Оригинальная статья

УДК 004.93

DOI: 10.57070/2304-4497-2023-1(43)-39-49

ИСПОЛЬЗОВАНИЕ СВЕРТОЧНЫХ НЕЙРОСЕТЕЙ ДЛЯ КЛАССИФИКАЦИИ ИЗОБРАЖЕНИЙ

© 2023 г. А. Г. Бычков, Т. В. Киселёва, Е. В. Маслова

Сибирский государственный индустриальный университет (Россия, 654007, Кемеровская обл. – Кузбасс, Новокузнецк, ул. Кирова, 42)

Аннотация. В работе рассматриваются структура сверточной нейронной сети и математические методы, используемые для подсчета ее значений. Приведены основные составные части сети, влияющие на результат: сверточные слои с маской как основа сетки данных, ядро для чтения сетки данных, шага и дополнения для настройки точности чтения, субдискретизирующие слои для обобщения данных. Показана история развития сверточных нейронных сетей с примерами их архитектуры и используемыми параметрами на примере сетей LeNet, AlexNet, VGG и ResNet. Показано сравнение точности распознавания образов при разных архитектурах. Описана концепция передаточного обучения.

Ключевые слова: сверточные нейронные сети, распознавание образов, точность работы

Для цитирования: Бычков А.Г., Киселёва Т.В., Маслова Е.В. Использование сверточных нейросетей для классификации изображений // Вестник Сибирского государственного индустриального университета. 2023. № 1 (43). С. 39–49. *http://doi.org/10.57070/2304-4497-2023-1(43)-39-49*

Original article

USAGE OF CONVOLUTIONAL NEURAL NETWORKS FOR IMAGE CLASSIFICATION

© 2023 A. G. Bychkov, T. V. Kiseleva, E. V. Maslova

Siberian State Industrial University (42 Kirova Str., Novokuznetsk, Kemerovo Region – Kuzbass, 654007, Russian Federation)

Abstract. The paper discusses the structure of a convolutional neural network and the mathematical methods used to calculate its values. The main components of the network that affect the result are given: convolutional layers with a mask as the basis of the data network, a core for reading the data network, steps and additions for adjusting the reading accuracy, subsampling layers for generalizing data. The history of the development of convolutional neural networks with examples of their architecture and parameters used is shown on the example of networks LeNet, AlexNet, VGG and ResNet. A comparison of the accuracy of pattern recognition for different architectures is shown. The concept of transfer learning is described.

Keywords: convolutional neural networks, pattern recognition, transfer learning, performance accuracy

For citation: Bychkov A.G., Kiseleva T.V., Maslova E.V. Usage of convolutional neural networks for image classification. *Bulletin of the Siberian State Industrial University.* 2023, no. 1 (43), pp. 39–49. (In Russ.). *http://doi.org/10.57070/2304-4497-2023-1(43)-39-49*

Введение

Машинное обучение (англ. machine learning, ML) – класс методов искусственного интеллекта, характерной чертой которых является не прямое решение задачи, а обучение в процессе применения решений множества сходных задач. Для построения таких методов используются средства математической статистики, численные методы, методы оптимизации, теории вероятностей, теории графов, различные техники работы с данными в цифровой форме. Искусственный интеллект сыграл колоссальную роль в преодолении разрыва между возможностями людей и машин. Как исследователи, так и энтузиасты работают над многочисленными аспектами этой области, добиваясь удивительных результатов. Одним из них является компьютерное зрение. Примером машинного обучения в компьютерном зрении являются сверточные нейронные сети [1].

В настоящей работе приведен обзор и сравнительный анализ известных архитектур сверточных нейронных сетей, их преимущества, история развития и способы использования на практике.

Структура сверточных нейронных сетей

У базового перцептрона имеется недостаток: требование огромного количества входных значений, причем малейший сдвиг изображения заставляет все делать заново [2 – 7]. Чтобы избежать этого ограничения, было предложено использовать сверточные нейросети. Идея состоит в том, что веса и значения активации какого-либо нейрона зависят не от всех входов, а только от некоторой окрестности «вокруг» него. Такое «движущееся окно» проходит по всей картинке [8, 9]. В итоге для соседнего «пикселя» в сверточном слое будут те же самые веса, но входной кусок картинки окажется сдвинутым.

В результате в одной ячейке сверточного слоя записывается определенный паттерн изображения (линии, углы, цветовые пятна и т.п.), который в дальнейшем прогоняется с целью нахождения похожих паттернов в других частях изображения либо других изображениях [4]:

$$O_{xyc'} = \sum_{i,j,c} W_{ijcc'} X_{x-i,y-j,c}; \quad (1)$$

$$O_{x_1 y_1 c'} = \sum_{i,j,c} W_{ijcc'} X_{x_1-i,y_1-j,c}, \quad (2)$$

где O – итоговое значение в ячейке сверточного слоя; W – веса связей; X – значение в исходной ячейке; i, j, c, c' – координаты ячеек.

Далее активационная функция работает по такому же принципу.

Возникает некоторая иерархичность между слоями. Если первые слои распознают лишь простейшие элементы изображения (линии, точки, дуги), то последующие слои уже начинает объединять эти части в фигуры [9, 10].

Двумя важными параметрами являются дополнение и шаг (padding и stride).

Не всегда выгодно, чтобы на каждом шаге размер сетки уменьшался. Например, если мы проходим исходный набор 4×4 ядром (kernel) 3×3, то на выходе получится матрица 2×2 (рис. 1).

Для удобства подсчета лучше на каждом шаге иметь одинаковые размерности матриц весов. Для этого используется дополнение (padding). Пример показан на рис. 2. Вход дополняют каким-либо числом по краям, чаще всего ставится 0 [11]. Таким образом, размер входа будет равен размеру выхода.

Шаг (stride) показывает, сколько элементов проходится за одну итерацию. Пример использования шага показан на рис. 3.

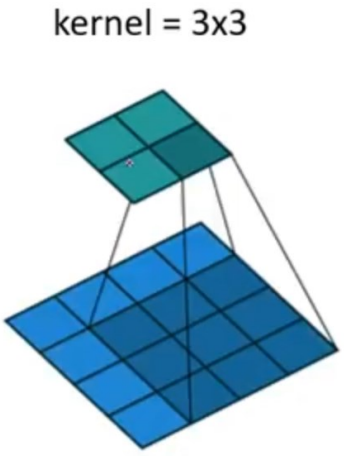

Рис. 1. Пример ядра свертки
Fig. 1. Example of convolution kernel

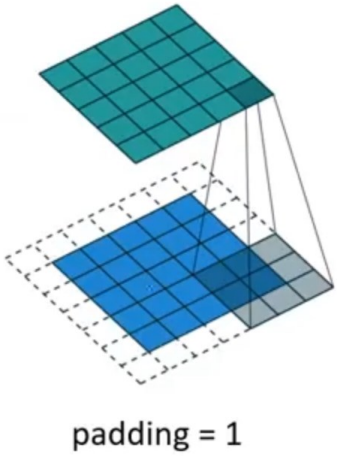

Рис. 2. Пример использования дополнения
Fig. 2. Example of add-on usage

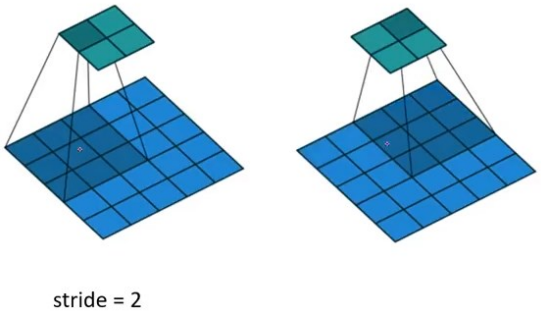

Рис. 3. Пример использования шага
Fig. 3. Example of step usage

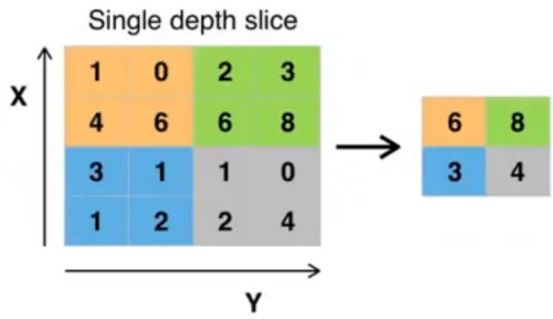

Рис. 4. Схема Max-Pooling
Fig. 4. Max pooling scheme

Оба этих метода вполне можно комбинировать.

Важной особенностью сверточных нейронных сетей является Pooling layer или, как называют его в некоторых источниках, субдискретизирующий слой [6]. Принцип действия его работы приведен на рис. 4.

Этот слой дает возможность менять длину и высоту активационных карт. Поставив этот слой после ReLU (ReLU – активационная функция), можно сжать получившиеся значения. Делается это по простому правилу, к примеру, используется Max Pool: выбирается максимальное значение из квадрата 2×2 и передается дальше (рис. 5). Это позволяет во время обратного прохода отнести весь градиент именно к этому значению, а остальным поставить нули (максимум зависит от одного входа).

Такое сжатие позволяет сделать информацию, которой оперирует сеть, более глобальной. В вышеуказанном примере в левом верхнем квадранте будет самым важным число 6, а остальные входы имеют меньшее значение, что позволит применять обученную в дальнейшем сеть к более широкому классу изображений, избегая тем самым переобучения модели. Это дает возможность передавать дальше лишь самые выделяющиеся признаки на картинке, не вдаваясь в мелкие подробности (избегая переобучения), но при этом, обучая модель, видеть всю картинку целиком [11]. В итоге условная сверточная нейросеть может выглядеть, как это показано на рис. 5. На ее вход подается изображение, дальше сверточный слой его обрабатывает, затем оно пропускается через активационную функцию ReLU. Может быть использовано несколько комбинаций сверточных и ReLU-слоев. Далее происходит сжатие с помощью Pooling Level. Вся эта конструкция может повторяться, пока не получится сверточный слой, достаточно малый для того, чтобы его можно было сделать «плоским» – перевести в одномерный массив, – связать с полносвязным слоем, который можно пропустить через Softmax и получить уже готовый результат [12].

Данная архитектура имеет большое преимущество: она по умолчанию адаптирована к сдвигам изображений. Если, к примеру, нужный элемент переместился в другой угол изображения, то сверточные веса тоже сдвинутся. Но в отличие от перцептрона, есть еще и Pooling-слои, которые учитывают максимальное значение активации. Через несколько итераций таких слоев в итоге на выходе окажется тот слой, который отвечает за распознавание именно данного элемента [13].

История развития базовой структуры сверточных сетей

В плане архитектур одной из самых первых была сеть LeNet (рис. 6).

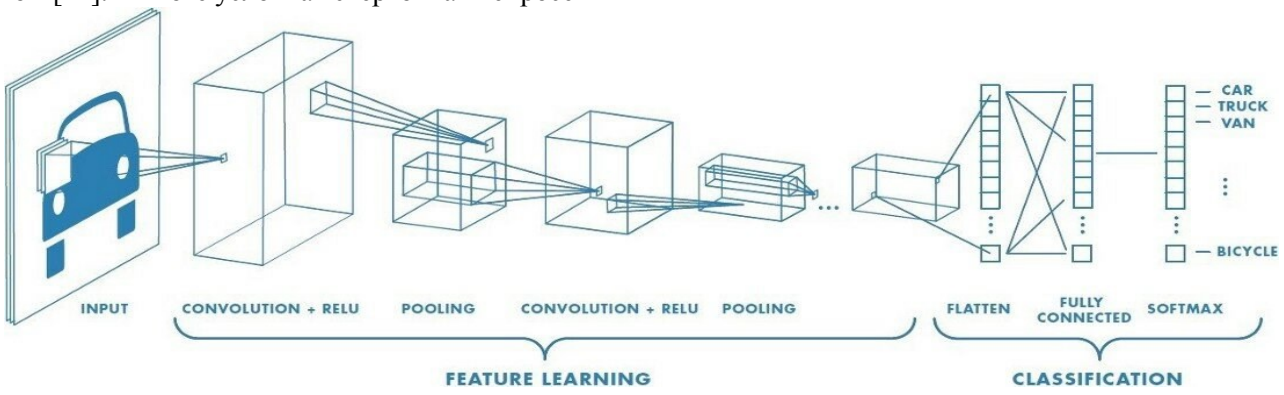

Рис. 5. Схема условной сверточной нейросети
Fig. 5. Diagram of a conditional convolutional neural network

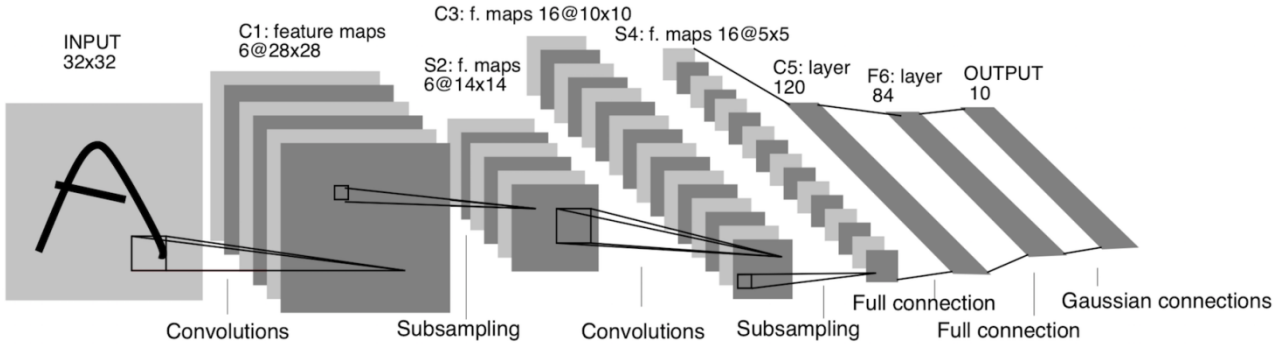

Рис. 6. Схема LeNet
Fig. 6. LeNet diagram

Архитектура сети LeNet была придумана Яном Лекуном (поэтому и такое название) в 1989 году как продолжение модели неокогнитрона (neocognitron) [14]. Модель сверточной сети состоит из трех типов слоев: сверточные (convolutional); субдискретизирующие (subsampling, подвыборка); "обычной" нейронной сети (перцептрона). Первые два типа слоев, чередуясь между собой, формируют входной вектор признаков для многослойного перцептрона. Сеть можно обучать с помощью градиентных методов.

Изначально Лекун использовал эту систему для распознавания отдельных цифр почтовых индексов. Сейчас подобная архитектура применяется в основном для обучения студентов и пробы сил на датасете MNIST. Все современные сети обычно проверяют на наборе IMAGENET, который содержит 1 000 000 изображений, принадлежащих 1000 классов [15]. Примеры успешности различных архитектур приведены на рис. 7 (штриховой линией показан уровень человеческого восприятия: около 5 % неверно распознанных изображений). Как видно из графика, современные архитектуры сверточных нейросетей намного превышают человеческий глаз. Стоит отметить, что эти 5 % показаны весьма условно, так как для выявления такой оценки один из сотрудников компании ImageNet обучался месяц и показал в финальном тестировании примерно такой результат [16].

Как видно на графике, серьезный прорыв произошел с появлением архитектуры AlexNet в 2012 году (рис. 8). Данная сеть имела примерно 60 000 000 параметров.

Именно на этой архитектуре впервые производилось обучение на GPU (видеокартах). В ходе разработки возникало множество ошибок, связанных с тем, что видеокарты не были подготовлены к таким вычислениям. Собственно, именно из-за этого пришлось дробить сеть на две части, что видно на рис. 8. Именно появление сети AlexNet стало поворотной точкой, после которой

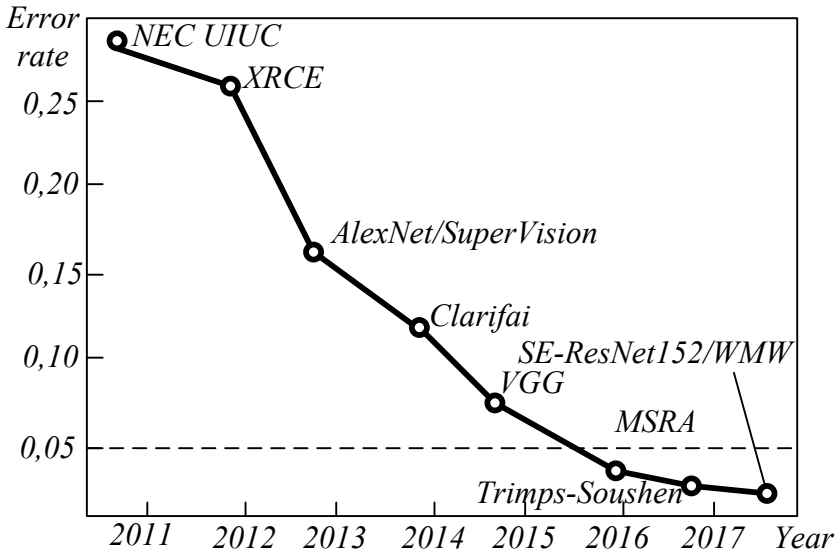

Рис. 7. График результатов различных архитектур:
– – – – уровень человеческого восприятия
Fig. 7. Graph results of different architectures:
– – – – level of human perception

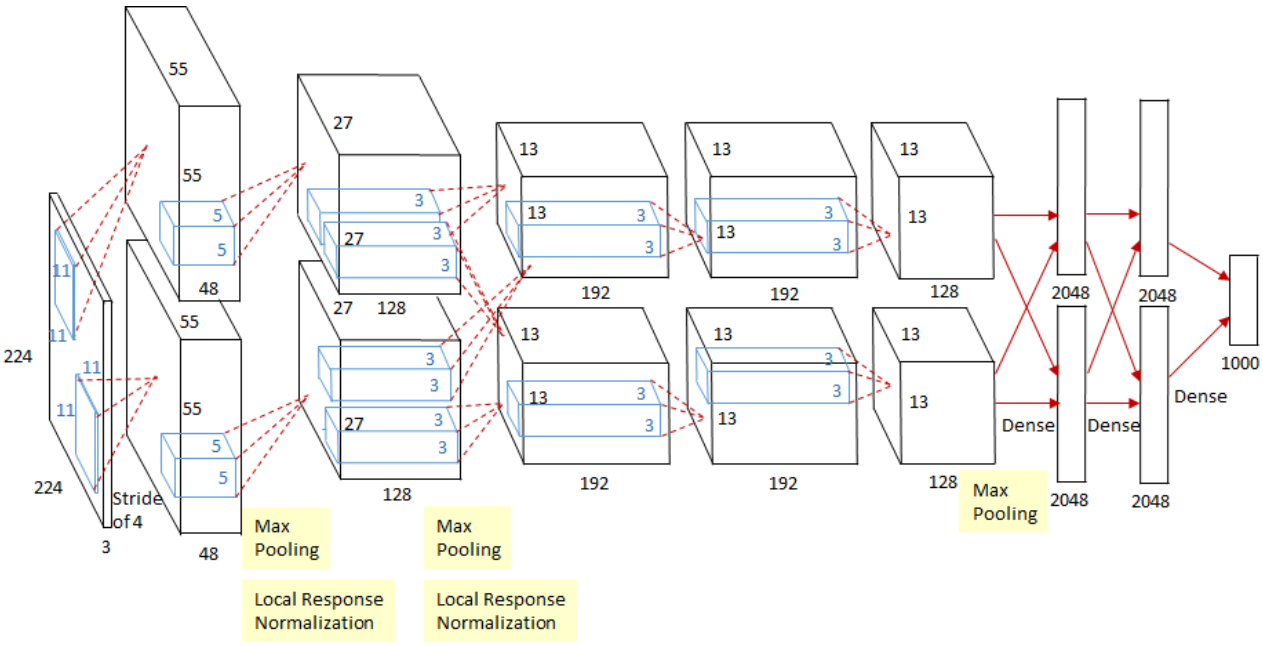

Рис. 8. Схема сети AlexNet
Fig. 8. AlexNet Network diagram

начал проявляться активный интерес к Deep Learning [17]. Эта сеть была разработана А. Крыжевским совместно с И. Суцкевером и Дж. Хинтоном. Необходимо отметить, что и до AlexNet были сети на GPU (к примеру, сеть от К. Челлапиллы в 2006 г.). Сеть AlexNet на наборе IMAGENET показала ошибку примерно 15 %.

Сравнение и описание устройства сетей LeNet и AlexNet приведено на рис. 9.

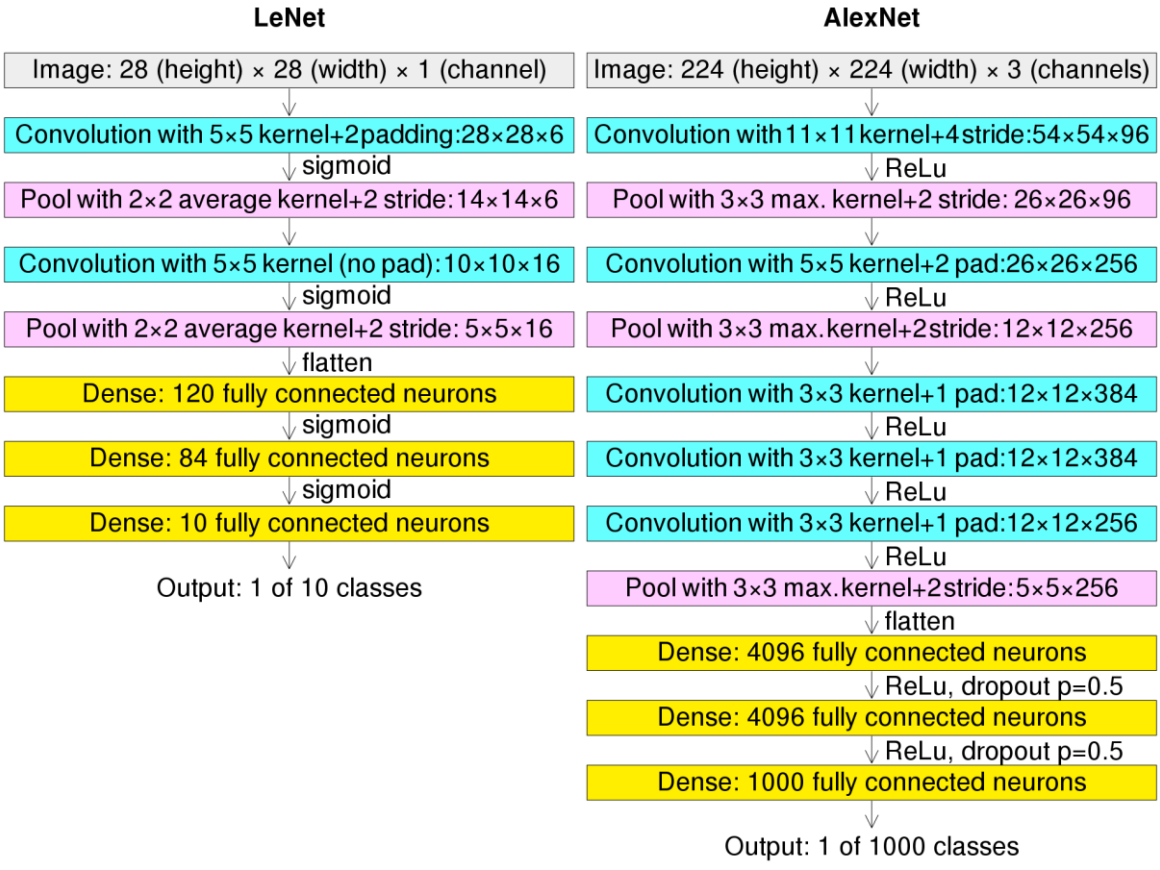

Рис. 9. Сравнительный анализ сетей LeNet и AlexNet
Fig. 9. Comparative analysis of LeNet and AlexNet networks

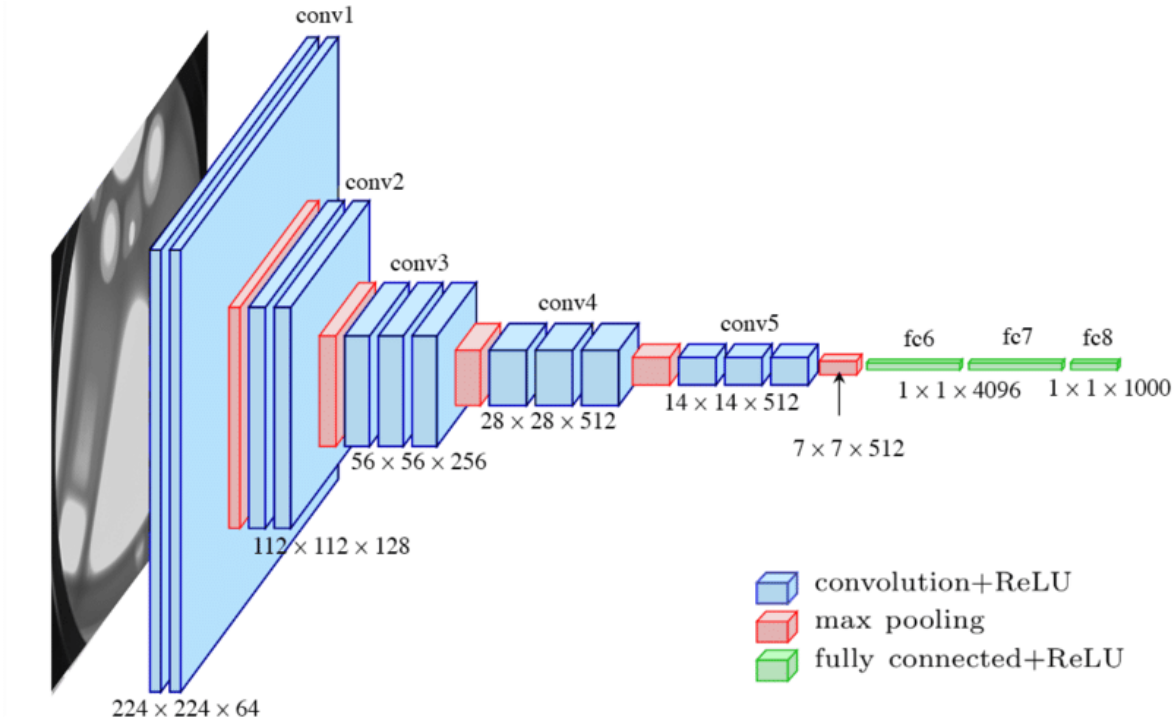

Рис. 10. Схема сети VGG
Fig. 10. VGG network diagram

Одним из важных моментов в истории сверточных нейросетей является архитектура VGG от 2014 года. Схема этой сети приведена на рис. 10, 11.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Рис. 11. Описание схемы сети VGG
Fig. 11. Description of the VGG network diagram

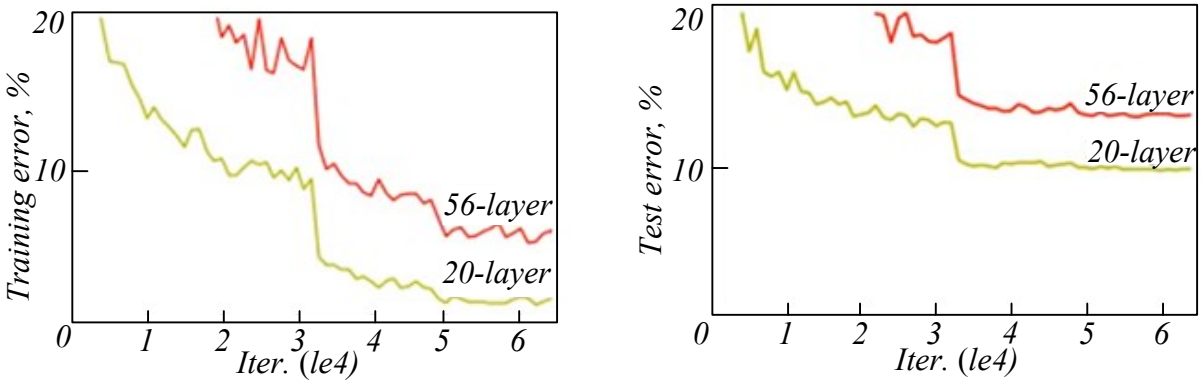

Рис. 12. График сравнения эффективности 20- и 56-слойных сетей
Fig. 12. Graph comparing the efficiency of 20- and 56-layer networks

Было предложено несколько вариантов VGG (рис. 11), которые различаются количеством весов, в некоторых случаях количеством слоев и ядром свертки. Всего было около 140 000 000 параметров. Основной причиной пристального внимания служит то, что все слои очень схожи, почти как «кирпичи», что позволяет использовать снова и снова те же блоки в других моделях. При этом архитектура является очень простой в освоении и понимании, что также дает большие возможности для модификации под свои нужды [17]. Сеть VGG показала ошибку на наборе IMAGENET примерно 6 %.

Важной точкой является ResNet (Residual Connections Network) 2015 года. Одна из основных проблем сети VGG и подобных ей сетей: чем дальше, чем более глубокие сети применяются (с большим количеством слоев), тем большая выразительная емкость у этих сетей. То есть, они начали распознавать все более и более высокоуровневые признаки. Однако выяснилось, что чем больше сеть (например, в 56 слоев, как приведено на рис. 11), тем хуже она тренируется. Это не было связано напрямую с переобучением (сеть показывает удовлетворительный результат на тренировочном наборе, но плохой результат на валидационном) – такие сети тренируются плохо. Установлено, что 56-слойная сеть тренируется хуже, чем 20-слойная, хотя, казалось бы, чем больше выразительная сила, тем лучше должна тренироваться сеть. Проблема состояла в том, что в начале тренировки все слои весов заполняются случайными числами [18]. И если возникает ситуация, что какой-то из слоев не успел натренироваться, то он испортит показатели всем слоям, следующим за ним. Более того, во время обратного прохода от него пойдут некорректные градиенты в дальнейшие слои, что еще сильнее замедлит обучение. И чем больше слоев в сети, тем больше вероятность возникновения такого результата, из-за чего сети с большим количеством слоев показывают себя хуже, как это видно на рис. 12.

В ходе решения этой проблемы возникла идея: необходимо упрощать тренировку. Было предложено не обучать каждый слой с нуля, а все, что пошло на вход слоя, передавать дальше (рис. 13). Единственным условием является возможность скорректировать данные значения: то есть слой предсказывает не напрямую выход, а его поправку (residual). Отсюда и название ResNet [18]. Слои в данных сетях вычисляют не все изображение на выходе, они отдают не весь выход. Вход «течёт» на выход, а слой может его лишь поправить. Если раньше на входе был x, а на выходе $F(x)$, то теперь на входе x, а на выходе $F(x) + x$.

Оказалось, что такое изменение дает возможность обучать куда более глубокие сети, чем раньше. Обычно (рис. 14) берется архитектура VGG (нижняя схема), в нее добавляется множество сверточных слоев (средняя схема) и после этого еще добавляются residual connections (верхняя схема).

В итоге на практике оказалось возможным тренировать сети глубиной до сотни слоев и больше. Была даже попытка обучить сеть глубиной в 1000 слоев, и все равно в итоге сеть смогла обучиться, пусть и эффективность данной модели получилась не слишком высокой [18].

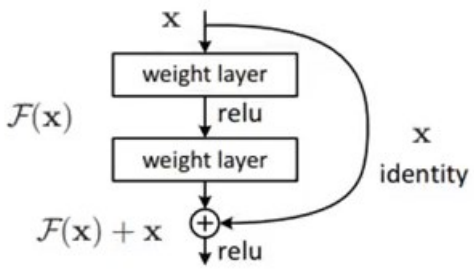

Рис. 13. Схема компонента ResNet
Fig. 13. ResNet Component Diagram

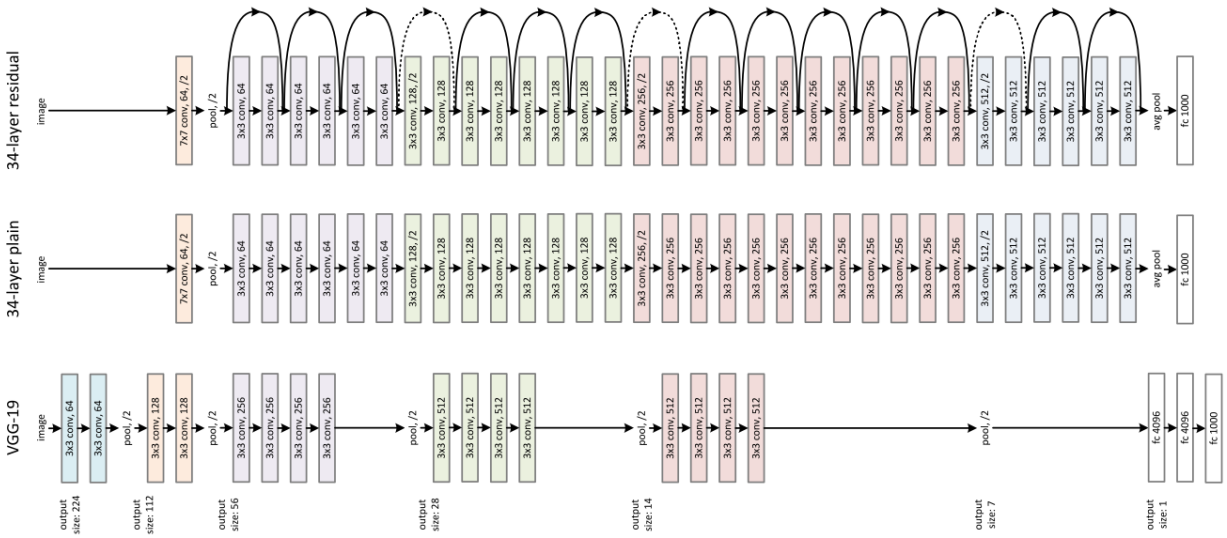

Рис. 14. Схема применения ResNet к VGG
Fig. 14. Diagram of applying ResNet to VGG

Сравнительный анализ сетей и дополнительные способы повышения их эффективности

На рис. 15 приведены все упомянутые архитектуры с их подвидами. По горизонтальной шкале показано, насколько они вычислительно затратны; по вертикали приведен лучший результат; размер круга соответствует количеству входных параметров.

Важным моментом является подготовка изображений для обучения и использование различных полезных приемов. В ходе решения задачи может возникнуть такая ситуация, когда для обучения дано слишком малое количество изображений (10 – 100). Обучать с нуля на таком количестве исходных данных невозможно (будет огромное переобучение). Для решения такой проблемы используется Transfer Learning: берется уже натренированная на схожей задаче сеть, все ее слои «замораживаются» (веса делаются неизменными) за исключением последнего. Этот последний слой, на котором и происходит выдача нейроном результата, меняется и обучается на желаемом наборе с использованием уже готовых весов замороженных слоев. То есть до этого сеть училась извлекать признаки из данных, что отражено в замороженных слоях, а теперь сеть должна научиться интерпретировать эти признаки [19, 20].

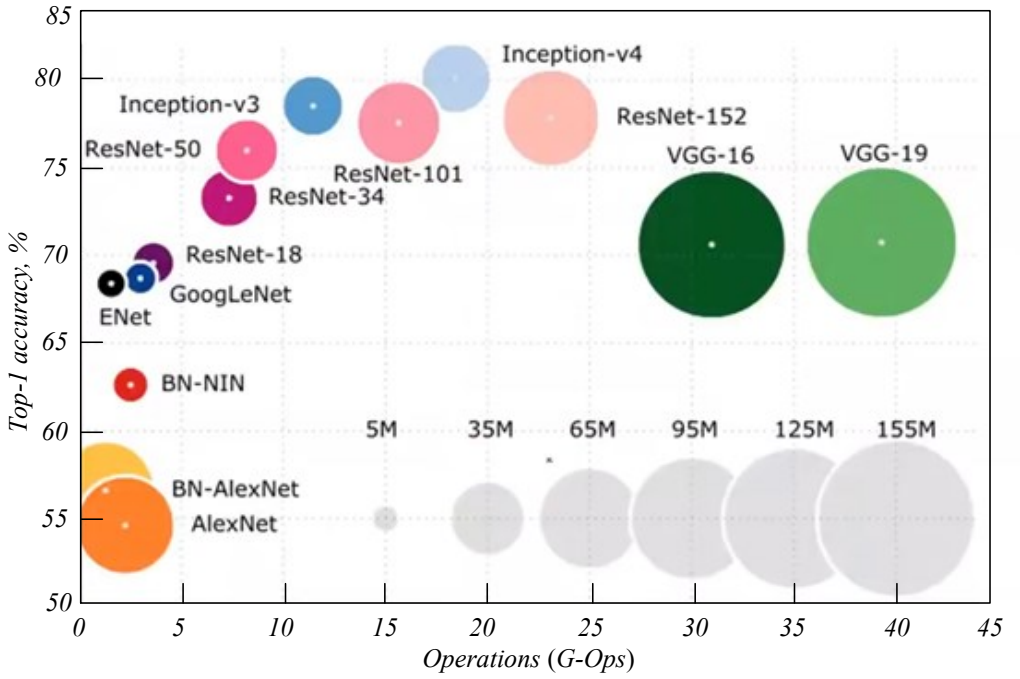

Рис. 15. Сравнительный график эффективности различных сетей
Fig. 15. Comparative graph of the efficiency of various networks

Иногда бывают ситуации, когда данных чуть больше (примерно 1000 и больше). В таких случаях можно не замораживать всю сеть. Замораживаются первые слои, а последующие вполне обучаются, причем эти слои обучаются с разной скоростью обучения. Скажем, одной из тактик является применение весов, близких к последнему слою, с коэффициентом 0,1, а веса, лежащие между замороженными слоями и уменьшенными в 0,1 раз, применяются с коэффициентом 0,01. Точные цифры и соотношение заморозки слоев определяются экспериментально в зависимости от задачи.

Основная причина, почему Transfer Learning и заморозка слоев работает, состоит в том, что по определению сверточной сети первые слои отвечают за распознавание самых базовых признаков: черточек, точек, прочих примитивных фигур [19]. Эти примитивы встречаются почти в любом изображении.

Выводы

Рассмотрено текущее состояние области задач искусственного интеллекта. Особенность указанных задач состоит в том, что, как только они будут решены, они переходят в разряд задач обычного вычисления.

Приведено определение машинного обучения в целом и «обучения с учителем» в частности. Описана типовая задача распознавания образов.

Приведено определение сверточных нейронных сетей. Описана их структура и основные отличия от обычного перцептрона: наличие сверточных слоев и pooling-слоев.

Показана история развития сверточных нейросетей, их эволюция. Описаны методы, которые были использованы для повышения эффективности работы сетей, проведен сравнительный анализ сетей разной архитектуры.

Описана концепция Transfer Learning, которая позволяет использовать уже обученные на одной задаче сети для решения других задач.

СПИСОК ЛИТЕРАТУРЫ

1. Комашинский В.И. Нейронные сети и их применение в системах управления и связи. Москва: Горячая линия-Телеком, 2002. 94 с.
2. Киселева Т.В., Маслова Е.В., Бычков А.Г. Машинное обучение для решения задач распознавания образов. В кн.: Сборник трудов 50-ой Международной конференции по информационным технологиям в науке, образовании и управлении. Гурзуф, 2021. С. 19–24.
3. Киселева Т.В., Маслова Е.В., Бычков А.Г. Машинное обучение в задачах распознавания изображений // Информатизация и связь. 2021. № 8. С. 15–19.
4. Бычков А.Г., Киселева Т.В., Маслова Е.В. Методика оптимизации элементов нейронной сети на примере перцептрона // Системы управления и информационные технологии. 2022. № 1 (87). С. 4–8. *https://doi.org/10.36622/VSTU.2022.88.1.001/*
5. Червяков Н.И. Применение нейронных сетей для задач прогнозирования и проблемы идентификации моделей прогнозирования // Нейрокомпьютеры: разработка и применение. 2003. № 10. С. 11–14.
6. Круглов В.В. Искусственные нейронные сети. Теория и практика. Москва: Горячая линия-Телеком, 2001. 382 с.
7. Хайкин С. Нейронные сети. Полный курс / Пер. с англ. Москва: ИД «Вильямс», 2006. 1104 с.
8. Николенко С. Глубокое обучение. Санкт-Петербург: Питер, 2018. 480 с.
9. Бычков А.Г., Киселева Т.В., Маслова Е.В. Использование сегментации в сверточных нейронных сетях для повышения точности // Системы управления и информационные технологии. 2022. № 3 (89). С. 7–10. *https://doi.org/10.36622/VSTU.2022.88.3.003/*
10. LeCun Y., Boser B., Denker J.S., Henderson D., Howard R.E., Hubbard W., Jackel L.D. Backpropagation applied to handwritten zip code recognition // Neural Computation. 1989. Vol. 1 (4). P. 541–551.
11. LeCun Y. Generalization and network design strategies. In: Technical Report CRG-TR-89-4 Department of Computer Science, University of Toronto, 1989. P. 1276–1289.
12. LeCun Y., Boser B., Denker J.S., Henderson D., Howard R.E., Hubbard W., Jackel L.D. Handwritten digit recognition with a back-propagation network. In: Advances in Neural Information Processing Systems 2 (NIPS 89). 1990. P. 1008–1019.
13. LeCun Y., Bottou L., Bengio Y., Haffner P. Gradient-based learning applied to document recognition // Proceedings of the IEEE. 1998. Vol. 86 (11). P. 2278–2324.
14. Chellapilla K., Puri S., Simard P. High Performance Convolutional Neural Networks for Document Processing. In: Tenth International Workshop on Frontiers in Handwriting Recognition. Guy L. ed. Suvisoft. 2006. P. 2877–2889.
15. Krizhevsky A., Sutskever I., Hinton G.E. Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. 2012. P. 1097–1105.

16. Shelhammer L.E., Darrell T. Fully convolutional networks for semantic segmentation. In: The IEEE Conf. On Computer Vision and Pattern Recognition (CVPR). 2015. P. 3431–3440.

17. Badrinarayanan V., Kendall A., Cipolla R. SegNet: A deep convolutional encoder-decoder architecture for image segmentation // ArXiv. 2015. Vol. 1511. Article 00561. *https://doi.org/10.48550/arXiv.1511.00561/*

18. Ronneberger O., Fischer P., Brox T. U-net: convolutional networks for biomedical image segmentation. In: Proc. Med. Image Comput. Comput.-Assisted Intevention. 2015. P. 234–241.

19. Iglovikov V., Shvets A. TernausNet: U-net with vgg11 encoder pre-trained on imagenet for image segmentation // ArXiv. 2018. Vol. 1801. Article 05746.

20. Santos J., Ferro E., Orozco J., Cayssials R. A heuristic approach to the multitask-multiprocessor assignment problem using the empty-slots method and rate monotonic scheduling // J. of Real-Time Systems. 1997. Vol. 13 (2). P. 167–199.

REFERENCES

1. Komashinskii V.I. *Neural networks and their application in control and communication systems.* Moscow: Hotline-Telecom, 2002, 94 p. (In Russ).

2. Kiseleva T.V., Maslova E.V., Bychkov A.G. Machine learning for solving image recognition problems. In: *Proceedings of the 50th International Conference on Information Technologies in Science*, Education and Management. Gurzuf, 2021, pp. 19–24. (In Russ).

3. Kiseleva T.V., Maslova E.V., Bychkov A.G. Machine learning in image recognition tasks. *Informatting and communication*. 2021, no. 8, pp. 15–19. (In Russ).

4. Bychkov A.G., Kiseleva T.V., Maslova E.V. Methods of optimization of neural network elements by the example of a perceptron. *Management systems and information technologies*. 2022, no. 1 (87), pp. 4–8. (In Russ). *https: //doi.org/10.36622/VSTU.2022.88.1.001/*

5. Chervyakov N.I. Application of neural networks for forecasting tasks and problems of identification of forecasting models. *Neurocomputers: development and application*. 2003, no. 10, pp. 11–14. (In Russ).

6. Kruglov V.V. *Artificial neural networks. Theory and practice.* Moscow: Hotline-Telecom, 2001, 382 p. (In Russ).

7. Khaikin S. *Neural networks. Full course*. Moscow: Williams Publishing House, 2006, 1104 p. (In Russ).

8. Nikolenko S. *Deep learning. Sankt-Peterburg: Piter,* 2018, 480 p. (In Russ).

9. Bychkov A.G., Kiseleva T.V., Maslova E.V. Using segmentation in convolutional neural networks to improve accuracy. *Management systems and information technologies*. 2022, no. 3 (89), pp. 7–10. (In Russ). *https: //doi.org/10.36622/VSTU.2022.88.3.003/*

10. LeCun Y., Boser B., Denker J.S., Henderson D., Howard R.E., Hubbard W., Jackel L.D. Backpropagation applied to handwritten zip code recognition. *Neural Computation*. 1989, vol. 1 (4), pp. 541–551.

11. LeCun Y. Generalization and network design strategies. In: *Technical Report CRG-TR-89-4 Department of Computer Science*, University of Toronto, 1989, pp. 1276–1289.

12. LeCun Y., Boser B., Denker J.S., Henderson D., Howard R.E., Hubbard W., Jackel L.D. Handwritten digit recognition with a back-propagation network. In: *Advances in Neural Information Processing Systems 2* (*NIPS 89*). 1990, pp. 1008–1019.

13. LeCun Y., Bottou L., Bengio Y., Haffner P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE.* 1998, vol. 86 (11), pp. 2278–2324.

14. Chellapilla K., Puri S., Simard P. High Performance Convolutional Neural Networks for Document Processing. In: *Tenth International Workshop on Frontiers in Handwriting Recognition*. Guy L. ed. Suvisoft. 2006, pp. 2877–2889.

15. Krizhevsky A., Sutskever I., Hinton G.E. Imagenet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

16. Shelhammer L.E., Darrell T. Fully convolutional networks for semantic segmentation. In: *The IEEE Conf. On Computer Vision and Pattern Recognition* (*CVPR*). 2015, pp. 3431–3440.

17. Badrinarayanan V., Kendall A., Cipolla R. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *ArXiv.* 2015, vol. 1511, article 00561. *https://doi.org/10.48550/arXiv.1511.00561/*

18. Ronneberger O., Fischer P., Brox T. U-net: convolutional networks for biomedical image segmentation. In: *Proc. Med. Image Comput. Comput.-Assisted Intevention.* 2015, pp. 234–241.

19. Iglovikov V., Shvets A. TernausNet: U-net with vgg11 encoder pre-trained on imagenet for image segmentation. *ArXiv.* 2018, vol. 1801, article 05746.

20. Santos J., Ferro E., Orozco J., Cayssials R. A heuristic approach to the multitask-multiprocessor assignment problem using the empty-slots method and rate monotonic sched-

uling. *J. of Real-Time Systems.* 1997, vol. 13 (2), pp. 167–199.

Сведения об авторах
Александр Григорьевич Бычков*, аспирант кафедры прикладных информационных технологий и программирования,* Сибирский государственный индустриальный университет
ORCID*:* 0000-0003-0258-4807
E-mail*:* aleksds1@yandex.ru

Тамара Васильевна Киселёва*, д.т.н., профессор кафедры прикладных информационных технологий и программирования,* Сибирский государственный индустриальный университет
ORCID*:* 0000-0002-1120-029X
E-mail*:* kis@siu.sibsiu.ru

Елена Владимировна Маслова*, к.т.н., доцент кафедры прикладных информационных технологий и программирования,* Сибирский государственный индустриальный университет
ORCID*:* 0000-0001-9697-523X
E-mail*:* Elenamaslova1805@yandex.ru

Information about the authors
Alexander G. Bychkov*, Postgraduate of Department of Applied Information Technologies and Programming,* Siberian State Industrial University
ORCID*:* 0000-0003-0258-4807
E-mail*:* aleksds1@yandex.ru

Tamara V. Kiseleva*, Dr. Sci. (Eng.), Prof. of the Department of Applied Information Technologies and Programming,* Siberian State Industrial University
ORCID*:* 0000-0002-1120-029X
E-mail*:* kis@siu.sibsiu.ru

Elena V. Maslova*,* Cand. Sci. (Eng.), Assist. Prof. *of the Department of Applied Information Technologies and Programming,* Siberian State Industrial University
ORCID*:* 0000-0001-9697-523X
E-mail*:* Elenamaslova1805@yandex.ru

Авторы заявляют об отсутствии конфликта интересов.
The authors declare that there is no conflict of interest.

Поступила в редакцию 21.07.2022
После доработки 09.02.2023
Принята к публикации 13.02.2022

Received 21.07.2022
Revised 09.02.2023
Accepted 13.02.2022