

ФИЗИКО-МАТЕМАТИЧЕСКИЕ НАУКИ

Научная статья
УДК 004.032.26
doi: 10.34822/1999-7604-2022-2-66-74

ФОРМАЛИЗАЦИЯ ПРОЦЕДУРЫ ПОДБОРА СВЕРТОЧНЫХ ЯДЕР

**Вахит Мансурович Гиниятуллин^{1✉}, Евгений Витальевич Ермолаев²,
Александр Владимирович Хлыбов³**

^{1, 2, 3}Уфимский государственный нефтяной технический университет, Уфа, Россия

¹fentazer@mail.ru[✉], <http://orcid.org/0000-0002-4686-2752>

²evg.ermo2009@gmail.com, <http://orcid.org/0000-0003-0439-3035>

³brinkinvision@gmail.com, <http://orcid.org/0000-0002-9586-0681>

Аннотация. В работе представлено описание методики подбора сверточных ядер на примере датасета MNIST. Подробно рассмотрен алгоритм, приведено обоснование возможности его применения, проведен анализ полученных результатов. Разработанный алгоритм в перспективе позволит заменить обучаемые сверточные ядра, состоящие из чисел с плавающей запятой, на дискретные, которые подбираются для каждой прикладной области единожды и позволяют обойтись без обучения сверточных слоев в нейронных сетях.

Ключевые слова: нейронные сети, сверточный слой, матрица весов, глубокое обучение, подбор ядер

Благодарности: работа выполнена в рамках программы стратегического академического лидерства «Приоритет-2030» при поддержке федерального государственного бюджетного учреждения «Фонд содействия развитию малых форм предприятий в научно-технической сфере» (Фонд содействия инновациям).

Для цитирования: Гиниятуллин В. М., Ермолаев Е. В., Хлыбов А. В. Формализация процедуры подбора сверточных ядер // Вестник кибернетики. 2022. № 2 (46). С. 66–74. DOI 10.34822/1999-7604-2022-2-66-74.

Original article

CHARACTERIZING A PROCEDURE FOR SEARCHING CONVOLUTION KERNELS

Vakhit M. Giniyatullin^{1✉}, Evgeniy V. Ermolaev², Aleksandr V. Khlybov³

^{1, 2, 3}Ufa State Petroleum Technological University, Ufa, Russia

¹fentazer@mail.ru[✉], <http://orcid.org/0000-0002-4686-2752>

²evg.ermo2009@gmail.com, <http://orcid.org/0000-0003-0439-3035>

³brinkinvision@gmail.com, <http://orcid.org/0000-0002-9586-0681>

Abstract. The article describes a method for searching convolution kernels using the MNIST dataset. The algorithm is studied in detail. The possibility to apply the algorithm is substantiated. The obtained results are analyzed. The algorithm developed will make it possible to substitute discrete kernels, which are selected once for each applied field and do not require training of convolutional layers in neural networks, for trainable convolution kernels consisting of numbers with floating point.

Keywords: neural networks, convolutional layer, matrix of weights, deep learning, searching of kernels

Acknowledgments: the article is a part of a strategical academic leadership program “Priority-2030”, funded by the Fund for Assistance to Small Innovative Enterprises (Innovation Assistance Fund).

For citation: Giniyatullin V. M., Ermolaev E. V., Khlybov A. V. Characterizing a Procedure for Searching Convolution Kernels // *Proceedings in Cybernetics*. 2022. No. 2 (46). P. 66–74. DOI 10.34822/1999-7604-2022-2-66-74.

ВВЕДЕНИЕ

В настоящее время остро стоит вопрос решения одной из проблем глубокого обучения – возрастающей сложности архитектуры разрабатываемых моделей нейронных сетей и, как следствие, необходимой вычислительной мощности. Сложность существующих нейронных сетей растет вместе с объемами данных, которые необходимо обрабатывать [1]. Вычислительная сложность моделей ограничивает возможность их внедрения в устройства малой вычислительной мощности (мобильные устройства, встраиваемая техника и т. д.), а также увеличивает время расчетов.

Применяемая в нейронных сетях парадигма «черного ящика» приводит к тому, что получаемое нейронной сетью множество весовых коэффициентов невозможно интерпретировать [2]. В то же время необходимо четко осознать, почему модель сделала ошибку, и найти пути ее исправления, для чего все чаще практикуется увеличение числа нейронов и/или слоев, то есть усложняется архитектура модели.

Для решения части этих проблем был разработан алгоритм подбора сверточных ядер. Он состоит из следующих шагов: подбора на ядрах малой размерности (2×2 , 3×3), масштабирования ядер и/или добавления дополнительных ядер для достижения требуемой точности нейронной сети.

Предпосылки к упрощению нейронных сетей. Поскольку вычислительные мощности не являются неограниченными, логично появление различных методик упрощения применительно к существующим нейронным сетям [3–5]. В последнее время идет развитие методики прунинга (прореживания) нейронных сетей [6–7], которая заключается в том, что элементы (веса, нейроны), практически не влияющие на конечный результат, удаляются из обученной модели. Однако применение данной методики не распространено ввиду

отсутствия четкой формализации самого процесса прореживания. Нет четких указаний, удаление какого элемента не приведет к существенному падению точности.

Авторами были проведены исследования [8] на общедоступных наборах данных (dataset) и структурах нейронных сетей с сайта kaggle.com [9]. Исходные структуры сетей обучались «как есть», после обучения значения коэффициентов сверточных ядер первого сверточного слоя усекались до одного-трех десятичных знаков после запятой (табл. 1). Сверточные ядра представляют собой числовые матрицы, коэффициенты которых (веса) изменяются в ходе обучения нейронной сети. Такая матрица перемещается по изображению и уменьшает его размерность (операция свертки). В результате наложения сверточного ядра на определенный участок изображения получается число, полученное путем суммирования произведения коэффициентов сверточного ядра на значения пикселей исходного изображения. Таким образом, исходная матрица изображения преобразуется в матрицу меньшей размерности.

Во втором столбце приведена первоначальная точность модели. Поскольку для коэффициентов ядер используется тип float, количество значащих десятичных знаков не более 8 [10]. В третьем столбце представлены результаты после усечения до 1–3 десятичных знаков после запятой. При этом потеря точности после усечения оказалась несущественной. При усечении значений коэффициентов до одного десятичного знака для одного из датасетов (EGG Heartbeat Categorization Dataset) точность возросла. На рис. 1 представлены полученные коэффициенты ядер. В структуре этой нейронной сети в первом сверточном слое имеется 64 ядра размером 1×6 .

```
array([[[-0.1, -0. , -0. , 0.1, -0.1, 0.2, 0.1, -0.1, 0.1, -0.2,
        0.2, 0.3, 0.3, 0. , 0.3, 0.1, 0.1, 0.1, 0.1, -0.2,
        -0.3, -0.2, -0.1, 0. , 0.2, -0.2, -0. , -0.1, 0. , 0.1,
        0.1, -0.3, -0. , 0.1, 0.1, 0. , 0.1, 0.2, -0.1, -0.2,
        0.1, -0.2, 0.3, -0.3, -0.1, 0.3, -0.2, -0. , 0.1, -0.2,
        -0. , -0.3, -0.2, -0. , -0. , -0.1, -0.1, -0.2, 0.2, -0.1,
        0.2, -0.2, 0.2, 0.3]],

       [[-0. , 0.1, 0.2, 0.2, -0.1, 0. , 0. , -0.1, 0.1, -0.2,
        0.2, 0.1, 0.1, -0.1, -0.1, 0.1, 0.2, -0.1, -0.1, -0. ,
        -0.1, -0.1, 0.1, -0.1, -0.1, -0.2, -0.2, -0.1, -0.2, -0.2,
        -0.1, -0. , 0.2, 0. , 0. , -0.1, 0.2, -0.2, -0. , 0. ,
        0.2, 0.1, 0.2, -0. , -0.2, -0.2, -0.2, 0.1, -0. , 0.1,
        -0.1, -0.1, -0.3, -0.2, -0.2, 0.2, -0.3, -0. , 0. , 0.1,
        0.2, -0.2, -0. , 0.1]],

       [[-0.1, 0.1, 0.2, 0.2, -0. , -0.2, -0. , -0.2, 0.1, -0.1,
        0.1, -0.1, -0.1, -0.1, 0. , 0.1, -0.1, -0.2, -0.2, -0. ,
        -0.1, 0.2, 0.1, 0.2, -0.2, -0. , 0. , -0.1, -0.2, 0.2,
        -0.1, -0. , -0.1, 0.3, 0.1, 0. , -0.1, 0.1, 0.2, -0.1,
        0.1, -0.2, 0. , 0.2, -0. , -0.1, -0. , 0.2, -0.2, 0.2,
        -0.1, 0. , -0. , -0. , -0.1, 0.2, -0. , -0.2, 0.1, 0.2,
        -0. , 0.1, -0.1, -0. ]],

       [[-0.2, 0.1, 0.1, 0.1, -0.1, -0. , 0.1, 0. , 0. , -0. ,
        -0.1, -0. , -0. , 0. , -0. , 0.2, 0.1, 0. , -0.1, 0. ,
        -0. , 0.2, 0.1, 0.2, -0.2, 0.1, -0.1, -0.2, -0.1, 0.1,
        0. , -0.2, -0.1, -0. , 0.1, 0.2, 0.1, -0.1, 0.1, 0.2,
        0.1, -0.1, 0.1, -0.1, 0.1, 0. , 0. , 0.1, -0.1, -0.1,
        -0. , 0.1, -0. , 0.1, -0. , -0.2, -0. , -0.1, 0.1, 0. ,
        -0.1, -0.1, -0.1, -0.1]],

       [[-0.2, 0.1, -0.1, 0.1, 0.2, 0.1, 0.1, 0.1, 0. , 0. ,
        -0.1, 0. , -0. , -0.1, -0.1, 0. , -0.1, 0.1, 0. , 0. ,
        0.1, -0.1, 0.2, -0.1, -0. , 0.1, -0.2, -0. , -0. , -0.2,
        -0.1, 0.1, -0.2, 0.1, 0. , 0.1, 0. , -0. , 0.2, -0. ,
        -0.1, -0.1, 0. , 0. , -0. , -0. , 0.1, 0.1, -0.2, 0. ,
        -0.1, 0. , -0.1, 0.1, -0.1, 0.1, -0.2, 0.1, 0.1, -0.1,
        0. , -0.1, -0. , 0.1]],

       [[-0.1, 0.1, 0. , 0. , 0. , 0. , 0.1, 0.1, -0.1, 0.1, 0.1,
        -0. , 0. , -0. , -0. , 0. , 0.2, -0.1, -0.1, -0.1, 0.3,
        0.1, 0.1, 0.1, -0. , 0.1, 0. , 0.1, -0.2, -0.1, -0.1,
        0.2, 0.2, -0.1, -0.1, 0.1, 0.2, 0.1, 0.2, 0. , -0.2,
        -0. , 0. , 0.1, 0. , -0. , 0.1, 0. , 0.1, -0.1, -0.1,
        -0.2, 0. , -0. , 0. , -0.2, 0.1, -0.1, -0.2, -0.1, -0.1,
        0. , 0.1, -0.3, 0.1]]])
```

Рис. 1. Коэффициенты ядра первого сверточного слоя
после усечения до одного десятичного знака после запятой
Примечание: составлено авторами.

Из рис. 1 следует, что коэффициенты ядер могут принимать всего 7 различных значений в диапазоне от $-0,3$ до $+0,3$ с шагом

в $0,1$. В подобных ситуациях имеется возможность рассматривать их как многозначные (в данном случае 7-значные) логики.

Таблица 1

Усечение коэффициентов ядер сверток

Набор данных	Точность после первоначального обучения, %	Точность после обучения с усеченными коэффициентами ядер до десятичных знаков, %
MalariaDataset	94,75	94,54 (до двух знаков)
EGGHeartbeatCategorizationDataset	80,72	81,04 (до одного знака)
Dogsvs. Cats	78,73	76,28 (до трех знаков)
ObjectClassification	88,74	85,6 (до двух знаков)
SignLanguageMNIST	100	95,8 (до двух знаков)

Примечание: составлено авторами.

Дальнейшее усечение значности приводит к более существенной потере точности.

В табл. 2 приведены результаты усечения до 3 знаков (-0.1 ; 0 ; $+0.1$).

Таблица 2

Усечение значений сверток до 3 знаков

Набор данных	Точность после первоначального обучения, %	Точность после усечения до 3 знаков, %
MalariaDataset	94,75	86
EKGHeartbeatCategorizationDataset	80,72	72,11
ObjectClassification	88,74	82,2
SignLanguageMNIST	100	98,45

Примечание: составлено авторами.

Таким образом, коэффициенты сверточных ядер могут быть дискретными, в частности троичными, и тогда возникает возможность применения в сверточных слоях не обучаемых ядер, а подбираемых.

МАТЕРИАЛЫ И МЕТОДЫ

Реализация разработанных алгоритмов осуществлена на языке программирования Python с использованием библиотеки глубокого обучения Tensorflow. Процесс перебора проводился на общедоступном наборе изображений рукописных цифр MNIST [11]. Размерность каждого изображения 28×28 пикселей. Значения пикселей изображения изменяются в диапазоне от 0 до 255 включительно.

Исходная структура нейронной сети LeNet-5 [12] содержит первый сверточный слой с 6 ядрами 5×5 , maxpooling-слой с размером ядра 2×2 и шагом 2, второй сверточный слой с 16 ядрами 5×5 , еще один аналогичный

maxpooling-слой и два следующих друг за другом полносвязных слоя: первый содержит 120 нейронов, второй – 84 нейрона. Выходной полносвязный слой содержит 10 нейронов. Точность распознавания по обучающей выборке равна 99,17 %, тестовой – 98,8 %. Для вычисления точности нейронной сети была использована метрика ассигасу, которая представляет собой отношение количества правильно предсказанных экземпляров к общему их количеству.

РЕЗУЛЬТАТЫ И ИХ ОБСУЖДЕНИЕ

Полный перебор ядер 3×3 . Структура сети LeNet-5 была упрощена до следующей: один сверточный слой с одним ядром 3×3 и один полносвязный слой с десятью нейронами. В результате точность модели снизилась по обучающей выборке до 90,91 %, по тестовой – до 91,06 %. На рис. 2 представлено ядро после обучения нейронной сети.

0.4992	0.4859	-0.2383
-0.1556	-0.097	0.1565
0.0781	-0.3754	-0.0652

Рис. 2. Обученное сверточное ядро (train_acc = 90,91 %, test_acc = 91,06 %)

Примечание: составлено авторами.

Предлагается в качестве коэффициентов сверточного ядра использовать дискретные подбираемые значения вместо многозначных обучаемых. Для исследования был выбран трехзначный диапазон, состоящий из чисел $-1, 0$ и $+1$.

Всего вариантов перебора $3^9 = 19\,683$. Таким образом, из каждого ядра свертки можно сформировать девятиразрядное число троично-сбалансированной системы счисления (далее – ТСС) [13]. Троично-сбалансированная

система счисления – позиционная система счисления, основание которой равно 3, состоит из цифр $-1, 0, +1$. Среди особенностей ТСС обычно называют емкость, близкую к максимальной [14]; эквивалентность операций округления и усечения [15]. В работе [16] обсуждается малоизвестный эффект компенсации ошибок округления.

Ячейки матрицы сверточного ядра нумеруются, начиная с нуля, младший разряд слева вверху (рис. 3). В результате полного

перебора наибольшая точность нейронной сети по обучающей выборке (train_acc) составила 92,49 %, а по тестовой (test_acc) –

92,60 %, что соответствует ядрам № –5 468 и № 6 168 (рис. 3).

Ядро №-5468			Ядро №6168		
0	-1	1	0	2	1
3	1	4	1	5	1
6	1	7	1	8	1

Рис. 3. Ядра с наибольшей точностью распознавания по обучающей (№ -5 468, train_acc = 92,49 %, test_acc = 92,57 %) и тестовой (№ 6 168, train_acc = 92,28 %, test_acc = 92,60 %) выборке
 Примечание: составлено авторами.

Ядра 3×3 с максимальной точностью распознавания оказались несбалансированными, «скошенными» в положительную область. Они содержат одно значение «-1», два-три значения «0» и остальные значения «+1», в отличие от обученного ядра (рис. 2),

где значения распределены примерно равномерно.

Ядра с наименьшей точностью распознавания имеют примерно одинаковую точность (train_acc = 11,24 %, test_acc = 11,35 %). Примеры таких ядер представлены на рис. 4.

Ядро №-9841			Ядро №-30		
0	-1	1	-1	2	-1
3	-1	4	-1	5	-1
6	-1	7	-1	8	-1

Рис. 4. Примеры ядер с наименьшей точностью распознавания
 Примечание: составлено авторами.

Масштабирование ядра. Увеличение размера ядра позволяет сократить выходной вектор сверточного слоя и увеличить рецептивное поле восприятия [17]. Для динамических случаев (видеопоток данных) необходимы уже не двумерные ядра, а объемные (3D). Полный перебор таких ядер практически невозможен, поэтому предлагается подобрать ядро малой размерности (2×2 или 3×3) с высокой точностью распознавания, затем увеличивать его размер.

Процедура масштабирования сводится к последовательному добавлению строк/столбцов к исходному ядру. Таким образом, формируются вполне работоспособные ядра,

возможно, не оптимальные. Для масштабирования было выбрано ядро с максимальной точностью по обучающей выборке (№ –5 468). Именование получаемых таким образом ядер состоит в следующем: сначала дается номер исходного ядра, затем, через точку с запятой, – номера перебираемых строк/столбцов.

В процессе масштабирования можно менять длину шага прохода сверточного ядра по изображению. Ниже представлены результаты для двух случаев: 1) когда шаг меняется пропорционально увеличению размера ядра; 2) когда шаг остается постоянным и равен шагу для малого ядра (табл. 3 и 4).

Таблица 3

Масштабирование с пропорциональным увеличением шага

Размерность	Десятичное представление строки/столбца в виде ТСС	Шаг	Точность, %
4×3	4	(3, 2)	91,86
4×4	37	(3, 3)	91,59
5×4	39	(4, 3)	89,60
5×5	94	(4, 4)	88,57
6×5	-41	(5, 4)	87,11
6×6	283	(5, 5)	85,53

Примечание: составлено авторами.

Таблица 4

Масштабирование с постоянным малым шагом

Размерность	Десятичное представление строки/столбца в виде ТСС	Шаг	Точность, %
4×3	-9	(2, 2)	92,78
4×4	28	(2, 2)	92,82
5×4	-30	(2, 2)	92,89
5×5	-35	(2, 2)	92,80
6×5	-18	(2, 2)	92,94
6×6	208	(2, 2)	93,17

Примечание: составлено авторами.

Таким образом, в случае с пропорциональным увеличением шага получаемое путем масштабирования ядро 6×6 именуется следующим образом: -5 468; 4; 37; 39; 94;

-41; 283 (рис. 5). Процедура масштабирования с постоянным малым шагом приводит к другому ядру: -5 468; -9; 28; -30; -35; -18; 208 (рис. 6).

Ядро №-5468; 4; 37; 39; 94; -41; 283						
-1	0	1	37	1	94	283
1	1	1	1	0	0	
1	1	1	0	1	1	
-5468	0	1	1	1	1	
4	1	1	1	0	1	1
39	1	1	1	0	1	1
-41	1	1	1	1	1	1

Рис. 5. Ядро, полученное путем масштабирования с пропорциональным увеличением шага

Примечание: составлено авторами.

Ядро №-5468; -9; 28; -30; -35; -18; 208						
-1	0	1	28	1	-35	208
1	1	1	0	-1	0	
1	1	1	0	-1	-1	
-5468	-1	0	0	1	0	-1
-9	-1	0	-1	0	1	0
-30	0	-1	0	0	0	1
-18	0	-1	1	0	0	1

Рис. 6. Ядро, полученное путем масштабирования с постоянным малым шагом

Примечание: составлено авторами.

Из табл. 3 и 4 видно, что при малом шаге смещения ядра точность сети увеличивается, при большом – наблюдается существенное падение точности. Таким образом, возникает необходимость в подборе такого шага и размера ядра, когда точность является приемлемой для решения задачи и реализуются преимущества больших ядер (сокращение размерности выходного вектора сверточного слоя и увеличение рецептивного поля).

Добавление дополнительных ядер 3×3 . Чтобы увеличить точность модели, к исходному ядру 3×3 , полученному в процессе перебора, можно добавить дополнительные ядра. Для эксперимента было выбрано ядро с максимальной точностью по обучающей выборке. В табл. 5 представлены номера

дополнительных ядер и прирост точности; на рис. 7 представлены сами ядра. По отдельности эти ядра дают следующие значения точности: для ядра № 2 903 точность равна 90,60 %, ядра № –8 969 – 89,67 %, ядра № –3 873 – 88,04 %.

Очевидно, что увеличивая количество ядер, можно добиться требуемой точности, при этом архитектура нейронной сети радикально упростилась:

- полносвязных (скрытых) слоев нет совсем (204 нейрона в двух слоях);
- maxpooling-слои отсутствуют (2 ядра в двух слоях);
- сверточный слой один с 4 ядрами 3×3 против двух слоев с ядрами 5×5 (6 и 16 ядер соответственно).

Таблица 5

Изменение точности от количества ядер

Кол-во ядер	Номер ядра	Точность, %
1	–5 468	92,49
2	2 903	96,16
3	–8 969	97,58
4	–3 873	98,31

Примечание: составлено авторами.

-5468			2903			-8969			-3873		
-1	0	1	0	1	1	-1	-1	0	-1	1	1
1	1	1	0	0	0	-1	0	1	-1	0	1
1	1	1	-1	-1	-1	-1	1	1	-1	-1	0

Рис. 7. Дополнительные ядра 3×3

Примечание: составлено авторами.

ЗАКЛЮЧЕНИЕ

Использование в качестве коэффициентов ядер сверточных слоев подбираемых, а не обучаемых ядер возможно и приводит к результатам с приемлемой точностью. Подбор ядра и использование дискретных коэффициентов позволят радикально упро-

стить архитектуру модели и снизить требования к вычислительным ресурсам.

На длительность перебора влияют три составляющие: размер ядра, шаг свертки и количество ядер. Формулировка критерия их выбора позволит прогнозировать длительность перебора и объем необходимых ресурсов.

Список источников

- Виноградов О. В., Морозова О. А. Аспекты применения нейронных сетей для прогнозирования чрезвычайных ситуаций // Технологии граждан. безопасности. 2021. № 18. С. 23–26.
- Андреев О. А., Трофимов А. Т. Интерпретация весовых коэффициентов и функций активации

References

- Vinogradov O. V., Morozova O. A. Aspects of Neural Networks Use for Predicting Emergency Situations // Civil Security Technology. 2021. No. 18. P. 23–26. (In Russian).
- Andreev O. A., Trofimov A. T. Interpretation of Weights and Transfer Functions of Artificial Neural

- искусственных нейронных сетей // Цифровая обработка сигналов и ее применение – DSPA-2019. М., 2019. С. 565–569.
3. Gokulanathan S., Feldsher A., Malca A., Barret C., Katz G. Simplifying Neural Networks Using Formal Verification // Proceedings of the 12th NASA Formal Methods Symposium (NFM). 2020. P. 85–93.
 4. Schirrmester T. R., Liu R., Hooker S., Ball T. When Less Is More: Simplifying Inputs Aids Neural Networks Understanding // ArXiv. 2022. URL: <https://arxiv.org/abs/2201.05610> (дата обращения: 22.03.2022).
 5. Pan W., Dong H., Huo Y. DropNeuron: Simplifying the Structure of Deep Neural Networks // ArXiv. 2016. URL: <https://arxiv.org/abs/1606.07326> (дата обращения: 22.03.2022).
 6. Tanaka H., Kunin D., Yamins D., Ganguli S. Pruning Neural Networks without Any Data by Iteratively Conserving Synaptic Flow // ArXiv. 2020. URL: <https://arxiv.org/abs/2006.05467> (дата обращения: 22.03.2022).
 7. Wang C., Zhang G., Grosse R. Picking Winning Tickets before Training by Preserving Gradient Flow // ArXiv. 2020. URL: <https://arxiv.org/abs/2002.07376> (дата обращения: 22.03.2022).
 8. Хлыбов А. В., Ермолаев Е. В., Зеленов А. С., Кonyaева Е. Э., Фаррахова К. А., Хужиахметов В. Э., Яковлев А. С. Преобразование весов матриц сверточных нейронных сетей // Теория и практика процессов химической технологии (Марушкинские чтения) : материалы VI Междунар. науч. конф. Уфа : Изд-во УГНТУ, 2021. С. 269–270.
 9. EGG Heartbeat Categorization Dataset. URL: <https://www.kaggle.com/shayanfazeli/heartbeat> (дата обращения: 23.03.2022).
 10. IEEE – представление с плавающей точкой. URL: <https://docs.microsoft.com/ru-ru/cpp/build/ieee-floating-point-representation?view=msvc-170> (дата обращения: 22.03.2022).
 11. Digit Recognizer. URL: <https://www.kaggle.com/c/digit-recognizer/data> (дата обращения: 22.03.2022).
 12. Module05_mnist_conv.ipynb. URL: https://github.com/SlinkoIgor/Neural_Networks_and_CV/blob/master/module05_mnist_conv.ipynb (дата обращения: 22.03.2022).
 13. Buntine A. The Balanced Ternary Machines of Soviet Russia. URL: <https://dev.to/buntine/the-balanced-ternary-machines-of-soviet-russia> (дата обращения: 22.03.2022).
 14. Кушнеров А. Троицкая цифровая техника. Ретроспектива и современность. URL: <http://314159.ru/kushnerov/kushnerov1.pdf> (дата обращения: 22.03.2022).
 15. Кнут Д. Э. Искусство программирования / под общ. ред. Ю. В. Козаченко. М. : Вильямс, 2007. 822 с.
 16. Гиниятуллин В. М., Салихова М. А. Эффект компенсации ошибок округления в троично-сбалансированной системе счисления // Вестник кибернетики. 2020. № 4 (40). С. 14–20.
 - Networks // Digital Signal Processing – DSPA-2019. Moscow, 2019. P. 565–569. (In Russian).
 3. Gokulanathan S., Feldsher A., Malca A., Barret C., Katz G. Simplifying Neural Networks Using Formal Verification // Proceedings of the 12th NASA Formal Methods Symposium (NFM). 2020. P. 85–93.
 4. Schirrmester T. R., Liu R., Hooker S., Ball T. When Less Is More: Simplifying Inputs Aids Neural Networks Understanding // ArXiv. 2022. URL: <https://arxiv.org/abs/2201.05610> (accessed: 22.03.2022).
 5. Pan W., Dong H., Huo Y. DropNeuron: Simplifying the Structure of Deep Neural Networks // ArXiv. 2016. URL: <https://arxiv.org/abs/1606.07326> (accessed: 22.03.2022).
 6. Tanaka H., Kunin D., Yamins D., Ganguli S. Pruning Neural Networks without Any Data by Iteratively Conserving Synaptic Flow // ArXiv. 2020. URL: <https://arxiv.org/abs/2006.05467> (accessed: 22.03.2022).
 7. Wang C., Zhang G., Grosse R. Picking Winning Tickets before Training by Preserving Gradient Flow // ArXiv. 2020. URL: <https://arxiv.org/abs/2002.07376> (accessed: 22.03.2022).
 8. Khlybov A. V., Ermolaev E. V., Zelenov A. S., Konyaeva E. E., Farrakhova K. A., Khuzhiakhmetov V. E., Yakovlev A. S. Preobrazovanie vesov matrits svertochnykh neironnykh setei // Teoriia i praktika protsessov khimicheskoi tekhnologii (Marushkinskie chteniia) : Proceedings of the VI International Research Conference. Ufa : Publishing House Ufa State Petroleum Technological University, 2021. P. 269–270. (In Russian).
 9. EGG Heartbeat Categorization Dataset. URL: <https://www.kaggle.com/shayanfazeli/heartbeat> (accessed: 23.03.2022).
 10. IEEE – predstavlenie s plavaiushchei tochkoi. URL: <https://docs.microsoft.com/ru-ru/cpp/build/ieee-floating-point-representation?view=msvc-170> (accessed: 22.03.2022). (In Russian).
 11. Digit Recognizer. URL: <https://www.kaggle.com/c/digit-recognizer/data> (accessed: 22.03.2022).
 12. Module05_mnist_conv.ipynb. URL: https://github.com/SlinkoIgor/Neural_Networks_and_CV/blob/master/module05_mnist_conv.ipynb (accessed: 22.03.2022).
 13. Buntine A. The Balanced Ternary Machines of Soviet Russia. URL: <https://dev.to/buntine/the-balanced-ternary-machines-of-soviet-russia> (accessed: 22.03.2022).
 14. Kushnerov A. Troichnaia tsifrovaia tekhnika. Retrospektiva i sovremennost. URL: <http://314159.ru/kushnerov/kushnerov1.pdf> (accessed: 22.03.2022). (In Russian).
 15. Knuth D. E. The Art of Computer Programming / Ed. Yu. V. Kozachenko. Moscow : Williams, 2007. 822 p. (In Russian).
 16. Giniyatullin V. M., Salikhova M. A. Effect of Rounding Error Compensation in the Ternary Balanced Number System // Proceedings in Cybernetics. 2020. No. 4 (40). P. 14–20. (In Russian).

17. Немков Р. М. Исследование сверточной нейронной сети, обученной с помощью метода применения нестандартных рецептивных полей при распознавании изображений // Изв. ЮФУ. Технич. Науки. 2015. С. 79–90.

Информация об авторах

В. М. Гиниятуллин – кандидат технических наук, доцент.

Е. В. Ермолаев – магистрант.

А. В. Хлыбов – аспирант.

17. Nemkov R. M. Investigation of Convolution Neural Network which Was Trained by the Method of Use of Non-Standard Receptive Fields in the Process of Image Recognition // Izvestiya SFedU. Engineering Sciences. 2015. P. 79–90. (In Russian).

Information about the authors

V. M. Giniyatullin – Candidate of Sciences (Engineering), Associate Professor.

E. V. Ermolaev – Master's Degree Student.

A. V. Khlybov – Postgraduate.