x_{m-1}x_{m-2}\cdots x_1 x_0\rangle.$$

Для $m=0$ имеем $z^0 = \langle u_{n-1}u_{n-2}\cdots u_1\rangle$, это означает, что независимо от выбора точки привязки, все ядра слоя будут настраиваться, причём номер ядра определяется из условия: $z^0 = i^0$. Настройка элементов ядер этого слоя выполняется по правилу:

$$w_{z^0}^0(u_0, v_0) = \phi_{i^0}^k(u_0),$$

Очевидно, должно быть задано взаимно-однозначное соответствие $k \leftrightarrow v_0$ между номером опорной функции и разрядной переменной $v_0$. Эта разрядная переменная принимает значения $0,1,\cdots,g_0-1$. Отсюда следует вывод, что число опорных функций не может быть больше чем $g_0$, а если ещё потребовать выполнения условия ортогональности ядер, то матрица такого преобразования будет содержать только одну произвольную функцию в качестве столбца. Этого явно недостаточно для реализации произвольного ортогонального базиса Карунена–Лоэва, что согласуется с утверждением об отсутствии быстрого алгоритма для преобразования Карунена–Лоэва. Однако, в следующем разделе будет показано, что отсутствие быстрого алгоритма не препятствует процедуре факторизации преобразования.

## III. Самоподобные нейронные сети с дополнительными плоскостями

Дополним топологическую модель быстрого преобразования (1) дополнительными плоскостями нейронных ядер [5], номера которых определим переменной $\pi_m$, новая топологическая модель в этом случае будет описываться набором кортежей:

$$\begin{aligned} U^m &= \langle u_{n-1}u_{n-2}\cdots u_{m+1}u_m v_{m-1}v_{m-2}\cdots v_1 v_0\rangle, \\ V^m &= \langle u_{n-1}u_{n-2}\cdots u_{m+1}v_m v_{m-1}v_{m-2}\cdots v_1 v_0\rangle, \\ z^m &= \langle u_{n-1}u_{n-2}\cdots u_{m+1}v_{m-1}v_{m-2}\cdots v_1 v_0\rangle, \\ \pi_m &= \langle v_{n-1}v_{n-2}\cdots v_{m+2}v_{m+1}\rangle. \end{aligned}$$

Максимальное количество дополнительных плоскостей появится в нулевом слое. Номер плоскости в нулевом слое будет определяться кортежем $\pi_0 = \langle v_{n-1}v_{n-2}\cdots v_2 v_1\rangle$. Плоскость с номером $\pi_0 = 0$ будем

считать плоскостью исходной базовой топологической структуры. Число плоскостей в нулевом слое будет равно произведению оснований: $g_{n-1}g_{n-2}\cdots g_2 g_1$. По мере движения к выходному слою число дополнительных плоскостей будет уменьшаться и для последнего слоя $\pi_{n-1}=\langle\ \rangle$, т. е. их не будет совсем, останется только одна плоскость базовой топологии. Таким образом, в новой топологии плоскость последнего слоя останется прежней, а в младших слоях появятся дополнительные плоскости. Номер ядра, теперь следует уточнять его размещением в дополнительной плоскости. На рис. 2 показана новая топология, построенная на базе трёхслойной сети с основаниями 2.

В сети условно показаны коммутаторы (SW), которые служат для пояснения принципа работы сети. Коммутаторы управляются разрядными переменными точек выходной плоскости. Фактически коммутаторов нет, они реализуются в составе алгоритма и не препятствуют параллельной обработки входных данных. В контексте спектрального преобразования точки выходной плоскости точки ассоциируются со спектральными коэффициентами, поэтому привязка спектрального коэффициента к координатам выходной плоскости предопределяет выбор дополнительных плоскостей, которые используются для обработки входных образов. При построении полного спектрального анализатора все векторные входы параллельно объединяются.

Поскольку правило порождения новых плоскостей не противоречит базовой топологической модели, то для настройки ядер преобразования к эталону можно использовать прежнее правило (2), расширив его аргументом $\pi_m$ для дополнительных плоскостей:

$$w^m_{z^m}\langle\pi_m\rangle(u_m,x_m)=\phi^k_{i^m}(u_m),$$

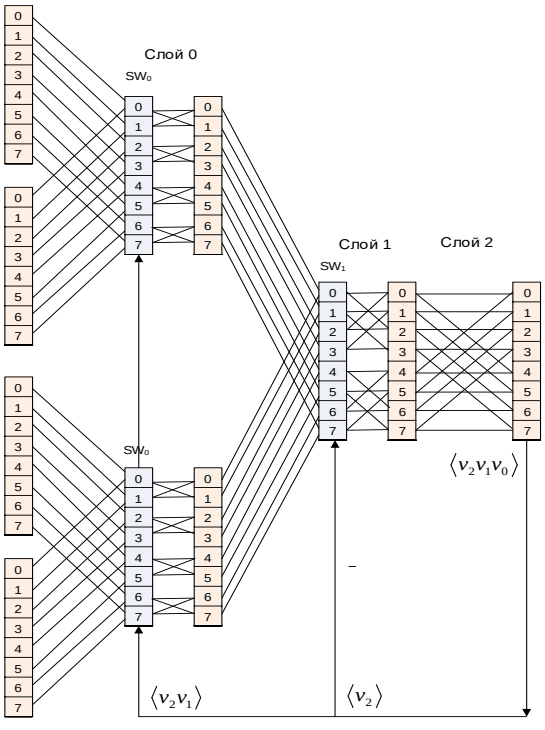


Рис. 2. Топология самодобной нейронной сети с дополнительными плоскостям

здесь $k$ – номер эталонной функции, $x=\langle x_{n-1}x_{n-2}\cdots x_0\rangle$, точка привязки, $z^m=\langle u_{n-1}u_{n-2}\cdots u_{m+1}x_{m-1}x_{m-2}\cdots x_1x_0\rangle$ номер настраиваемых ядер по слоям, $\pi_m=\langle x_{n-1}x_{n-2}\cdots x_{m-1}\rangle$ – номер плоскости размещения ядер. Индекс $k$ в правой части нумерует точку приспособления. Для $m=0$ имеем $z^0=i^0=\langle u_{n-1}u_{n-2}\cdots u_1\rangle$, а варьируемыми переменными в левой части являются номер плоскости $\pi_0=\langle x_{n-1}x_{n-2}\cdots x_1\rangle$ и разряд $x_0$. Вместе они покрывают весь диапазон координат выходной плоскости. Этому диапазону отвечают возможные значения индекса $k$ в правой части, отсюда следует, что каждая точка выходной плоскости может быть связана с собственной опорной функцией. Т.е. построенная сеть обладает максимально возможным числом точек привязки, покрывающих всю выходную плоскость и, таким образом может быть использована для реализации произвольного линейного преобразования размерности $N\times M$. В частности, при реализации ортогонального преобразования Карунена–Лоэва для которого выполняется $N=M$.

## IV. Факторизация двумерных преобразований

Обозначим через $F(U_y,\ U_x)$ матрицу изображения размером $N_y\times N_x$. При воздействии на изображение линейного преобразования $H(U_y,U_x;V_y,V_x)$ на выходе получается массив из $M_y\times M_x$ коэффициентов.

Необходимым условием существования быстрого алгоритма является возможность мультипликативной декомпозиции значений обоих размерностей изображения в равное число сомножителей:

$$N_y=p_0^y p_1^y\ldots p_{n-1}^y,\qquad M_y=g_0^y g_1^y\ldots g_{n-1}^y,$$
$$N_x=p_0^x p_1^x\ldots p_{n-1}^x.\qquad M_x=g_0^x g_1^x\ldots g_{n-1}^x.$$

Индексы $x, y$ здесь означают принадлежность к осям координат исходного изображения. Используя сомножители декомпозиций как основания системы счисления, представим значения координат точек изображения в позиционной форме:

$$U_y=\langle u^y_{n-1}u^y_{n-2}\cdots u^y_1u^y_0\rangle,\quad U_x=\langle u^x_{n-1}u^x_{n-2}\cdots u^x_1u^x_0\rangle,$$

Аналогично можно выразить координаты спектральных коэффициентов в плоскости $[V_y,V_x]$;

$$V_y=\langle v^y_{n-1}v^y_{n-2}\cdots v^y_1v^y_0\rangle,\quad V_x=\langle v^x_{n-1}v^x_{n-2}\cdots v^x_1v^x_0\rangle.$$

Как и в одномерном случае можно доказать [4], что элементы четырехмерной матрицы быстрого преобразования выражаются через элементы нейронных ядер:

$$H(U_y,U_x;V_y,V_x)=W^{n-1}_{z_x^{n-1},\,z_y^{n-1}}(u^y_{n-1}u^x_{n-1};v^y_{n-1}v^x_{n-1})\ldots$$
$$\ldots W^0_{z_x^0,z_y^0}(u^y_0u^x_0;v^y_0v^x_0),$$

Доказывается также, что произвольная двумерная функция, заданная на дискретном поле представима в мультипликативной форме:

$$F\left(U_y, U_x\right) = f_{j^y_{n-1} j^x_{n-1}}\left(u^y_{n-1}, u^x_{n-1}\right) f_{j^y_{n-2} j^x_{n-2}}\left(u^y_{n-2}, u^x_{n-2}\right)\ldots$$
$$\ldots f_{j^y_1 j^x_1}\left(u^y_1, u^x_1\right) f_{j^y_0 j^x_0}\left(u^y_0, u^x_0\right).$$

Будем использовать прежнюю топологическую модель быстрого преобразования (1), но в двумерном варианте. Нарастим модель дополнительными плоскостями, номера плоскостей в пределах слоя определим правилом:

$$\pi_m = \left\langle v^x_{n-1} v^x_{n-2} \ldots v^x_{m+1} v^y_{n-1} v^y_{n-2} \ldots v^y_{m+1} \right\rangle .$$

Очевидно, что максимальное количество дополнительных плоскостей будет в нулевом слое, а по мере увеличения номера слоя количество дополнительных плоскостей будет уменьшаться, и в последнем слое получим $\pi_{n-1} = \langle\ \rangle$, т. е. дополнительных плоскостей не будет совсем. Таким образом, в новой топологии плоскость последнего слоя останется прежней, а в младших слоях появятся дополнительные плоскости.

Архитектура сети с дополнительными плоскостями показана на рис. 3. При выполнении спектральной обработки входное изображение подаётся одновременно на все плоскости входного слоя. Слои сети разделены коммутаторами, которые управляются разрядными переменными координатных чисел выходной плоскости спектральных коэффициентов.

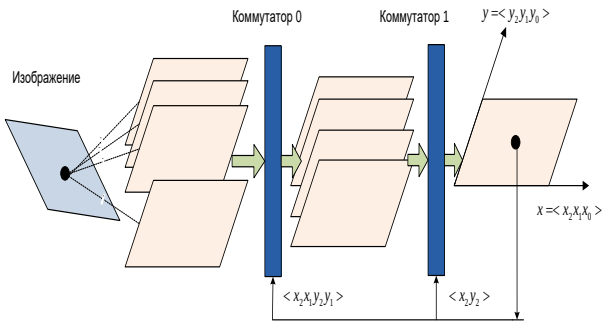


Рис. 3. Архитектура двумерной регулярной нейронной сети с дополнительными плоскостями

Для настройки ядер преобразования можно использовать правило [4]:

$$W^m_{z^m_x, z^m_y}\left\langle \pi_m \right\rangle \left(u^y_m u^x_m; v^y_m v^x_m\right) = f^k_{j^m_x j^m_y}\left(u^y_m, u^x_m\right).$$

Индекс $k$ в правой части нумерует точку привязки. Для $m = 0$ имеем $z^0_x = j^0_x$ и $z^0_y = j^0_y$, а варьируемыми переменными в левой части являются разрядные переменные $v^y_0 v^x_0$ и номер плоскости $\pi_0 = \left\langle x_{n-1} x_{n-2} \ldots x_1, y_{n-1} y_{n-2} \ldots y_1 \right\rangle$ (здесь разрядные переменные $y_*, x_*$ определяют координаты точки привязки в плоскости $\left[V_y, V_x\right]$). Вместе набор варьируемых переменных покрывают весь диапазон координат выходной плоскости. Этому диапазону отвечают возможные значения индекса $k$ в правой части последнего выражения. Таким образом, быстрое преобразование с дополнительными плоскостями может быть настроено к $D = M_y M_x$ произвольным изображениям, т. е. каждой точке выходной плоскости будет соответствовать один опорный образ, например, это может быть образ соответствующий двумерной базисной функции преобразования Карунена–Лоэва.

## V. Заключение

В статье показано, что топология одномерных и двумерных БНС легко расширяется дополнительными плоскостями, при этом число реализуемых опорных функций кардинально возрастает и покрывает все элементы выходной плоскости. Причём расширение топологии не нарушает принципа построения обучающего алгоритма. Полученные результаты позволяют построить факторизованное представление для любого линейного преобразования, в том числе и для ортогонального преобразования Карунена–Лоэва. Вычислительный эффект от факторизации достигается при использовании специализированных конвейерных вычислителей. Предложенная архитектура реализуют сети с глубокой степенью обучения. Можно показать, что они сегментируются в лес независимых пирамидальных сетей [6]. Это порождает уникальное качество – возможность дообучения сети к новым образам без изменения или потери ранее накопленных знаний. Сети могут работать в системах реального времени, поскольку время обучения не превышает времени обработки данных в сети. Алгоритмы обучения являются абсолютно устойчивыми и завершаются за конечное число шагов. Выбор структуры сети не вызывает проблем и определяется системными инвариантами самоподобных сетей. Возможны различные структурные решения, обладающие одинаковыми качествами.

## Список литературы

[1] Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. // Journal of Educational Psychology, 24, 417–441, and 498–520..

[2] K. Karhunen, Kari, Uber lineare Methoden in der Wahrscheinlichkeitsrechnung // Ann. Acad. Sci. Fennicae. Ser. A. I. Math.-Phys., 1947, No. 37, 1-79.

[3] Лоев М., Теория вероятностей, М.: ИЛ, 1962.

[4] Дорогов А.Ю. Теория и проектирование быстрых перестраиваемых преобразований и слабосвязанных нейронных сетей. СПб.: «Политехника», 2014. 328с.

[5] Дорогов А.Ю. Быстрые нейронные сети глубокого обучения // III Международная научная конференция по проблемам управления в технических системах (CTS'2019). Сборник докладов. Санкт-Петербург. 30 октября – 1 ноября 2019 г. СПб.: СПбГЭТУ «ЛЭТИ». С. 275-280.

[6] Дорогов А.Ю. Быстрые пирамидальные нейронные сети глубокого обучения для адаптивной цифровой обработки сигналов // 78-я Научно-техническая конференция Санкт-Петербургского НТО РЭС им. А.С. Попова, посвящённая Дню радио. Сборник материалов. 2023 СПб.: СПбГЭТУ «ЛЭТИ». с. 107-111. Электронное издание. ISBN 978-5-7629-3183-0.