ОБРАБОТКА СИГНАЛОВ, ИЗОБРАЖЕНИЙ, РЕЧИ, ТЕКСТА И РАСПОЗНАВАНИЕ ОБРАЗОВ

SIGNAL, IMAGE, SPEECH, TEXT PROCESSING AND PATTERN RECOGNITION

УДК 004.21
DOI: 10.37661/1816-0301-2025-22-4-36-54

Оригинальная статья
Original Article

Нейронные сети на основе обучаемого двумерного разделимого преобразования для классификации изображений: теория и аппаратная реализация на FPGA

Е. А. Кривальцевич, М. И. Вашкевич✉

Белорусский государственный университет информатики и радиоэлектроники,
ул. П. Бровки, 6, Минск, 220013, Беларусь
✉E-mail: vashkevich@bsuir.by

Аннотация

Цели. Целями исследования являются разработка методов построения компактных и эффективных нейронных сетей для задач распознавания изображений, а также их аппаратная реализация на базе программируемых логических интегральных схем (ПЛИС) типа FPGA.

Методы. Предложена концепция обучаемого двумерного разделимого преобразования (ОДРП) для построения нейронных сетей прямого распространения для задач распознавания изображений. Особенностью ОДРП является последовательная обработка строк изображения полносвязным слоем, после чего полученное представление обрабатывается по строкам вторым полносвязным слоем. В предлагаемой архитектуре нейронной сети прямого распространения ОДРП рассматривается как способ извлечения вектора признаков из исходного изображения. Аппаратная реализация нейронной сети на базе ОДРП основана на концепции вычисления «на месте» (общая память для хранения исходных и промежуточных данных), а также на использовании единого набора вычислительных ядер для расчета всех слоев нейронной сети.

Результаты. Предложено семейство компактных нейросетевых архитектур LST-1, различающихся размерностью векторного представления изображения. Эксперименты по классификации рукописных цифр базы MNIST показали высокую эффективность данных моделей: сеть LST-1-28 достигает точности 98,37 % при 9,5 тыс. параметров, а более компактная LST-1-8 показывает 96,53 % точности при 1,1 тыс. параметров. Тестирование аппаратной реализации LST-1-28 подтверждает устойчивость архитектуры к ошибкам квантования параметров.

Заключение. Предложенная концепция ОДРП обеспечивает проектирование компактных и эффективных нейросетевых архитектур, характеризующихся малым числом обучаемых параметров, высокой точностью распознавания и регулярной структурой алгоритма, что позволяет получать их эффективные реализации на базе ПЛИС.

Ключевые слова: обучаемое двумерное разделимое преобразование, нейронные сети, программируемые логические интегральные схемы, распознавание изображений, база данных MNIST

Благодарности. Авторы выражают благодарность резиденту ПВТ компании «Инженерный Центр Ядро», которая является одним из центров разработки YADRO, за предоставленное оборудование для проведения экспериментов в рамках работы совместной учебной лаборатории с Белорусским государственным университетом информатики и радиоэлектроники.

Для цитирования. Кривальцевич, Е. А. Нейронные сети на основе обучаемого двумерного разделимого преобразования для классификации изображений: теория и аппаратная реализация на FPGA / Е. А. Кривальцевич, М. И. Вашкевич // Информатика. – 2025. − Т. 22, № 4. – С. 36–54. – DOI: 10.37661/1816-0301-2025-22-4-36-54.

Конфликт интересов. Авторы заявляют об отсутствии конфликта интересов.

Поступила в редакцию | Received 16.09.2025
Подписана в печать | Accepted 07.10.2025
Опубликована | Published 30.12.2025

Neural networks based on a learnable two-dimensional separable transform for image classification: theory and hardware implementation on FPGA

Egor A. Krivalcevich, Maxim I. Vashkevich✉

Belarusian State University
of Informatics and Radioelectronics,
st. Brovki, 6, Minsk, 220013, Belarus
✉*E-mail: vashkevich @bsuir.by*

Abstract.
Objectives. Development of methods for design compact and efficient neural networks for image recognition tasks, as well as their hardware implementation based on FPGA.
Methods. The paper proposes the concept of a learnable two-dimensional separable transformation (LST) for designing feedforward neural networks for image recognition tasks. A feature of the LST is the sequential processing of image rows by a fully connected layer, after which the resulting representation is processed by columns using second fully connected layer. In the proposed architecture of a feedforward neural network, the LST is considered as a feature extractor. The hardware implementation of LST-based neural network is based on the concept of in-place computing (shared memory for storing source and intermediate data), as well as using a single set of computing cores to calculate all layers of the neural network.
Results. A family of compact neural network architectures LST-1 is proposed, differing in the image embedding size. Experiments on the classification of MNIST handwritten digits have shown the high efficiency of these models: the LST-1-28 network achieves 98.37 % accuracy with 9.5 K parameters, and the more compact LST-1-8 shows 96.53 % accuracy with 1.1 K parameters. Testing of the LST-1-28 hardware implementation confirms the architecture's resistance to parameter quantization errors.
Conclusion. The proposed concept of a learnable two-dimensional separable transformation provides the design of compact and efficient neural network architectures characterized by: a small number of learnable parameters, high recognition accuracy, and the regular structure of the algorithm, which makes it possible to obtain their effective implementations based on FPGAs.

Keywords: learnable two-dimensional separable transform, neural networks, FPGA, image recognition, MNIST dataset

Acknowledgements. The authors express their gratitude to the HTP resident company "Engineering Center Yadro", which is one of the YADRO development centers, for providing equipment for conducting experiments as part of the joint educational laboratory with the Belarusian State University of Informatics and Radioelectronics.

For citation. Krivalcevich E. A., Vashkevich M. I. *Neural networks based on a learnable two-dimensional separable transform for image classification: theory and hardware implementation on FPGA*. Informatika [*Informatics*], 2025, vol. 22, no. 4, pp. 36−54 (In Russ.). DOI: 10.37661/1816-0301-2025-22-4-36-54.

Conflict of interest. The authors declare of no conflict of interest.

Введение. Глубокие нейронные сети (НС) являются ключевыми компонентами многих систем компьютерного зрения и обработки изображений благодаря их высокой эффективности и способности моделировать сложные зависимости. Однако реализация НС на аппаратных платформах с ограниченными вычислительными ресурсами, таких как ПЛИС, сталкивается с рядом проблем, главной из которых является высокая вычислительная нагрузка, обусловленная большим числом параметров НС. Для повышения эффективности необходимо разрабатывать НС с уменьшенным числом параметров [1–5]. Такой подход позволяет снизить требования к объему памяти и вычислительным ресурсам. Однако большинство существующих архитектур с малым числом параметров основаны на классическом многослойном персептроне [6, 7], что зачастую приводит к снижению точности распознавания.

Таким образом, актуальной научной задачей является разработка эффективных архитектур НС, способных достигать компромисса между числом параметров и уровнем точности распознавания. Данный факт подтверждается и проводимыми в научной среде соревнованиями по разработке высокоскоростных и малопотребляющих реализаций НС на базе ПЛИС. В частности, на Международной конференции по обработке изображений в 2025 г. проводилось соревнование «Digit Recognition Low Power and Speed Challenge», где предлагалось разработать архитектуру ускорителя на базе ПЛИС для распознавания рукописных цифр из базы MNIST. К соревнованиям допускались только проекты, которые на тестовой выборке демонстрировали точность не менее 97,5 %.

Одним из перспективных подходов к построению компактных и эффективных НС для распознавания изображений является использование ОДРП [8]. В настоящей статье рассматриваются принципы работы ОДРП и практические аспекты реализации НС с его применением на базе ПЛИС. Экспериментальные исследования показывают, что использование ОДРП в качестве средства получения векторного представления (вложения) изображения позволяет получить компактную архитектуру НС, демонстрирующую высокие показатели точности в задаче классификации изображений. Таким образом, данная работа направлена на развитие методов построения компактных и эффективных НС для задач распознавания изображений с реализацией на ПЛИС.

Обучаемое двумерное разделимое преобразование. С вычислительной точки зрения ОДРП (англ. *LST, learned 2D separable transform*) преобразует двумерное изображение $\mathbf{X}$ размером $d_{in} \times d_{in}$ в изображение $\mathbf{Y}$ размером $d_{out} \times d_{out}$. Преобразование называется разделимым, поскольку вначале выполняется обработка строк изображения $\mathbf{X}$, после чего образуется промежуточное представление изображения, имеющее размерность $d_{in} \times d_{out}$. Далее полученное промежуточное представление обрабатывается по столбцам, в результате чего получается выходное изображение $\mathbf{Y}$ размером $d_{out} \times d_{out}$. Таким образом, разделимое преобразование над изображением можно представить как композицию двух преобразований, работающих с одномерными данными (рис. 1).

Особенностью ОДРП, представленного в работе [8], является то, что для обработки строк и столбов изображения предложено использовать однослойные полносвязные НС. Математически ОДРП можно записать в виде

$$\mathbf{Y} = LST(\mathbf{X}) = \sigma(\mathbf{W}_2\sigma(\mathbf{W}_1\mathbf{X}^T + \mathbf{b}_1)^T + \mathbf{b}_2), \quad (1)$$

где $\sigma(\cdot)$ – функция активации; $\mathbf{W}_1, \mathbf{W}_2$ – матрицы $d_{out} \times d_{in}$ линейных преобразований, описывающих полносвязные слои НС для обработки строк и столбцов соответственно; $\mathbf{b}_1, \mathbf{b}_2$ – векторы смещений размером $d_{out} \times 1$.

На рис. 2 показано графическое представление математических действий, выполняемых при реализации ОДРП.

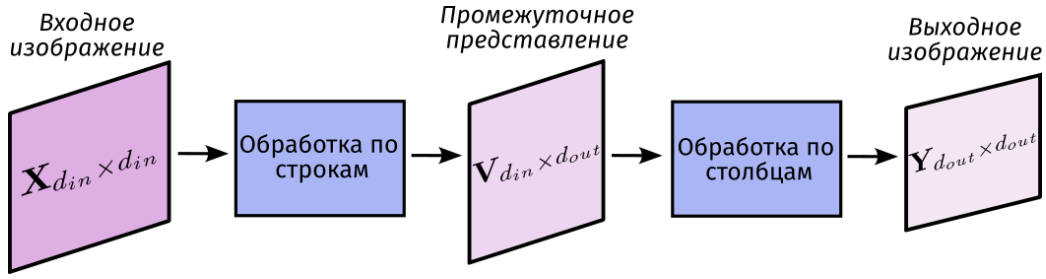


Рис. 1. Принцип работы разделимого преобразования
Fig. 1. The principle of operation of a separable transform

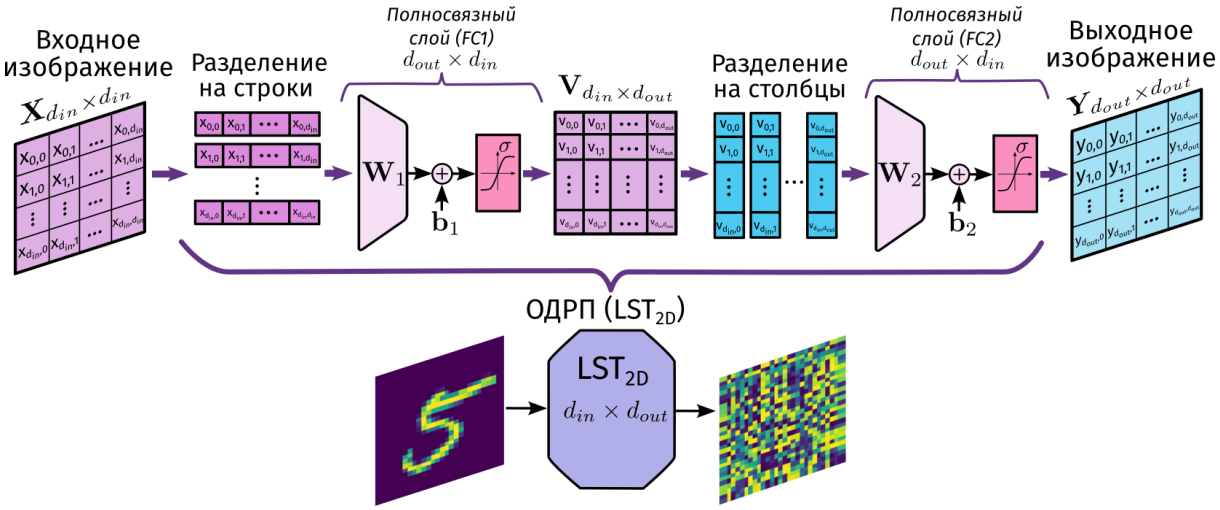


Рис. 2. Обработка изображения обучаемым двумерным разделимым преобразованием
Fig. 2. Image processing by a learnable two-dimensional separable transform

Можно заметить, что ОДРП состоит из двух этапов обработки. На первом этапе входное изображение разделяется на строки и обрабатывается полносвязным слоем FC1, к выходу которого применяется нелинейная функция активации. После этого результирующее представление обрабатывается по столбцам с использованием слоя FC2. Выход $\mathbf{Y}$ ОДРП будем называть вложением (англ. *embedding*). Размер полученного вложения определяется параметром d_{out}. Общее число настраиваемых параметров ОДРП находится по формуле

$$N_{LST} = 2 \cdot (d_{in} + 1) \cdot d_{out} . \quad (2)$$

НС LST-1 для распознавания изображений на основе ОДРП. ОДРП можно использовать в качестве базового блока при конструировании НС для распознавания изображений. В настоящей работе предлагается рассмотреть НС, состоящую из одного блока ОДРП и классифицирующего полносвязного слоя с активационной функцией softmax (рис. 3). Внутри блока ОДРП в качестве активационной функции используется гиперболический тангенс.

НС, изображенная на рис. 3, работает следующим образом: на вход подается изображение размером 28×28, которое при помощи блока ОДРП преобразуется во вложение размером $d_{out} \times d_{out}$. Слой Flatten выполняет преобразование вложения в одномерный вектор размером d_{out}^2. Далее полносвязный слой, параметрами которого являются матрица весов $\mathbf{W}_o$ размером

$d_{out}^2 \times n_{classes}$ и вектор смещений $\mathbf{b}_o$ размерности $n_{classes}$, с активационной функцией softmax выполняет классификацию изображения, вычисляя вероятности отнесения изображения к каждому из $n_{classes}$ классов. В качестве результата классификации выбирается класс, который имеет наибольшую вероятность.

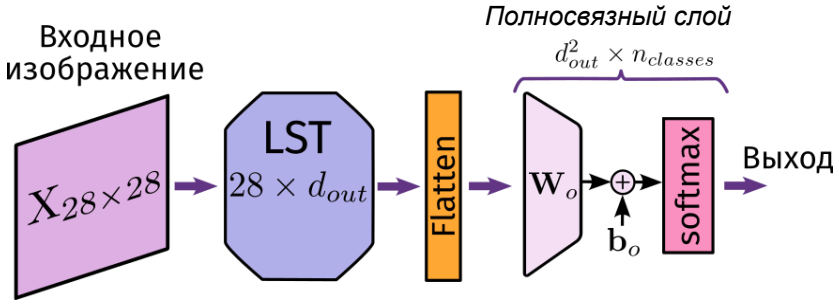


Рис. 3. Архитектура нейронной сети LST-1- d_{out}

Fig. 3. Architecture of neural network LST-1- d_{out}

Представленная модель в дальнейшем будет обозначаться как LST-1- d_{out}, где d_{out} – число, определяющее размерность получаемого внутри НС вложения, от которого зависит общее число настраиваемых параметров модели. Более строго общее число параметров модели LST-1- d_{out} определяется как сумма числа параметров ОДРП и числа параметров полносвязного выходного слоя:

$$N_{\text{LST-1}} = N_{\text{LST}} + N_{\text{FC}} = 2 \cdot (d_{in} + 1) \cdot d_{out} + n_{classes} \cdot (d_{out}^2 + 1), \quad (3)$$

где $n_{classes}$ – число распознаваемых классов.

Значения d_{in} и $n_{classes}$, как правило, определяются условиями задачи и являются фиксированными, в то время как d_{out} может определяться на этапе разработки модели. Заметим, что от параметра d_{out} зависит емкость (англ. *capacity*) модели и ее обобщающая способность. Однако, как следует из выражения (3), его увеличение вызывает квадратичный рост числа настраиваемых параметров.

На рис. 4 показано, как происходит обработка изображения в обученной сети LST-1-28.

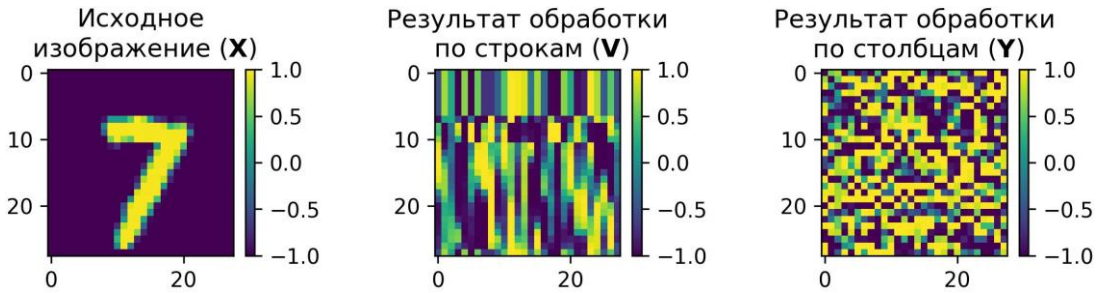


Рис. 4. Обработка изображения в нейронной сети LST-1-28

Fig. 4. Image processing in the LST-1-28 neural network

На первом этапе исходное изображение $\mathbf{X}$ обрабатывается по строкам, в результате чего образуется промежуточное представление $\mathbf{V}$. Можно заметить, что первые семь строк $\mathbf{V}$ имеют одинаковые значения. Это связано с тем, что на исходном изображении первые семь

строк также имеют одинаковые значения, равные −1. Результат обработки по столбцам $\mathbf{Y}$ (или выход ОДРП) представляет собой вложение, которое после «разворачивания» его в вектор подается на вход классифицирующего слоя. Можно отметить, что $\mathbf{Y}$ является рандомизированым представлением иходного изображения, соседние пиксели $\mathbf{Y}$ не имеют или имеют очень слабую корреляционную связь между собой.

Реализация НС LST-1 на ПЛИС. В данном разделе описывается функциональная схема IP-блока для аппаратной реализации модели LST-1 на ПЛИС. Интерфейс разработанного аппаратного модуля показан на рис. 5.

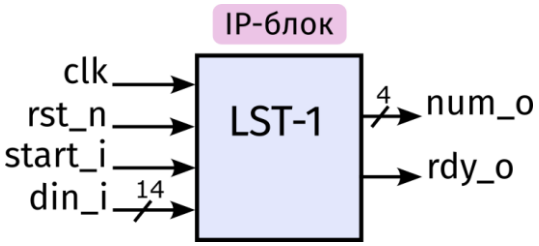


Рис. 5. Интерфейс IP-блока нейронной сети
Fig. 5. Interface of the neural network IP block

IP-блок имеет входы для управляющих сигналов сброса (`rst_n`) и запуска (`start_i`), а также информационный вход `din_i`, использующийся для подачи пикселей изображения. После подачи на вход всех пикселей изображения IP-блок автоматически переходит в режим вычисления выходной метки. По окончании вычислительного процесса на выходе `rdy_o` устанавливается значение логической единицы, а метка распознанного класса поступает на вход `num_o`.

Общий подход, применяемый в реализации данного IP-блока, заключается в переиспользовании аппаратных ресурсов ПЛИС. IP-блок имеет ОЗУ, в которое в начальный момент времени помещается исходное изображение. Затем выполняется обработка изображения по строкам и запись результата в то же ОЗУ. Далее происходит обработка полученного представления изображения по столбцам с записью результата в ОЗУ. На заключительном этапе производятся вычисления, реализующие полносвязный классифицирующий слой, данные для которого также берутся из общего ОЗУ. Используя возможности ПЛИС по реализации параллельных вычислений, предлагается параллельно вычислять элементы строк и столбцов ОДРП, для чего в структуре IP-блока применяются d_{out} вычислительных элементов (англ. *PE, processing element*). Каждый вычислительный элемент (ВЭ) состоит из MAC-ядра (англ. *Multiply and ACcumulate*) и набора ПЗУ, которые хранят параметры модели LST-1.

Ниже приведен общий алгоритм работы IP-блока, реализующего модель LST-1- d_{out} :

1. `cnt_o = 0`
2. **`if`** `(start_i == 1)` перейти к шагу 3, **`else`** перейти к шагу 2
3. `ОЗУ[cnt_o] = din_i`
4. `cnt_o = cnt_o + 1`
5. **`if`** `(cnt_o ==` $d_{out}^2 - 1$ `)` перейти к шагу 6, **`else`** перейти к шагу 2
6. **`for`** `cnt_r = 0, 1,…,` $d_{in} - 1$
7. **`for`** `i = 0, 1,…,` $d_{in} - 1$
8. `ACC[i] = b`$_1$`[i]`
9. **`end for`**
10. **`for`** `cnt_c = 0, 1,…,` $d_{in} - 1$
11. **`for`** `i = 0, 1,…,` $d_{in} - 1$
12. `ACC[i] = ACC[i] + W`$_1$`[i,cnt_c]*ОЗУ[cnt_r*` d_{in} `+ cnt_c]`

```
13.     end for
14.   end for
15.   for cnt_c = 0, 1,…, d_in −1
16.     ОЗУ[cnt_r* d_in + cnt_c] =Tanh(ACC[cnt_c])
17.   end for
18. end for
19. for cnt_c = 0, 1,…, d_in −1
20.     for i = 0, 1,…, d_in −1
21.         ACC[i] = b2[i]
22.     end for
23.     for cnt_r = 0, 1,…, d_in −1
24.         for i = 0, 1,…, d_in −1
25.             ACC[i] = ACC[i] + W2[i,cnt_r] * ОЗУ[cnt_c* d_in + cnt_r]
26.         end for
27.     end for
28.     for cnt_r = 0, 1, …, d_in −1
29.         ОЗУ[cnt_c* d_in + cnt_r] = Tanh(ACC[cnt_r])
30.     end for
31. end for
32. for j = 0, 1,…, n_classes −1
33.    ACC[i] = b[i]
34. end for
35. for cnt_o = 0, 1,…, d_out −1
36.     for j = 0, 1,…, n_classes −1
37.         ACC[j] = ACC[j] + W[j,cnt_o] * ОЗУ[cnt_o]
38.     end for
39. end for
40. num_o = argmax(ACC[0,1,…, n_classes −1])
41. rdy_o = 1
```

В представленном алгоритме шаги 1–5 нужны для начальной записи изображения в ОЗУ. Шаги 6–18 описывают обработку изображения по строкам, а шаги 19–31 – обработку по столбцам. Наконец, шаги 32–41 описывают вычисления, связанные с классифицирующим слоем. Общая структура IP-блока, реализующего данный алгоритм вычисления для случая $d_{out} = 28,$ изображена на рис. 6.

Для корректной выборки данных в системе используются три счетчика: два из них отвечают за определение строки и столбца соответствующих данных, а третий служит для сквозной адресации ОЗУ. Вычисление функции гиперболического тангенса (tanh) осуществляется последовательно, что позволяет за 56 системных тактов выполнить обработку одной строки изображения (28 тактов тратится на расчет суммы произведений и 28 тактов – на запись результата в ОЗУ). Для определения выходного значения применяется упрощенная по сравнению с softmax функция argmax, которая не использует вычисление экспоненциальных функций, что способствует уменьшению вычислительной сложности.

Все вычисления, относящиеся к ОДРП и к выходному полносвязному слою, осуществляются с использованием $d_{out} = 28$ ВЭ. При реализации одного физического вычислительного слоя возникает необходимость разделения ВЭ на два типа (рис. 7): rco – row-column-output и rc – row-column.

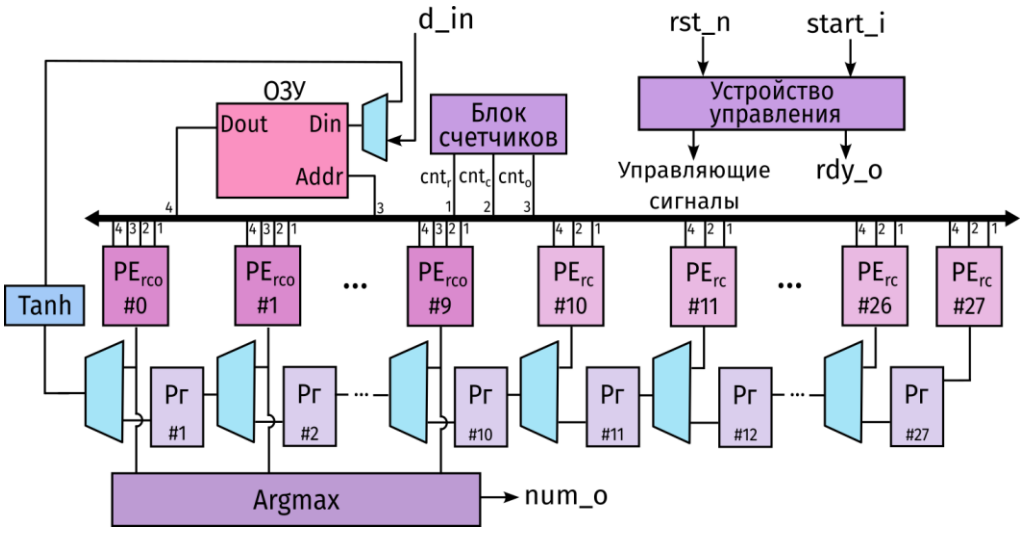


Рис. 6. Структура IP-блока нейронной сети LST-1-28

Fig. 6. The structure of the LST-1-28 neural network IP block

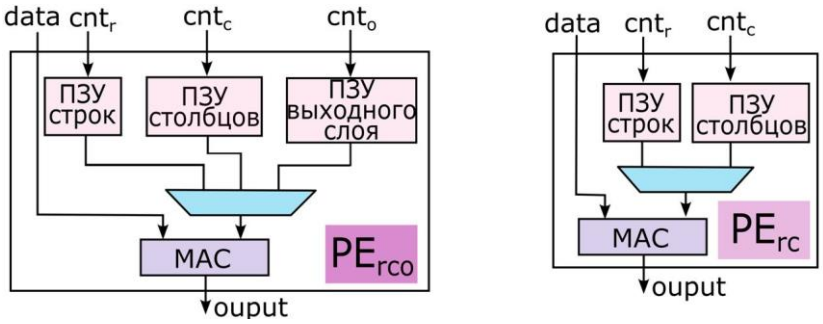


Рис. 7. Архитектура вычислительных блоков нейронной сети

Fig. 7. Architecture of neural network processing elements

В IP-блоке LST-1 используются 10 ВЭ типа rco, которые многократно применяются как для вычислений ОДРП, так и для формирования значений выходного полносвязного слоя НС. Для поддержки двойной функциональности ВЭ требуются хранилище для коэффициентов выходного слоя и дополнительный адресный вход, обеспечивающий управление доступом к памяти. Оставшиеся 18 ВЭ типа rc предназначены исключительно для обработки строк и столбцов в ОДРП. Переключение режимов вычислений осуществляется устройством управления.

Блок счетчиков (рис. 6) служит для генерации адресов доступа как к памяти весов, так и к ОЗУ, где хранятся промежуточные результаты на всех этапах вычислений.

Слой softmax (см. рис. 3) отвечает за формирование распределения вероятностей по 10 выходным классам (цифры от 0 до 9). Однако нет необходимости использовать функцию активации softmax при аппаратной реализации НС. Вместо этого можно использовать функцию argmax для сравнения всех 10 выходных значений классифицирующего слоя и выбора класса с наивысшим значением (англ. *score*). Кроме того, аппаратная реализация функции argmax задействует значительно меньше ресурсов ПЛИС по сравнению с softmax.

Устройство управления реализовано как конечный автомат с комбинационной логикой для генерации следующего состояния и соответствующих управляющих сигналов для других модулей.

Нелинейная функция активации tanh аппроксимируется с помощью аппаратно-ориентированной кусочной функции, поскольку прямая реализация потребовала бы бóльших вычислительных затрат. Аппроксимация выполняется по формуле [2]

$$\tanh(x) \approx F(x) = \begin{cases} \text{sign}(x), \ |x| > 2, \\ (1+\frac{x}{4}) \cdot x, \ -2 < x < 0, \\ (1-\frac{x}{4}) \cdot x, \ 0 < x < 2. \end{cases} \quad (4)$$

На рис. 8 показаны исходная функция гиперболического тангенса и ее аппроксимация.

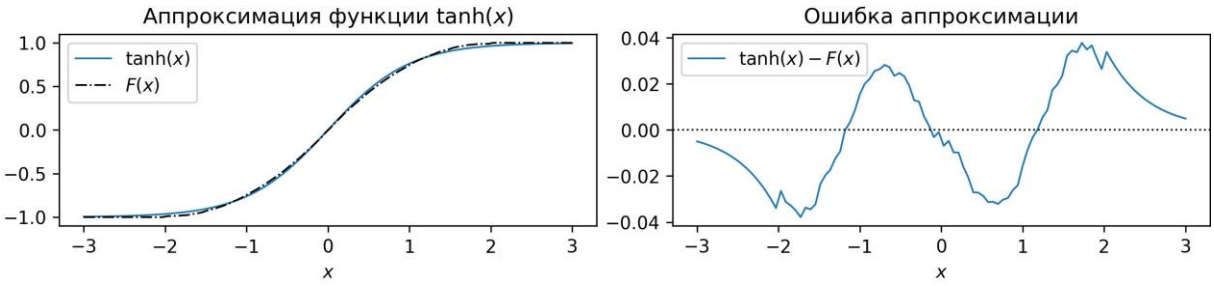


Рис. 8. Сравнение функции tanh с ее аппроксимацией

Fig. 8. Comparison of the tanh function with its approximation

Исходя из рис. 8 можно сделать вывод, что аппроксимация позволяет корректно отразить поведение функции активации tanh. Это говорит о релевантности ее использования с целью снижения аппаратных затрат при реализации на ПЛИС.

Процесс обучения НС LST-1. Обучение НС LST-1 осуществлялось с использованием базы изображений рукописных цифр MNIST. База MNIST состоит из 70 тыс. изображений размером 28×28 пикселей в градациях серого и разбита на две части – тренировочный набор (60 тыс. изображений) и тестовый (10 тыс. изображений). На этапе обучения из тренировочного набора случайным образом выделялась 1 тыс. изображений для использования в качестве валидационного набора, оставшиеся 59 тыс. применялись для обучения модели. Для обучения модели использовались язык Python и библиотека PyTorch.

Перед подачей в НС выполнялась предварительная нормализация изображений. Изначально пиксели изображений представлены числами в диапазоне от 0 до 255. Для упрощения процесса обучения выполняется нормализация данных таким образом, чтобы значение каждого пикселя было в диапазоне от −1 до 1, среднее значение равнялось нулю, а среднеквадратическое отклонение составляло 0,5. Нормализация изображений – распространенный практический прием [9], повышающий устойчивость градиентного спуска и ускоряющий сходимость модели НС. В качестве функции потерь использовался отрицательный логарифм функции правдоподобия (англ. *negative log-likelihood loss*, *NLLLoss*).

Для оптимизации параметров НС LST-1 применялся метод Adam (англ. *ADAptive Moment estimation*) – один из самых эффективных алгоритмов градиентного спуска с адаптивным шагом. Основной параметр любого метода стохастического градиентного спуска – скорость обучения η, которая, по сути, является гиперпараметром, влияющим на результат обучения. Слишком большое значение η может привести к тому, что в процессе минимизации функции потерь модель не сможет стабилизироваться в точке минимума (глобального или хорошего локального). С другой стороны, слишком малое значение η, как правило, приводит к тому, что модель «застревает» в точке непроизводительного локального минимума. Чтобы избежать двух

описанных сценариев, в данной работе использовался планировщик скорости обучения – метод, который плавно изменяет η в процессе обучения. В частности, скорость обучения определялась функцией циклического косинусного отжига [10]:

$$\eta(t) = \eta_{\min} + \frac{1}{2}(\eta_{\max} - \eta_{\min})\left(1 + \cos\left(\frac{\mathrm{mod}(t, T_0)}{T_0}\pi\right)\right), \quad (5)$$

где t – номер текущей эпохи; $\eta_{\min}$ – минимальная скорость обучения; $\eta_{\max}$ – максимальная скорость; T_0 – количество эпох, в течение которых происходит спад косинуса, прежде чем скорость сбросится.

Таким образом, после каждой эпохи обучения параметр скорости обучения перерассчитывался по формуле (5) и с помощью валидационного набора для текущей модели рассчитывалось значение функции потерь. В качестве итоговой выбиралась модель, которая за все время обучения имела наименьшее значение функции потерь на валидационном наборе.

Рассмотренные в настоящей работе модели обучались в течение 300 эпох, длительность одного цикла отжига (параметр T_0 в выражении (5)) выбиралась равной 100, начальная скорость обучения $\eta_{\max} = 0{,}001$, а минимальная скорость обучения $\eta_{\min} = 5 \cdot 10^{-6}$. На рис. 9 показан график изменения скорости обучения, использовавшийся при обучении моделей.

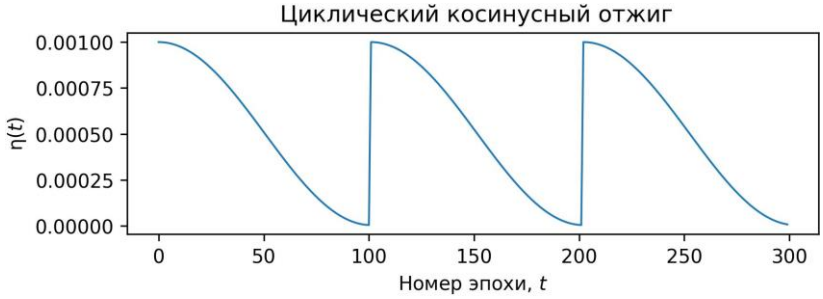


Рис. 9. График изменения скорости обучения с использованием метода косинусного циклического отжига
Fig. 9. Graph of the change in the learning rate using the cosine annealing with warm restart

Для регуляризации рассмотренных в работе моделей применялся метод дропаут (англ. *dropout*) [11], который заключается в случайном отключении заданной доли p_{drop} нейронов в полносвязном слое в процессе обучения. Это заставляет модель учиться более устойчивым представлениям и давать верные ответы даже при наличии неполных данных на входе. В полносвязных слоях ОДРП модели LST-1, отвечающих за обработку строк и столбцов, значение p_{drop} выбиралось равным $0{,}1$.

Экспериментальная часть работы включает два основных этапа: первый – исследование производительности модели LST-1 при использовании внутренних вложений различной размерности, второй – валидация и тестирование аппаратной реализации модели LST-1 на базе ПЛИС.

Результаты обучения НС LST-1. В работе выполнен анализ производительности модели LST-1-d_{out} для различных значений параметра d_{out}. Как говорилось ранее, в предлагаемой модели LST-1 параметр d_{out} отвечает за размер внутреннего представления входного изображения, а также влияет на общее число параметров модели. Ниже приведены результаты обучения 11 различных моделей LST-1 с разными значениями параметра d_{out}. Для каждой модели проведены 10 независимых экспериментов обучения с различной начальной инициали-

зацией весов. Статистические показатели (среднее значение и стандартное отклонение) точности, рассчитанные по результатам этих экспериментов, представлены в табл. 1. Для регуляризации приведенных в табл. 1 моделей к классифицирующему слою в процессе обучения также применялся метод дропаут. Для моделей со значением параметра $d_{out} = 2,4,...,16$ для последнего слоя устанавливалось значение $p_{drop} = 0,1$. Для моделей со значениями параметра $d_{out} = 20, 24, 28, 32, 36$ и 40 использовались значения $p_{drop} = 0,15; 0,15; 0,18; 0,20$ и 0,22 соответственно.

Таблица 1
Число параметров моделей LST-1- d_{out} и их точность на тестовой выборке MNIST
Table 1
The number of parameters of the LST-1- d_{out} models and their accuracy on the MNIST test set

Модель *Model*	Число параметров *Number of parameters*	Точность, % *Accuracy, %*
LST-1-2	166	75,13 ± 2,28
LST-1-4	402	92,93 ± 0,66
LST-1-8	1114	96,28 ± 0,19
LST-1-12	2146	97,14 ± 0,17
LST-1-16	3498	97,27 ± 0,13
LST-1-20	5170	97,61 ± 0,06
LST-1-24	7162	97,83 ± 0,14
LST-1-28	9474	98,03 ± 0,14
LST-1-32	12 106	98,17 ± 0,18
LST-1-36	15 058	98,23 ± 0,13
LST-1-40	18 330	98,32 ± 0,08

Для примера на рис. 10 показаны кривые обучения модели LST-1-20 на тестовом и валидационном наборах. Кривые показывают, что модель LST-1 имеет стабильную сходимость, хотя процесс обучения значительно замедляется после 50-й эпохи. Следует отметить, что после 100-й и 200-й эпох видны возмущения функции потерь как на тренировочном, так и на валидационном наборах, которые объясняются применением метода косинусного отжига для управления процессом обучения модели. Если обратиться к графику на рис. 9, то можно заметить, что именно на 100-й и 200-й эпохах происходят скачкообразные переходы к начальной скорости обучения, которые и вызывают возмущения.

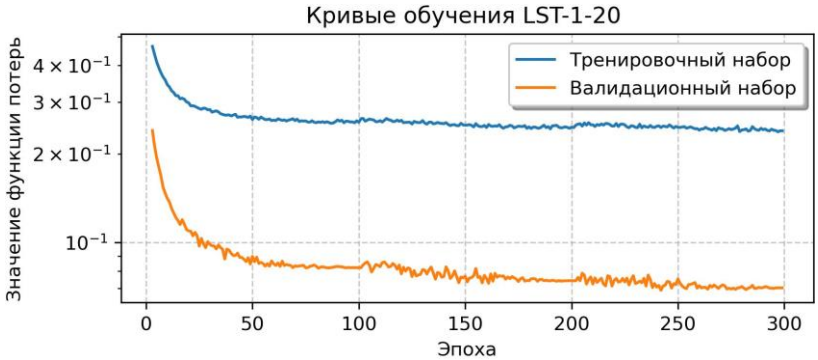


Рис. 10. Функция потерь на тренировочном и валидационном наборах для модели LST-1-20
Fig. 10. Loss function on the training and validation sets for the LST-1-20 model

Разрыв между значениями функции потерь на тренировочном и валидационном наборах обусловлен применением дропаут-регуляризации в процессе обучения. Данный метод, реализуемый через стохастическое зануление активаций скрытых нейронов, вносит дополнительный шум в данные, что приводит к смещению оценки потерь в сторону увеличения для тренировочного набора относительно валидационного.

Для полноты оценки и выявления преимуществ модели LST-1 проведен сравнительный анализ с архитектурами НС, которые чаще всего используются для аппаратной реализации изображений рукописных цифр на базе ПЛИС.

Чаще всего в работах, посвященных реализации НС на ПЛИС для распознавания изображения, в качестве базовой архитектуры рассматривается многослойный перцептрон (МСП) (англ. *MLP, multilayer perceptrone*), состоящий из каскада полносвязных слоев [2, 4, 6, 7, 12, 13]. Такой выбор можно объяснить тем, что регулярная структура полносвязных слоев МСП хорошо отображается на архитектуру ПЛИС. Вычислительное ядро таких слоев – операции умножения с накоплением – эффективно реализуется с использованием встроенных DSP-блоков, имеющихся во многих современных ПЛИС. Наконец, в отличие от сверточных НС, МСП имеет простые и предсказуемые паттерны доступа к памяти данных и весовых коэффициентов, что существенно упрощает логику управления памятью.

Для обозначения топологии конкретного МСП используется специальная нотация. Например, МСП 784-13-10 обозначает, что НС имеет 784 нейрона на входном слое (соответствует числу пикселей изображения), 13 нейронов скрытого слоя и 10 нейронов выходного слоя, что соответствует количеству распознаваемых классов цифр (0-9).

В табл. 2 приведены результаты сравнения различных МСП с моделью LST-1 по числу параметров и достигаемой точности. Для моделей LST-1-8 и LST-1-28 указаны максимальные значения точности, полученные в серии из 10 независимых экспериментов обучения с различной начальной инициализацией весов (см. табл. 1).

Таблица 2
Сравнение модели LST-1 с различными архитектурами МСП

Table 2

Comparison of the LST-1 model with various MLP architectures

Авторы *Authors*	Модель *Model*	Число параметров *Number of parameters*	Точность, % *Accuracy, %*
[данная работа]	LST-1-8 (предлагаемая)	1114	96,53
Kwon, et al. [6]	МСП 196-14-10	2908	94,03
[данная работа]	LST-1-28 (предлагаемая)	9474	98,37
Westby, et al. [7]	МСП 784-12-10	9550	93,25
Huynh [13]	МСП 784-40-40-40-10	34 960	97,20
Huynh [13]	МСП 784-126-126-10	115 920	98,16
Medus, et al. [2]	МСП 784-600-600-10	891 610	98,63
Liang, et al. [12]	МСП 784-2048-2048-2048-10	10 100 000	98,32

Среди компактных архитектур для аппаратной реализации распознавания рукописных цифр выделяется МСП 196-14-10 [6]. Особенность данной архитектуры заключается в предварительном снижении размерности входного изображения 28×28 пикселей с помощью слоя субдискретизации (англ. *max-pooling*), который формирует уменьшенное изображение размером 14×14. Данное представление затем преобразуется в одномерный вектор длиной 196, который подается на вход двухслойной сети. Такой трюк с понижением размерности на входе позволяет значительно сократить число обучаемых параметров (до 2,9 тыс.), сохранив при этом высокую точность 94 %. Для сравнения однослойный перцептрон, обрабатывающий исходное изображение без предварительного понижения размерности (784 входных пикселя) и формирующий 10 выходных вероятностей, содержит 7,9 тыс. параметров и достигает точности 92,4 % [14]. Однако представленная в настоящем исследовании модель LST-1-8 превосходит

архитектуру [6] по двум основным показателям: точности (превышение на 2,5 %) и компактности (количество параметров меньше в 2,5 раза). МСП, предложенные в работе [13], позволяют достичь точности от 97,2 до 98,16 % с помощью небольших скрытых полносвязных слоев. В работе [12] применяется подход с увеличением числа параметров скрытых слоев. Это дает возможность добиться высокой точности (98,32 %) и требует более 10 млн параметров, что превышает все рассмотренные архитектуры.

Среди компактных архитектур, обрабатывающих все 784 пикселя исходного изображения, следует выделить МСП 784-12-10 [7], который имеет 9550 параметров и достигает точности 93,25 %. Однако предлагаемая в настоящей работе модель LST-1-28 имеет практически такое же число параметров и точность, превосходящую на 5,12 %.

Предлагаемая модель LST-1-28 показывает точность, сопоставимую с более сложными архитектурами МСП 784-126-126-10 [13] и МСП 784-2048-2048-2048-10 [12], которые, однако, имеют одна в 12, а вторая в 1060 раз больше параметров, чем LST-1-28. МСП 784-600-600-10, представленный в работе [2], имеет точность, на 0,26 % превышающую точность модели LST-1-28, но при этом число его параметров в 94 раза больше.

Квантование параметров модели LST-1 и реализация с фиксированной запятой. При аппаратной реализации НС важно предотвратить переполнение разрядной сетки. Поскольку модель LST-1, по сути, состоит из трех полносвязных слоев, достаточно рассмотреть вопрос определения разрядности данных на примере одного полносвязного слоя. Определение разрядности выполнялось по принципу «наихудшего случая». Известно, что на входе каждого слоя модели LST-1 данные находятся в диапазоне $[-1, 1]$. Наибольшие значения на выходе полносвязного слоя будут получаться в том случае, если на вход поступит вектор, каждая компонента которого по модулю будет равна единице, а знаки будут совпадать со знаками строки матрицы весов $\mathbf{W}$. Более формально наибольшие возможные значения на выходе полносвязного слоя (до функции активации) можно оценить по формуле

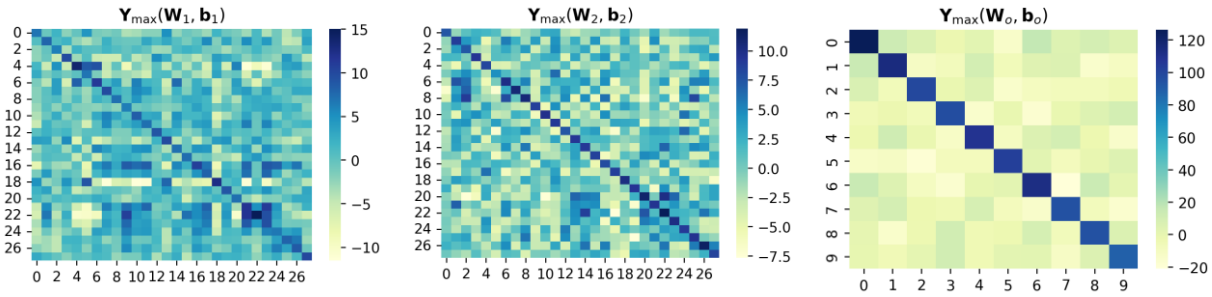
$$\mathbf{Y}_{\max}(\mathbf{W},\mathbf{b}) = \mathbf{W} \cdot \operatorname{sign}(\mathbf{W}^T) + \mathbf{b}. \quad (6)$$

На рис. 11 приведены значения $\mathbf{Y}_{\max}$ для трех полносвязных слоев обученной сети LST-1-28.

На выходе слоя, выполняющего обработку по строкам, максимально возможное значение равняется 15,02. Аналогичное значение для слоя, выполняющего обработку по столбцам, равно 11,9. Максимально возможное значение для выходного слоя равняется 126,3. Таким образом, можно сделать вывод, что для представления целой части данных потребуется восемь разрядов (один разряд на знак и оставшиеся семь для того, чтобы представить максимально возможное число – 126). Число дробных разрядов для представления данных выбиралось равным семи. Следует заметить, что восемь разрядов для представления целой части данных является завышенной оценкой. Дополнительные проверки путем симуляции модели LST-1 в арифметике с фиксированной запятой (ФЗ) показали, что уже при использовании семи разрядов для представления целой части переполнения разрядной сетки не наступает. Таким образом, для внутреннего представления данных в IP-блоке использовался формат Q7.7. При переводе значений параметров модели из формата с плавающей запятой (ПЗ) в формат с ФЗ использовался метод округления к ближайшему меньшему числу, который позволяет минимизировать ошибку округления.

Для верификации IP-блока LST-1 была разработана его эталонная модель на языке Python с использованием библиотеки fixedpoint. Данная библиотека дает возможность моделировать вычислительные процессы в арифметике с ФЗ. Модель LST-1 с ФЗ позволяет получать доступ ко всем промежуточным результатам вычислений, что существенно сократило время отладки IP-блока на этапе RTL-проектирования. Кроме того, проведенное моделирование выявило, что основной причиной расхождений между исходной версией LST-1 с ПЗ и реализацией с ФЗ является использование аппроксимации функции гиперболического тангенса.

На рис. 12 представлены результаты работы моделей с ФЗ и ПЗ на этапе вычисления выхода полносвязного слоя, выполняющего обработку строк изображения. Можно видеть, что до применения активационной функции обе модели дают очень близкий результат. Однако после применения активационной функции расхождение между моделями существенно увеличивается.



a) *b)* *c)*

Рис. 11. Оценка максимального значения на выходе полносвязных слоев модели LST-1-28:
a) слой обработки строк; *b)* слой обработки столбцов; *c)* классификационный слой

Fig. 11. Estimation of the maximum value at the output of fully connected layers of LST-1-28 model: a) row processing layer; b) column processing layer; c) classification layer

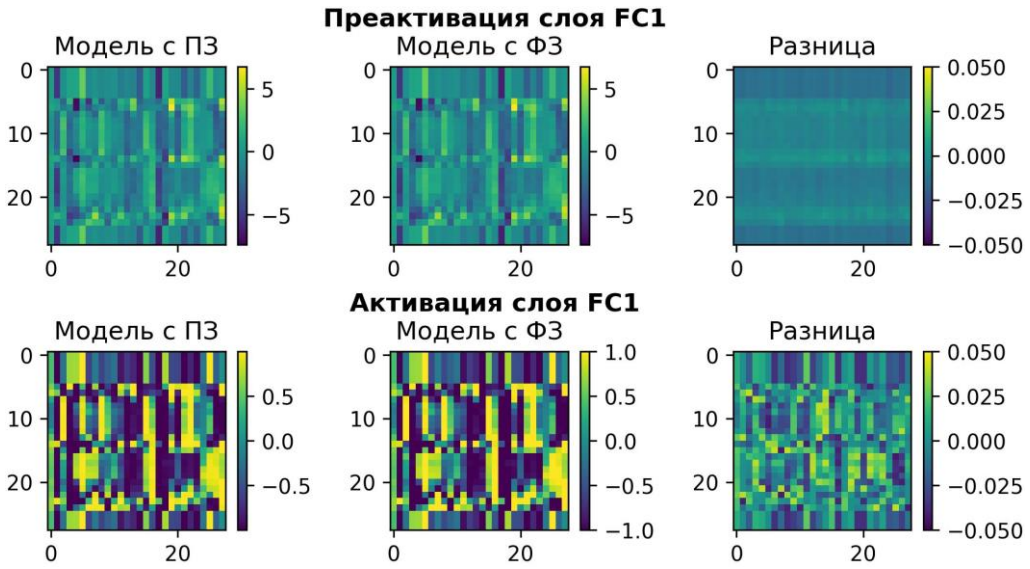


Рис. 12. Сравнение преактивации и активации в слоях FC1 моделей LST-1-28
с плавающей и фиксированной запятой

Fig. 12. Comparison of preactivation and activation in FC1 layers of floating-point and fixed-point models LST-1-28

Таким образом, можно сделать вывод, что квантование параметров модели вносит меньший вклад в расхождение моделей с ФЗ и ПЗ, чем использование аппроксимированной функции активации.

Тестирование и анализ аппаратной реализации модели LST-1. Для реализации модели LST-1 была выбрана отладочная плата Zybo на базе ПЛИС фирмы Xilinx Zynq-7000. Платформа Zynq объединяет процессор ARM с программируемой логикой FPGA, обеспечивая гибкое и эффективное аппаратно-программное решение. Для упрощения разработки и тестирования на данной платформе используется дистрибутив Linux-PYNQ, который запускался на процессоре ARM. PYNQ позволяет взаимодействовать с аппаратными блоками ПЛИС, реализованными в виде IP-ядер, с помощью ноутбука Jupyter, что делает процесс разработки более удобным.

IP-блок НС LST-1 управляется через регистровый файл, подключенный по uP-интерфейсу (данный интерфейс разработан фирмой Analog Devices). При разработке IP-блоков для платформ Xilinx стандартным является использование AXI-интерфейса. Поэтому для подключения к процессорной системе Zynq внутри IP-блока LST-1 применяется преобразователь интерфейсов uP-AXI4-lite. Общая архитектура системы, использованной для тестирования аппаратной реализации НС LST-1, показана на рис. 13.

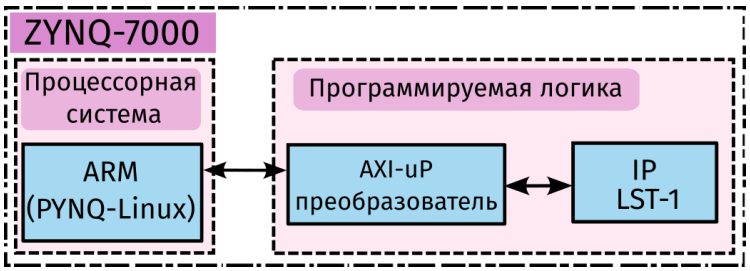
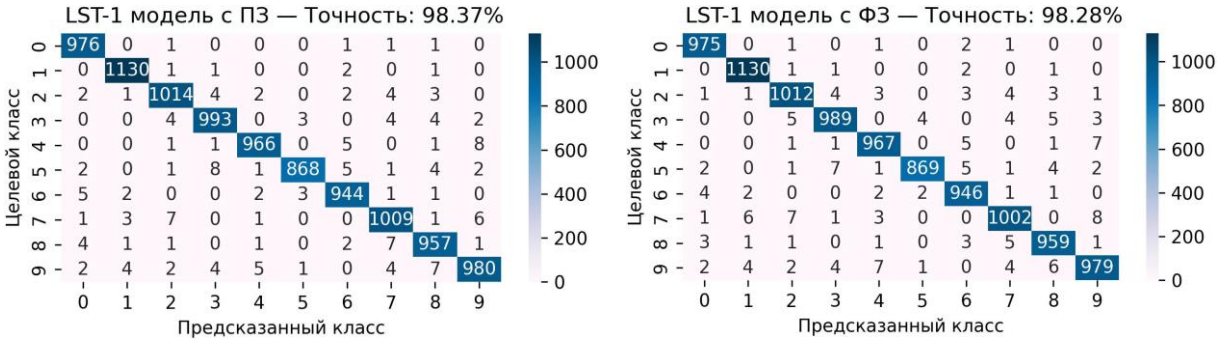
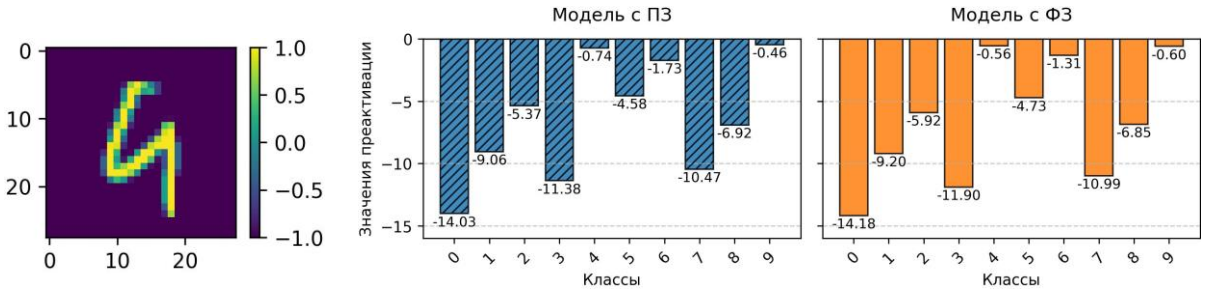


Рис. 13. Прототипирование нейронной сети LST-1 на ПЛИС
Fig. 13. Prototyping the LST-1 neural network on FPGA



a) *b)*

Рис. 14. Матрица ошибок нейронной сети LST-1-28: *a)* модель с плавающей запятой; *b)* модель с фиксированной запятой после квантования параметров
Fig. 14. Confusion matrix of the LST-1-28 : a) floating-point model; b) fixed-point model with quantized parameters



a) *b)*

Рис. 15. Анализ работы модели LST-1-28: *a)* изображение цифры девять на входе нейронной сети; *b)* сравнение значений преактивации до применения функции softmax/argmax
Fig. 15. Analysis of the LST-1-28 model: a) image of a digit nine at the input of a neural network; b) comparison of preactivations before applying the softmax/argmax function

Для тестирования аппаратной реализации модели LST-1 на нее подавались 10 тыс. тестовых изображений базы MNIST. Для анализа полученных результатов выполнялось построение матрицы ошибок, которая количественно отображает распределение предсказаний модели относительно истинных классов объектов.

На рис. 14 показаны матрицы ошибок, полученные для модели с ПЗ, а также при реализации LST-1 на ПЛИС (данная матрица также совпадают с матрицей, полученной с помощью модели с ФЗ на основе Python). Общая точность модели LST-1 с весами, квантованными в формате Q7.7, составляет 98,28 %, что всего на 0,09 % отличается от точности модели с ПЗ. Следует заметить, что для некоторых классов точность предсказаний даже выросла. Так, исходная модель с ПЗ распознала правильно 944 изображения цифры шесть, в то время как модель с ФЗ правильно распознала 946 изображений цифры шесть.

Для того чтобы прояснить характер ошибок, допускаемых моделью с ФЗ, был проанализирован случай классификации тестового изображения 4823 базы MNIST (рис. 15, *a*), которое было правильно распознано моделью с ПЗ и ошибочно – моделью с ФЗ.

На рис. 15, *b* представлены значения преактивации выходного слоя сравниваемых моделей. В реализации с ПЗ максимальное значение (–0,46) соответствует верному классу 9 (истинная метка MNIST), однако близкое значение на классе 4 (–0,74) свидетельствует о высокой схожести цифр девять и четыре. В модели с ФЗ эффекты квантования параметров и аппроксимации гиперболического тангенса вызывают изменение выходных значений: минимальное значение (–0,56) соответствует ошибочному классу 4, тогда как правильному классу 9 соответствует следующее по величине значение (–0,60). Данный пример, с одной стороны, поясняет характер ошибок, возникающих в модели с ФЗ, а с другой – также позволяет понять причину, по которой в некоторых случаях модель с ФЗ (например, для класса 6) дала больше правильных прогнозов, чем модель с ПЗ. Имеется в виду, что ошибки квантования параметров модели вероятностно скорректировали значения выходного слоя таким образом, что максимальные значения преактивации переместились на истинный класс.

Синтез IP-блока LST-1 в САПР Vivado 2024.2 показал, что для реализации НС требуется 1288 LUT-блоков и 1071 триггер. Общие аппаратные затраты представлены в табл. 3.

Таблица 3
Аппаратные затраты на реализацию IP-блока LST-1

Table 3
Hardware resources for the LST-1 IP block implementation

Блок *Resource*	Использовано *Utilization*	Доступно *Available*	Соотношение, % *Utilization, %*
LUT	1288	17 600	7,32
LUTRAM	54	6000	0,90
FF	1071	35 200	3,04
BRAM	33,5	60	55,83
DSP	57	80	71,25

Разработанная архитектура НС требует использования лишь 55,83 % блочной памяти (BRAM) для хранения всех весовых коэффициентов и смещений, что указывает на пониженные требования к памяти модели LST-1. DSP-блоки, представленные в табл. 3, используются для реализации MAC-ядер внутри вычислительных элементов PE_{rc}/PE_{rco}. Максимальная тактовая частота IP-ядра 90 МГц. На обработку одного изображения IP-ядру требуется 3921 такт. Для сравнения – аппаратная реализация НС с архитектурой VGG16, представленная в работе [15] и имеющая точность на наборе MNIST 98,34 %, требует 108 тыс. тактов на обработку одного изображения.

Суммарная потребляемая мощность IP-ядра на чипе составляет 1,53 Вт. Температура зафиксирована на уровне 42,7 °C, что находится в допустимых пределах для стабильной работы устройства. Температурный зазор составляет 42,3 °C, что указывает на запас по тепловой стабильности и безопасное функционирование системы.

Заключение. В работе предложена концепция ОДРП для построения эффективных и компактных архитектур НС. Разработано семейство компактных моделей LST-1 с настраиваемой размерностью векторных представлений изображения (LST-1-8, LST-1-28 и др.). Достигнута высокая точность распознавания на наборе данных MNIST (98,37 % для LST-1-28 при 9,5 тыс. параметров и 96,53 % для LST-1-8 при 1,1 тыс. параметров), что дает более чем двухкратное превосходство по числу параметров относительно аналогов. Алгоритм вычисления модели LST-1 имеет регулярную структуру, что позволило получить ее эффективную реализацию на ПЛИС. Перспективным направлением дальнейших исследований является интеграция ОДРП в архитектуры НС с остаточными связями (англ. *residual networks*, *ResNet*), а также в архитектуры сверточных НС. В качестве еще одного направления исследований можно предложить возможность построения трехмерного обучаемого преобразования для обработки мультиспектральных изображений.

Вклад авторов. *Е. А. Кривальцевич* участвовал в разработке ОДРП, подготовке численных экспериментов, разработал и протестировал аппаратную реализацию модели LST-1, подготовил первоначальный вариант статьи; *М. И. Вашкевич* предложил идею ОДРП, определил задачи исследования, разработал модель LST-1 с фиксированной запятой, корректировал текст статьи, принимал участие в подготовке графического материала, а также в интерпретации результатов экспериментов.

Список использованных источников

1. Park, J. FPGA based implementation of deep neural networks using on-chip memory only / J. Park, W. Sung // IEEE Intern. Conf. on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, China, 20–25 March 2016. – Shanghai, 2016. – P. 1011–1015.

2. A novel systolic parallel hardware architecture for the FPGA acceleration of feedforward neural networks / L. D. Medus, T. Iakymchuk, J. V. Frances-Villora [et al.] // IEEE Access. – 2019. – Vol. 7. – P. 76084–76103.

3. Han, S. Deep compression: compressing deep neural networks with pruning, trained quantization and Huffman coding / S. Han, H. Mao, W. J. Dally // Intern. Conf. on Learning Representations (ICLR), San Juan, Puerto Rico, 2–4 May 2016. – San Juan, 2016. – P. 1–14.

4. Samragh, M. Customizing neural networks for efficient FPGA implementation / M. Samragh, M. Ghasemzadeh, F. Koushanfar // Annual Intern. Symp. on Field-Programmable Custom Computing Machines (FCCM), Napa, CA, USA, 30 Apr. – 2 May 2017. – Napa, 2017. – P. 85–92.

5. Usatyuk, V. Boosting DNN efficiency: replacing FC layers with graph embeddings for hardware acceleration / V. Usatyuk, S. Egorov // Intern. Conf. on Digital Signal Processing and its Applications (DSPA), Moscow, Russia, 26–28 March 2025. – Moscow, 2025. – P. 1–6.

6. Kwon, J. Design of a low-area digit recognition accelerator using MNIST database / J. Kwon, S. Kim // JOIV: International Journal on Informatics Visualization. – 2022. – Vol. 6, no. 1. – P. 53–59.

7. FPGA acceleration on a multilayer perceptron neural network for digit recognition / I. Westby, X. Yang, T. Liu, H. Xu // The Journal of Supercomputing. – 2021. – Vol. 77, no. 12. – P. 14356–14373.

8. Vashkevich, M. Compact and efficient neural networks for image recognition based on learned 2D separable transform / M. Vashkevich, E. Krivalcevich // Intern. Conf. on Digital Signal Processing and its Applications (DSPA), Moscow, Russia, 26–28 March 2025. – Moscow, 2025. – P. 1–5.

9. Старовойтов, В. В. Нормализация данных в машинном обучении / В. В. Старовойтов, Ю. И. Голуб // Информатика. – 2021. – Т. 18, № 3. – С. 83–96.

10. Loshchilov, I. SGDR: Stochastic Gradient Descent with Warm Restarts / I. Loshchilov, F. Hutter. – 2016. – URL: https://arxiv.org/abs/1608.03983 (date of access: 01.08.2025).

11. Srivastava, N. Dropout: a simple way to prevent neural networks from overfitting / N. Srivastava, G. Hinton, A. Krizhevsky // The Journal of Machine Learning Research. – 2014. – Vol. 15, no. 1. – P. 1929–1958.

12. FP-BNN: Binarized neural network on FPGA / S. Liang, S. Yin, L. Liu [et al.] // Neurocomputing. – 2018. – Vol. 275. – P. 1072–1086.

13. Huynh, T. V. Deep neural network accelerator based on FPGA / T. V. Huynh // NAFOSTED Conf. on Information and Computer Science, Hanoi, Vietnam, 24–25 Nov. 2017. – Hanoi, 2017. – P. 254–257.

14. Кривальцевич, Е. А. Исследование аппаратной реализации нейронной сети прямого распространения для распознавания рукописных цифр на базе FPGA / Е. А. Кривальцевич, М. И. Вашкевич // Доклады БГУИР. – 2025. – Вып. 23, № 2. – С. 101–108.

15. Аппаратная реализация сверточной нейронной сети в ПЛИС на базе вычислений с фиксированной точкой / Р. А. Соловьев, А. Г. Кустов, В. С. Рухлов [и др.] // Известия Южного федерального университета. Технические науки. – 2017. – Т. 192, № 7. – С. 186–197.

References

1. Park J., Sung W. FPGA based implementation of deep neural networks using on-chip memory only. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, China, 20–25 March 2016.* Shanghai, 2016, pp. 1011–1015.

2. Medus L. D., Iakymchuk T., Frances-Villora J. V., Bataller-Mompean M., Rosado-Munoz A. A novel systolic parallel hardware architecture for the FPGA acceleration of feedforward neural networks. *IEEE Access*, 2019, vol. 7, pp. 76084–76103.

3. Han S., Mao H., Dally W. J. Deep compression: compressing deep neural networks with pruning, trained quantization and Huffman coding. *International Conference on Learning Representations (ICLR), San Juan, Puerto Rico, 2–4 May 2016.* San Juan, 2016, pp. 1–14.

4. Samragh M., Ghasemzadeh M., Koushanfar F. Customizing neural networks for efficient FPGA implementation. *Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), Napa, CA, USA, 30 April – 2 May 2017*. Napa, 2017, pp. 85–92.

5. Usatyuk V., Egorov S. Boosting DNN efficiency: replacing FC layers with graph embeddings for hardware acceleration. *International Conference on Digital Signal Processing and its Applications (DSPA), Moscow, Russia, 26–28 March 2025.* Moscow, 2025, pp. 1–6.

6. Kwon J., Kim S. Design of a low-area digit recognition accelerator using MNIST database. *JOIV: International Journal on Informatics Visualization*, 2022, vol. 6, no. 1, pp. 53–59.

7. Westby I., Yang X., Liu T., Xu H. FPGA acceleration on a multilayer perceptron neural network for digit recognition. *The Journal of Supercomputing*, 2021, vol. 77, no. 12, pp. 14356–14373.

8. Vashkevich M., Krivalcevich E. Compact and efficient neural networks for image recognition based on learned 2D separable transform. *International Conference on Digital Signal Processing and its Applications (DSPA), Moscow, Russia, 26–28 March 2025.* Moscow, 2025, pp. 1–5.

9. Starovoitov V. V., Golub Yu. I. *Data normalization in machine learning*. Informatika [*Informatics*], 2021, vol. 18, no. 3, pp. 83–96 (In Russ.).

10. Loshchilov I., Hutter F. *SGDR: Stochastic Gradient Descent with Warm Restarts*, 2016. Available at: https://arxiv.org/abs/1608.03983 (accessed 01.08.2025).

11. Srivastava N., Hinton G., Krizhevsky A. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 2014, vol. 15, no. 1, pp. 1929–1958.

12. Liang S., Yin S., Liu L., Luk W., Wei S. FP-BNN: Binarized neural network on FPGA. *Neurocomputing*, 2018, vol. 275, pp. 1072–1086.

13. Huynh T. V. Deep neural network accelerator based on FPGA. *NAFOSTED Conference on Information and Computer Science, Hanoi, Vietnam, 24–25 November 2017.* Hanoi, 2017, pp. 254–257.

14. Krivalcevich E. A., Vashkevich M. I. *Investigation of hardware implementation of a feedforward neural network for handwritten digit recognition based on FPGA*. *Doklady BGUIR*, 2025, vol. 23, no. 2, pp. 101–108 (In Russ.).

15. Solovyev R. A., Kustov A. G., Ruhlov V. S., Shchelokov A. N., Puzyrkov D. V. *Hardware implementation of a convolutional neural network in FPGA based on fixed point calculations*. Izvestija Juzhnogo federal'nogo universiteta. Tehnicheskie nauki [*Proceedings of Southern Federal University. Engineering Sciences*], 2017, vol. 192, no. 7, pp. 186–197 (In Russ.).

Информация об авторах

Кривальцевич Егор Александрович, магистрант кафедры электронных вычислительных средств, Белорусский государственный университет информатики и радиоэлектроники.

Вашкевич Максим Иосифович, доктор технических наук, профессор кафедры электронных вычислительных средств, Белорусский государственный университет информатики и радиоэлектроники.
E-mail: vashkevich@bsuir.by

Information about the authors

Egor A. Krivalcevich, Undergraduate of Computer Engineering Department, Belarusian State University of Informatics and Radioelectronics.

Maxim I. Vashkevich, D. Sc. (Eng.), Prof. of Computer Engineering Department, Belarusian State University of Informatics and Radioelectronics.
E-mail: vashkevich@bsuir.by