УДК 004.032.26 + 59.089

DOI: 10.33396/1728-0869-2021-5-53-64

ИНТЕЛЛЕКТУАЛЬНЫЕ МЕТОДЫ АНАЛИЗА ДАННЫХ В БИОМЕДИЦИНСКИХ ИССЛЕДОВАНИЯХ: СВЕРТОЧНЫЕ НЕЙРОННЫЕ СЕТИ

© 2021 г. **А. Н. Наркевич, К. А. Виноградов, К. М. Параскевопуло, Т. Х. Мамедов**

ФГБОУ ВО «Красноярский государственный медицинский университет им. проф. В. Ф. Войно-Ясенецкого», г. Красноярск

Одним из современных средств, позволяющих в медицинских исследованиях анализировать и распознавать не просто набор данных об исследуемых объектах или пациентах, а использовать в качестве изучаемого объекта изображение, являются сверточные нейронные сети. Ввиду значительной роли визуальных методов диагностики при оказании медицинской помощи использование интеллектуального распознавания результатов данных методов приобретает существенную актуальность. На текущий момент сверточные нейронные сети получили достаточно широкое распространение в работах, посвященных повышению качества диагностики в различных сферах медицины. Однако достаточно сложные математический аппарат, на основе которого функционируют сверточные нейронные сети, и инструменты их построения не позволяют широко внедрить данные модели в исследовательскую медицинскую практику. Целью данной статьи является представление методологии и возможностей применения сверточных нейронных сетей в медицинских исследованиях, а также представление примера построения сверточной нейронной сети для решения задачи классификации медицинских изображений. В статье приведены методологические основы функционирования сверточных нейронных сетей, описание набора данных для построения таких моделей, пример построения модели сверточной нейронной сети для решения задачи классификации дерматоскопических изображений с применением библиотек tensorflow и keras на языке python, а также рекомендации по представлению результатов построения сверточных нейронных сетей.

Ключевые слова: Интеллектуальный анализ данных, сверточные нейронные сети, tensorflow, keras, python, математическое моделирование, isic-archive

INTELLIGENT DATA ANALYSIS IN BIOMEDICAL RESEARCH: CONVOLUTIONAL ARTIFICIAL NEURAL NETWORKS

A. N. Narkevich, K. A. Vinogradov, K. M. Paraskevopulo, T. H. Mamedov

Voyno-Yasenetsky Krasnoyarsk State Medical University, Krasnoyarsk, Russia

Convolutional neural networks are one of the modern tools that allow medical research to analyze and recognize not just a set of data about the objects under study or patients, but to use an image as the object under study. Due to the significant role of visual diagnostic methods in the provision of medical care, the use of intelligent recognition of the results of these methods becomes essential. At the moment, convolutional neural networks become widespread in research on the quality of diagnostics in various fields of medicine. However, complex mathematical apparatus behind convolutional neural networks function, and the tools for their construction limit implementation of these models into medical research and practice. This paper provides a gentle introduction to the methodology and application possibilities of convolutional neural networks in medical research. In this paper the reader will find methodological foundations behind convolutional neural networks, a description of a data set for building such models, an example of construction of a convolutional neural network model for classification of dermatoscopic images using TensorFlow and Keras libraries in Python as well as recommendations on how to present the results of building convolutional neural networks.

Key word: Intelligent data analysis, convolutional neural networks, TensorFlow, Keras, Python, mathematical modeling, isic-archive

Библиографическая ссылка:

Наркевич А. Н., Виноградов К. А., Параскевопуло К. М., Мамедов Т. Х. Интеллектуальные методы анализа данных в биомедицинских исследованиях: сверточные нейронные сети // Экология человека. 2021. № 5. С. 53–64.

For citing:

Narkevich A. N., Vinogradov K. A., Paraskevopulo K. M., Mamedov T. H. Intelligent Data Analysis in Biomedical Research: Convolutional Artificial Neural Networks. *Ekologiya cheloveka (Human Ecology)*. 2021, 5, pp. 53-64.

В современных медицинских исследованиях все большую распространенность получают интеллектуальные методы анализа данных, автоматизированные средства распознавания объектов, а также математические алгоритмы, требующие минимального участия исследователя [4, 9, 12, 15, 16]. Одним из таких средств, позволяющих анализировать и распознавать не просто набор данных об исследуемых объектах, а использовать в качестве изучаемого объекта изображение, являются сверточные нейронные сети [2, 8, 13, 26]. Ввиду значительной роли визуальных методов диагностики при оказании медицинской помощи использование интеллектуального распознавания результатов данных методов приобретает существенную актуальность [3, 5, 10, 14, 24].

На текущий момент сверточные нейронные сети получили достаточно широкое распространение в работах, посвященных повышению качества диагностики в области онкологии [7, 17], рентгенологии [6, 21], гистологии [11, 23] и других областях медицины [1,

25] путем классификации медицинских изображений. Однако достаточно сложные математический аппарат, на основе которого функционируют сверточные нейронные сети, и инструменты их построения [22, 27] не позволяют широко внедрить данные модели в исследовательскую медицинскую практику.

Задача классификации при применении сверточных нейронных сетей возникает перед исследователем, когда необходима разработка автоматизированных или полуавтоматизированных инструментов, которые позволяют с минимальным участием человека отнести медицинское изображение к одному из классов, например, «есть патология на рентгенограмме» или «на рентгенограмме нет патологии», «злокачественное новообразование» или «доброкачественное новообразование».

Целью данной статьи является представление методологии и возможностей применения сверточных нейронных сетей в медицинских исследованиях, а также представление примера построения сверточной нейронной сети для решения задачи классификации медицинских изображений.

Методологические основы функционирования сверточных нейронных сетей

Если математическая модель классической искусственной нейронной сети это попытка построить математический аналог головного мозга и математически имитировать передачу нервного импульса между нейронами, то модель сверточной нейронной сети это попытка математически имитировать передачу визуального нервного импульса между слоями коры головного мозга [19]. Под визуальным нервным импульсом подразумевается цифровое изображение. Сверточные нейронные сети в большей мере используются при анализе и классификации именно цифровых изображений. В связи с этим структурной единицей сверточной нейронной сети является математический нейронный слой.

Сверточная нейронная сеть является комбинацией различных слоев. В сверточных нейронных сетях чаще всего используются сверточные, максимизирующие, усредняющие и полносвязные слои. Сверточный слой представляет собой набор двумерных ядер, которые «пробегают» по входному или полученному на предыдущем слое изображению, путем чего осуществляется его «свертка». На рис. 1 представлен пример двумерного ядра размером 3×3.

0,25	0,81	0,05
0,62	0,36	0,58
0,99	1,12	0,13

Рис. 1. Пример ядра для свертки размером 3×3

Каждое ядро является двумерной матрицей коэффициентов, на которые перемножаются значения пикселей входного изображения. До обучения сверточной нейронной сети коэффициенты ядра носят случайный характер, а в процессе обучения приобретают значения, позволяющие оценивать характерные признаки изображений, которые, по «мнению» сверточной нейронной сети, наиболее важны для распознавания подаваемых на вход цифровых изображений. Иллюстрация функционирования сверточного слоя представлена на рис. 2.

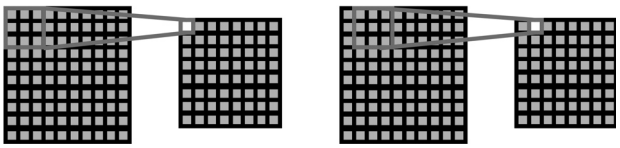


Рис. 2. Иллюстрация функционирования сверточного слоя

Как видно на рис. 2, с использованием ядра 3×3 путем перемножения ее коэффициентов на значения девяти пикселей изображения, образующих квадрат 3×3, производится свертка данного участка цифрового изображения. Таким образом, девять пикселей входного изображения преобразуются в один пиксель выходного. В связи с этим, как правило, размер выходного изображения меньше, чем входного. На следующем этапе осуществляется смещение ядра относительно входного изображения и осуществляется свертка следующего участка. Необходимо отметить, что смещение ядра может производиться не на один пиксель, а может регулироваться разработчиком.

Важным моментом является то, что при функционировании глубоких сверточных нейронных сетей все элементы выходных изображений преобразуются дополнительно с помощью функции активации. Наибольшее распространение среди функций активации в сверточных нейронных сетях получила функция ReLU (rectified linear unit). Данная функция обнуляет все отрицательные значения. Помимо функции ReLU могут быть использованы такие функции, как ступенчатая, сигмоидная или линейная, гиперболический тангенс, а также многочисленные их модификации [18, 20].

Как уже было отмечено выше, сверточный слой – набор различных ядер. Размер ядер может варьировать, но в рамках одного сверточного слоя, как правило, выбирается одинаковый размер всех входящих в слой ядер. Таким образом, на выходе из сверточного слоя формируется набор изображений, каждое из которых является сверткой входного изображения определенным ядром – чем больше ядер включает в себя сверточный слой, тем больше формируется изображений на выходе из него.

Максимизирующий и усредняющий слои функционируют практически по тому же принципу, что и сверточный. Важным отличием является непосредственное преобразование значений входного изображения. Если в сверточном слое каждый пиксель участка изображения перемножается на определенный коэффициент, то

в максимизирующем слое выбирается максимальное значение из всех, а в усредняющем определяется их среднее арифметическое. На рис. 3 приведены результаты применения максимизирующего слоя, а на рис. 4 – усредняющего.

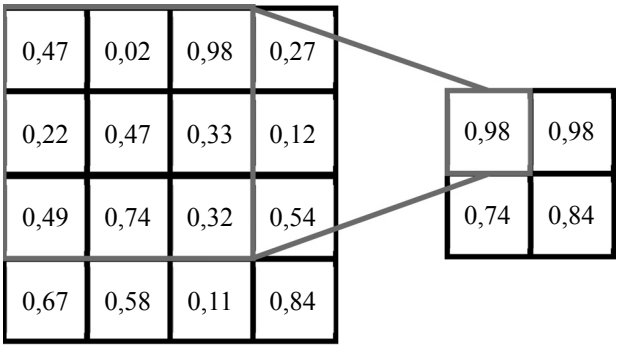


Рис. 3. Результаты применения максимизирующего слоя

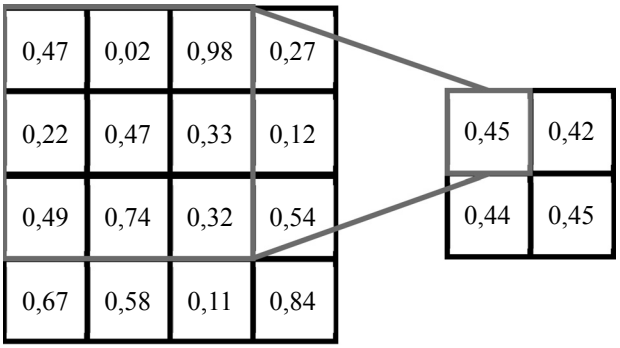


Рис. 4. Результаты применения усредняющего слоя

Как видно из представленного на рис. 3 примера, из девяти значений пикселей входного изображения выбрано максимальное (0,98), которое является результатом и присваивается значению в выходном изображении. При применении усредняющего слоя из этих же девяти значений получено среднее арифметическое (0,45).

Таким образом, максимизирующие и усредняющие слои не меняют количество изображений, а лишь уменьшают их размеры, то есть если на входе максимизирующего или усредняющего слоев имеется 100 изображений, полученных путем прохождения изначального изображения через другие слои, то и на выходе будет получено 100 изображений, но меньшего размера. Так, данные слои позволяют исключить менее важную информацию для классификации первоначального изображения. Необходимо отметить, что наибольшую распространенность в сверточных нейронных сетях получил именно максимизирующий слой.

Основой структуры сверточной нейронной сети является комбинация сверточных и максимизирующих или усредняющих слоев для постепенного преобразования входного изображения так, чтобы исключить все незначимые для классификации элементы и сконцентрировать те элементы, на основе которых будет приниматься окончательное «решение» о принадлежности изображения к одному из классов. Однако сами по себе данные слои не позволяют отнести изображение к одному из классов. Как правило, в конце сверточной нейронной сети все двумерные изображения, которые были получены в ходе преобразования, векторизуются, то есть двумерные матрицы преобразуются в одномерный вектор значений (рис. 5).

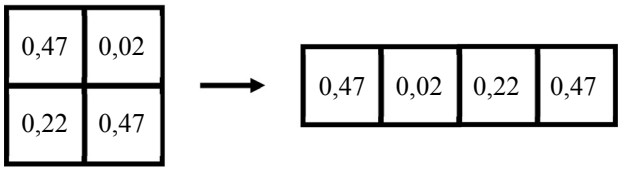


Рис. 5. Пример векторизации двумерного изображения в одномерное

После векторизации данные передаются в один или несколько полносвязных слоев, на выходе из которых и принимается окончательное «решение» сверточной нейронной сети, к какому из классов относится изображение. В наборе полносвязных слоев каждый нейрон предыдущего слоя связан с помощью синапсов с каждым нейроном следующего слоя (рис. 6).

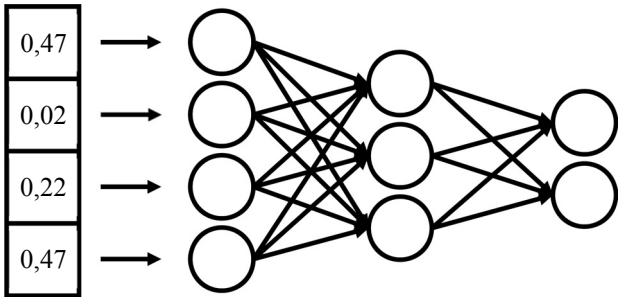


Рис. 6. Пример распространения векторизованных значений по полносвязным слоям

На первом шаге значения подаются на нейроны первого полносвязного слоя нейронной сети. После этого все значения передаются на все нейроны второго полносвязного слоя. То есть значение, поступившее на первый нейрон первого слоя, далее передается на каждый нейрон второго слоя (на рис. 6 на 1, 2 и 3 нейроны второго слоя). Признак со второго нейрона первого слоя также передается на все нейроны второго слоя и так далее.

После того как все значения с первого полносвязного слоя поступили на нейроны второго полносвязного слоя, начинается функционирование классического математического нейрона. Функционирование классического математического нейрона (например, первого нейрона второго слоя) заключается в проведении нескольких математических операций:

1. По входным синапсам первого нейрона второго слоя поступают значения со всех нейронов первого слоя.

2. Каждый синапс имеет свой вес, на который умножается значение, приходящее по этому синапсу.

Под весом синапса понимается обычный математический коэффициент.

3. Перемноженные на соответствующие коэффициенты значения со всех синапсов нейрона передаются на сумматор, где суммируются в одно значение.

4. Значение с сумматора преобразуется с помощью одной из функций активации (сигмоидная функция, гиперболический тангенс, Softmax и т. д.). Преобразование значения осуществляется для того, чтобы полученное после сумматора значение (которое может быть довольно большим) привести к определенному диапазону.

5. После преобразования значение передается на все нейроны следующего слоя нейронной сети.

Если полносвязных слоев много, то такая процедура повторяется на каждом полносвязном слое и затем на выходном слое. Количество нейронов на выходном слое, как правило, равно числу классов, на которые исследователь предполагает классифицировать изображения. Так как классически таких классов два, то и, как правило, выходных нейронов также два. При этом выходное значение на первом нейроне выходного слоя характеризует степень принадлежности изображения к первому классу, а на втором нейроне – ко второму классу. На каком нейроне выходного слоя будет значение больше, к тому классу и относится изображение.

Таким образом, классическая структура сверточной нейронной сети представляет собой комбинацию нескольких сверточных и максимизирующих или усредняющих слоев с различными размерами ядер и шагов их смещения с включением в конце нескольких полносвязных слоев.

Представленная структура сверточной нейронной сети и ее процесс функционирования позволяют осуществлять постепенное преобразование поступившего на первый слой сети цифрового изображения в ответ, выражающийся в значениях на выходных нейронах последнего полносвязного слоя.

Важно учитывать, что сама по себе сконструированная сверточная нейронная сеть не сможет осуществлять верную классификацию имеющихся у исследователя цифровых изображений. Для этого она должна быть обучена на наборе изображений с заранее известными классами, для классификации на которые предполагается обучить сверточную нейронную сеть. Данный набор делится на две части – на обучающее множество (обычно составляет 70 % от первоначального набора изображений) и тестовое (обычно составляет 30 % от первоначального набора изображений). Обучающее множество используется для непосредственного обучения сверточной нейронной сети, а тестовое – для оценки качества классификации.

Обучение сверточной нейронной сети осуществляется, как правило, по следующему алгоритму. Первоначально сверточная нейронная сеть создается (инициализируется) со случайными значениями ядер сверточных слоев и случайными весами синапсов полносвязных слоев. Затем на первый слой сверточной нейронной сети подается первое изображение из обучающего множества. Это изображение проходит до последнего полносвязного слоя, на нейронах которого оценивается предполагаемый класс изображения. Если предполагаемый класс изображения совпадает с заранее известным классом, то осуществляется переход к следующему изображению. Если предполагаемый класс изображения не совпадает с заранее известным классом, то получившаяся ошибка распространяется обратно от последнего полносвязного слоя к первому слою сверточной нейронной сети, корректируя веса синапсов и коэффициенты ядер сверточных слоев так, чтобы класс на выходе сверточной нейронной сети совпал с заранее известным классом. Затем осуществляется переход к следующему изображению.

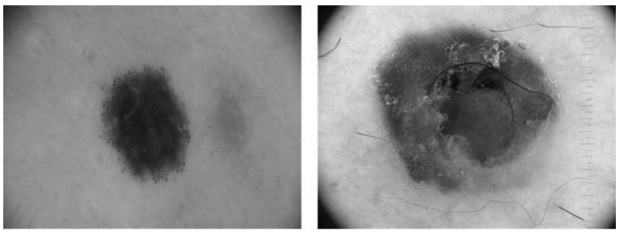
Таким образом осуществляется проход всех изображений обучающего множества по сверточной нейронной сети с попутной коррекцией весов синапсов и коэффициентов ядер сверточных слоев так, чтобы максимальное число изображений было классифицировано правильно. Так как для обучения сверточных нейронных сетей, как правило, используются довольно большие объемы изображений, обновление весов синапсов и коэффициентов ядер сверточных слоев осуществляется после прохода пакета изображений (тензора). Проход одного пакета изображений по сети называется эпохой. В зависимости от объема имеющихся изображений и структуры сверточной нейронной сети для ее обучения может понадобиться от одной до большого числа эпох.

После окончания обучения сверточной нейронной сети осуществляется оценка качества классификации с помощью изображений, входящих в тестовое множество. Для определения качества классификации используются показатели доли верно классифицированных изображений из тестового множества, а также показатели чувствительности, специфичности и точности.

Описание набора данных для построения сверточной нейронной сети

Примером набора данных для построения сверточной нейронной сети послужит набор дерматоскопических изображений образований кожи The International Skin Image Collaboration, расположенный на сайте https://www.isic-archive.com. Используемый нами набор данных состоял из 7 088 изображений, разделенных на 2 класса – изображения с доброкачественными образованиями кожи (5 670 изображений) и изображения со злокачественными образованиями кожи (1 418 изображений). На рис. 7 представлены примеры доброкачественного и злокачественного образований кожи.

Для использования представленных изображений для обучения сверточной нейронной сети они должны быть размещены на рабочем компьютере в папке с определенной структурой. Для начала создана папка



Доброкачественное образование | Злокачественное образование

Рис. 7. Примеры доброкачественного и злокачественного образований кожи

«Image», которая размещена в папке, имеющей путь «C:/Модель CNN/». В папке «Image» размещены две папки: «Train» и «Test». В обоих из этих папок созданы еще две папки: «d» и «z». В папке «Train» размещены изображения, которые будут использованы для обучения сверточной нейронной сети, а в папке «Test» – для тестирования. В папках «d» размещены изображения, на которых представлены доброкачественные образования кожи, а в папке «z» – злокачественные.

После размещения всех изображений по папкам на рабочем компьютере в папке «Train/d» находилось 3 906 изображений, в папке «Train/z» – 1 764, в папке «Test/d» – 975, а в папке «Test/z» – 443 изображения.

Дальнейшая иллюстрация построения сверточной нейронной сети будет осуществляться на представленных изображениях, размещенных указанным образом по папкам на рабочем компьютере.

Построение модели сверточной нейронной сети

Наиболее мощным аппаратом для построения и настройки сверточных нейронных сетей обладает язык программирования python с использованием дополнительных библиотек tensorflow (https://www.tensorflow.org) и keras (https://keras.io). При написании данной статьи использовались Python версии 3.8.6, библиотека tensorflow версии 2.3.1, библиотека keras версии 2.4.0. Построение сверточных нейронных сетей с применением библиотек tensorflow и keras осуществляется в несколько этапов. На первом этапе загружаются необходимые библиотеки и подключаются папки с изображениями, которые будут использоваться для построения сверточной нейронной сети (подготовительный этап). На следующем этапе настраивается предобработка изображений для их использования при построении сверточной нейронной сети (этап предобработки изображений). Дальнейший этап заключается в конструировании структуры сверточной нейронной сети и настройке ее обучения (этап инициализации). В последующем осуществляется непосредственное обучение инициализированной сети (этап обучения). На заключительном этапе оценивается качества полученной сверточной нейронной сети (этап оценки качества).

Как уже было отмечено выше, первым этапом построения сверточной нейронной сети является подготовительный этап. В рамках реализации данного этапа необходимо выполнить следующие манипуляции.

Если на рабочем компьютере установлена недостаточно «свежая» версия Python, необходимо импортировать обновленные функции, которые будут использоваться в процессе построения сверточной нейронной сети:

```
from __future__ import absolute_import, division, print_function, unicode_literals
```

Если при выполнении всего дальнейшего программного кода не будут возникать ошибки, то подключение обновленных функций не требуется, но для исключения данных ошибок предпочтительнее им воспользоваться.

Далее необходимо импортировать необходимые для построения сверточной нейронной сети модули:

```
import os
import matplotlib.pyplot as plt
import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import logging
from keras.utils import plot_model
```

После подключения всех необходимых модулей необходимо связать папки на рабочем компьютере, в которых расположены изображения, и переменные, которые будут представлять совокупность этих изображений в программном коде:

```
zip_dir = ‹C:/Модель CNN/›
zip_dir_base = os.path.dirname(zip_dir)
base_dir = os.path.join(os.path.dirname(zip_dir), ‘Image’)
train_dir = os.path.join(base_dir, ‘train’)
test_dir = os.path.join(base_dir, ‘test’)
train_d_dir = os.path.join(train_dir, ‘d’)
train_z_dir = os.path.join(train_dir, ‘z’)
test_d_dir = os.path.join(test_dir, ‘d’)
test_z_dir = os.path.join(test_dir, ‘z’)
```

После выполнения данного раздела программного кода к переменным train_d_dir и test_d_dir будут привязаны пути на рабочем компьютере к изображениям, которые представляют обучающую и тестовую выборки класса доброкачественных образований, а к переменным train_z_dir и test_z_dir – класса злокачественных образований.

Для того чтобы в процессе построения сверточной нейронной сети осуществлять контроль за числом изображений, которые расположены в целевых папках, и их правильным подключением, можно использовать следующий программный код:

```
num_d_tr = len(os.listdir(train_d_dir))
num_z_tr = len(os.listdir(train_z_dir))
num_d_test = len(os.listdir(test_d_dir))
num_z_test = len(os.listdir(test_z_dir))
total_train = num_d_tr + num_z_tr
total_test = num_d_test + num_z_test
print(‘Доброкачественных в обучающем наборе данных: ‘, num_dobro_tr)
```

print(‹Злокачественных в обучающем наборе данных: ‘, num_zlo_tr)

print(‹Доброкачественных в тестовом наборе данных: ‹, num_dobro_test)

print(‹Злокачественных в тестовом наборе данных: ‹, num_zlo_test)

print(‹—›)

print(‹Всего изображений в обучающем наборе данных: ‘, total_train)

print(‹Всего изображений в тестовом наборе данных: ‹, total_test)

Данный программный код позволит подсчитать число изображений, расположенных в каждой целевой папке и вывести данную информацию на экран. На этом подготовительный этап закончен, и может быть осуществлен переход к этапу настройки предобработки изображений.

При обучении сверточной нейронной сети проход изображений по сети осуществляется не по одному, а так называемыми тензорами (пакетами). Такое тензорное обучение сверточной нейронной сети приводит к тому, что обновление параметров сети осуществляется не после прохода каждого изображения, а после прохода всего тензора изображений. В связи с этим необходимо задать размер используемых тензоров.

Довольно часто возникает ситуация, когда в распоряжении исследователя имеются изображения разнообразного размера. Помимо этого изображения могут быть довольно больших размеров. Использование разнообразных по размеру изображений приведет к ошибкам при ее обучении, а большие размеры изображений приведут к существенному увеличению самой сети так, что ее обучение будет настолько длительным, что актуальность построения такой сверточной нейронной сети, скорее всего, исчезнет. Для того чтобы задать размер тензора и стандартизовать размеры изображений, необходимо использовать следующий программный код:

BATCH_SIZE_TRAIN = 32

BATCH_SIZE_TEST = 32

IMG_SHAPE = 224

Выполнение данного программного кода задаст размер тензора для обучающего и тестового наборов изображений, включающего 32 изображения, а размеры всех изображений будут приведены к значениям 224×224 пикселя.

Далее на этапе предобработки изображений устанавливаются дополнительные настройки предобработки. Прежде чем перейти к демонстрации программного кода установки дополнительных настроек предобработки изображений, необходимо отметить несколько достаточно важных моментов. Во-первых, каждый цветовой канал (R, G и B) всех пикселей изображений стандартно имеет диапазон значений от 0 до 255. Использование таких изображений неминуемо приведет к ошибкам в обучении сверточной нейронной сети. В связи с этим необходимо привести значения всех пикселей изображений к диапазону от 0 до 1 (нормализация изображений).

Во-вторых, при построении сверточных нейронных сетей, как и при построении классических нейронных сетей, существует так называемая проблема переобучения. Данная проблема заключается в том, что нейронная сеть достаточно хорошо распознает изображения, на которых она обучается, но практически неспособна распознавать изображения, не входящие в обучающий набор. Для контроля переобучения сверточной нейронной сети, как правило, из обучающего набора изображений выделяется часть, которая не участвует непосредственно в обучении, а используется для контроля способности распознавать изображения, на которых она не обучалась. Данная часть, как правило, называется валидационным набором.

В-третьих, использование для обучения сверточной нейронной сети ограниченного или стандартизованного набора изображений может привести к тому, что нейронная сеть обучится распознавать изображения, приведенные к определенному стандарту. При этом незначительное изменение поступающего на обученную таким образом сеть изображения приведет к его неправильной классификации. Для того чтобы избежать этого, могут быть использованы два наиболее распространенных подхода. Один заключается в том, что можно увеличить число изображений, которые входят в обучающий набор, тем самым представить сверточной нейронной сети при обучении довольно большое число разнообразных изображений. Данный подход можно считать весьма эффективным, но на практике его применение достаточно затруднительно ввиду того, что исследователь, как правило, ограничен числом имеющихся изображений по различным причинам. Второй подход заключается в том, что можно попытаться изменить различные характеристики имеющихся изображений для их разнообразия. Как правило, на практике наиболее подходящим является именно второй подход. Для его реализации необходимо использовать следующий программный код:

train_image_generator = ImageDataGenerator (rescale=1./255, validation_split=0.2, rotation_range=15, width_shift_range=40, height_shift_range=40, zoom_range=0.1, horizontal_flip=True, vertical_flip=True)

test_image_generator = ImageDataGenerator(rescale=1./255, rotation_range=15, width_shift_range=40, height_shift_range=40, zoom_range=0.1, horizontal_flip=True, vertical_flip=True)

Выполнение данного программного кода приведет к нормализации значения пикселей всех изображений (rescale=1./255), выделению из обучающего набора изображений валидационного набора в объеме 20 % изображений от числа изображений, входящих в обучающий набор (validation_split=0.2). Независимо от используемого подхода исследователя к увеличению разнообразия обучающих изображений (увеличение числа изображений или изменение их характеристик) данные два параметра функции ImageDataGenerator являются обязательными. Если исследователь при-

нимает решение об увеличении разнообразия изображений за счет увеличения их числа, то остальные параметры функции ImageDataGenerator могут быть упущены. Если же исследователь принимает решение об увеличении разнообразия изображений за счет изменения их характеристик, то это может быть реализовано за счет следующих параметров функции ImageDataGenerator. Может быть задан случайный поворот изображения на заданное число градусов (rotation_range=15), случайный сдвиг по горизонтали на заданное число пикселей (width_shift_range=40) и по вертикали (height_shift_range=40), случайное увеличение изображения на заданный процент (zoom_range=0.1 – случайное увеличение от 0 до 10 %), случайное зеркальное отражение изображения относительно горизонтальной (horizontal_flip=True) и вертикальной (vertical_flip=True) оси.

После установки дополнительных настроек предобработки изображений осуществляется формирование тензоров (пакетов изображений) с учетом следующих настроек:

```
train_data_gen = train_image_generator.flow_from_directory(directory=train_dir, target_size=(IMG_SHAPE, IMG_SHAPE), batch_size=BATCH_SIZE_TRAIN, shuffle=True, class_mode=›binary›, subset=›training›)
val_data_gen = train_image_generator.flow_from_directory(directory=train_dir, target_size=(IMG_SHAPE,IMG_SHAPE), batch_size=BATCH_SIZE_TRAIN, shuffle=True, class_mode=›binary›, subset=›validation›)
test_data_gen = test_image_generator.flow_from_directory(directory=test_dir, target_size=(IMG_SHAPE,IMG_SHAPE), batch_size=BATCH_SIZE_TEST, shuffle=False, class_mode=›binary›)
```

После выполнения данного программного кода будут сгенерированы три набора тензоров (для обучения – train_data_gen, для контроля переобучения – val_data_gen и тестирования – test_data_gen) из изображений, расположенных в соответствующих папках на рабочем компьютере (directory), которые будут приведены к указанному размеру (target_size). Каждый тензор будет включать в себя заранее указанное число изображений (batch_size). Помимо этого может быть указана необходимость случайного включения изображений в тензоры (shuffle=True). В случае указания shuffle=False изображения будут включаться в тензоры по порядку того, как они расположены в папках на рабочем компьютере. Также указывается число классов, на которое предполагается классифицировать изображения (class_mode=›binary›) и название набора тензоров (subset).

На этапе инициализации сети осуществляется ее конструирование. Как было указано выше, сверточная нейронная сеть – это комбинация сверточных и максимизирующих или усредняющих слоев с последующим добавлением классической многослойной нейронной сети. Таким образом и осуществляется конструирование нейронной сети:

```
model = tf.keras.models.Sequential([
tf.keras.layers.Conv2D(1000, (5, 5), activation='relu', input_shape=(224, 224, 3)),
tf.keras.layers.MaxPooling2D(3, 3),
tf.keras.layers.Conv2D(500, (5, 5), activation='relu'),
tf.keras.layers.MaxPooling2D(3, 3),
tf.keras.layers.Conv2D(250, (5, 5), activation='relu'),
tf.keras.layers.MaxPooling2D(3, 3),
tf.keras.layers.Flatten(),
tf.keras.layers.Dense(512, activation='tanh'),
tf.keras.layers.Dense(64, activation='tanh'),
tf.keras.layers.Dense(2, activation='softmax')
])
```

Приведенный программный код сформирует следующую структуру сверточной нейронной сети. Первым будет сверточный слой с ядром 5×5, функцией активации ReLU. Данный слой получает на входе тензор (пакет) изображений размеров 224×224 с тремя цветовыми каналами (R, G и B), а на выходе дает 1 000 масок размером 220×220. Уменьшение размера с 224×224 до 220×220 связано с тем, что ядро 5×5 может учитывать пиксели изображений, которые находятся от края больше чем на 2 пикселя. В связи с этим с каждого края изображения удаляются по 2 пикселя. Следующий максимизирующий слой на входе получает 1 000 масок размером 220×220 и уменьшает их в 3 раза за счет прохода ядром 3×3. Так, на выходе из второго слоя формируется 1 000 масок размером 73×73. Следующий сверточный слой из данных масок формирует 500 масок размером 69×69. Следующий максимизирующий слой уменьшает 500 масок до размера 23×23. Следующая комбинация сверточного и максимизирующего слоев сначала уменьшает число масок до 250 с размером 19×19, а затем уменьшает размер данных масок до 6×6. После трех комбинаций сверточных и максимизирующих слоев из трех каналов изображений 224×224 получаются 250 масок размером 6×6 пикселей. Общее число пикселей такого результата составляет 9 000 (6×6×250 = 9 000 пикселей).

Следующий слой (Flatten) переводит набор двумерных масок в один вектор значений (векторизация). То есть 250 масок размером 6x6 преобразуются в последовательный набор из 9 000 значений. Полносвязные слои (Dense) являются классическими слоями многослойной нейронной сети. Первый полносвязный слой получает 9 000 значений и передает их на 512 нейронов следующего полносвязного слоя с функцией активации гиперболический тангенс (tanh). После этого 512 значений полносвязного слоя передаются на слой, состоящий из 64 нейронов с функцией активации гиперболический тангенс. И на заключительном этапе 64 значения передаются на 2 выходных нейрона с функцией активации Softmax. Таким образом, из трех каналов изображений 224×224 на выходе получаются

всего 2 значения, каждое из которых отвечает за определенный класс изображения. Если на первом выходном нейроне значение больше, чем на втором, то изображению присваивается первый класс, если наоборот – то второй.

После конструирования нейронной сети выполнение следующей строки программного кода позволит получить упрощенную иллюстрацию процесса прохождения изображений по полученной сверточной нейронной сети (рис. 8):

model.summary()

На рис. 8 также можно увидеть число весов нейронов полносвязных слоев и коэффициентов ядер сверточных слоев, которое содержит полученная сверточная нейронная сеть, и сколько из них будут корректироваться в процессе обучения. В приведенном примере сверточная нейронная сеть содержит 20 343 224 веса нейронов и коэффициента ядер сверточных слоев, все из которых будут корректироваться при обучении сети.

Еще одним шагом этапа инициализации является настройка обучения сверточной нейронной сети. Данная настройка включает в себя указание оптимизатора (как правило, используется оптимизатор Adam), функцию ошибки (для классических классификационных задач используется sparse_categorical_crossentropy) и метрика, по которой будет оцениваться качество обучения (как правило, accuracy):

model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

После проведения этапа инициализации осуществляется переход к непосредственному этапу обучения сверточной нейронной сети с помощью следующего программного кода:

history = model.fit_generator(train_data_gen, steps_per_epoch=int(np.ceil(total_train / float(BATCH_SIZE_TRAIN))), epochs=10, validation_data=val_data_gen, class_weight={0 : 1, 1 : 2.2143})

Данная конструкция программного кода позволяет сохранять процесс обучения в переменную history, которая понадобится на заключительном этапе. Для обучения указывается обучающий набор (train_data_gen) тензоров (пакетов изображений), число эпох обучения (epochs), валидационный набор (validation_data_gen) тензоров, а также число шагов на одну эпоху. Число шагов на эпоху может задаваться непосредственно числом или конструкцией, как в приведенном примере. Данная конструкция позволяет получить отношение общего числа изображений в обучающем наборе к размеру обучающего тензора, что и является числом шагов на одну эпоху.

Помимо всего вышеперечисленного при обучении сверточной нейронной сети могут указываться веса классов изображений (class_weight). Необходимость указания весов классов может появиться, когда число изображений различных классов существенно отличается. Данная необходимость связана с тем, что при существенном преобладании числа изображений одного класса в обучающем наборе нейронная сеть будет хорошо обучаться распознавать именно этот класс, а распознавать

```
Model: "sequential_45"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_114 (Conv2D)          (None, 220, 220, 1000)    76000
_________________________________________________________________
max_pooling2d_108 (MaxPoolin (None, 73, 73, 1000)      0
_________________________________________________________________
conv2d_115 (Conv2D)          (None, 69, 69, 500)       12500500
_________________________________________________________________
max_pooling2d_109 (MaxPoolin (None, 23, 23, 500)       0
_________________________________________________________________
conv2d_116 (Conv2D)          (None, 19, 19, 250)       3125250
_________________________________________________________________
max_pooling2d_110 (MaxPoolin (None, 6, 6, 250)         0
_________________________________________________________________
flatten_30 (Flatten)         (None, 9000)              0
_________________________________________________________________
dense_96 (Dense)             (None, 512)               4608512
_________________________________________________________________
dense_97 (Dense)             (None, 64)                32832
_________________________________________________________________
dense_98 (Dense)             (None, 2)                 130
=================================================================
Total params: 20,343,224
Trainable params: 20,343,224
Non-trainable params: 0
_________________________________________________________________
```

Рис. 8. Упрощенная иллюстрация процесса прохождения изображения по сконструированной сверточной нейронной сети

меньший по численности изображений класс будет обучаться распознавать по остаточному принципу. Для того чтобы этого избежать, обычно используют установку весов классов. Среди используемых изображений в обучающем наборе данных содержится 3 906 изображений доброкачественных образований кожи и 1 764 изображений злокачественных новообразований кожи. То есть изображений злокачественных новообразований кожи в 2,2143 раза меньше. В связи с этим нулевому классу (классу изображений с доброкачественными образованиями кожи) присваивается вес, равный единице, а первому классу (классу изображений со злокачественными новообразованиями) – вес, равный 2,2143.

После обучения и сохранения результатов в переменную history на заключительном этапе может быть проведена оценка качества работы обученной сверточной нейронной сети:

accuracy = model.evaluate_generator(generator=test_data_gen, steps=np.ceil(total_test / float(BATCH_SIZE_TEST)))

print("Accuracy", accuracy)

Данный программный код позволяет протестировать полученную сверточную нейронную сеть на тестовом наборе изображений (test_data_gen) и вывести на экран показатели точности классификации (accuracy).

При необходимости сохранить полученную сверточную нейронную сеть можно воспользоваться следующим программным кодом:

model.save(‹C:/Модель CNN/Learning models/Model.h5›)

Необходимо отметить, что приведенные участки программного кода могут быть выполнены по частям поочередно, а могут быть выполнены единовременно. В случае запуска приведенного программного кода на экран будут выведены следующие результаты. После подготовительного этапа будут получены сводные данные по числу изображений в каждом наборе данных (рис. 9).

```
Доброкачественных в тренировочном наборе данных:  3906
Злокачественных в тренировочном наборе данных:  1764
Доброкачественных в тестовом наборе данных:  975
Злокачественных в тестовом наборе данных:  443
--
Всего изображений в тренировочном наборе данных:  5670
Всего изображений в тестовом наборе данных:  1418
```

Рис. 9. Сводные данные по числу изображений

При выполнении этапа предобработки изображений никакой информации на экран не выводится. На этапе инициализации сети будет выведена упрощенная иллюстрация процесса прохождения изображения по сконструированной сверточной нейронной сети (см. рис. 8).

На этапе обучения на экран будет выводиться последовательно процесс обучения (рис. 10). Этот этап самый длительный и в зависимости от числа изображений и числа весов и коэффициентов сверточной нейронной сети может занимать большое количество времени. В процессе обучения в каждую эпоху выводится время, затрачиваемое на эпоху обучения, время, затрачиваемое на один шаг обучения, значение функции ошибки, точность сети, значение функции ошибки и точность на валидационном наборе изображений. В приведенном примере на первую эпоху обучения было затрачено 4 898 секунд, на каждый шаг в среднем 28 секунд, значение функции ошибки (loss) составило 0,9461, точность (accuracy) – 0,6375, значение функции ошибки и точность на валидационном наборе изображений – 0,6232 и 0,6893 соответственно.

```
Epoch 1/10
178/178 [==============================] - 4898s 28s/step - loss: 0.9461 - accuracy: 0.6375 - val_loss: 0.6232 - val_accuracy:
0.6893
Epoch 2/10
178/178 [==============================] - 4849s 27s/step - loss: 0.7987 - accuracy: 0.7054 - val_loss: 0.8357 - val_accuracy:
0.6893
Epoch 3/10
178/178 [==============================] - 4818s 27s/step - loss: 0.7962 - accuracy: 0.7066 - val_loss: 0.8818 - val_accuracy:
0.6893
Epoch 4/10
178/178 [==============================] - 4811s 27s/step - loss: 0.7660 - accuracy: 0.7260 - val_loss: 0.8689 - val_accuracy:
0.6893
Epoch 5/10
178/178 [==============================] - 4820s 27s/step - loss: 0.7641 - accuracy: 0.7318 - val_loss: 0.8669 - val_accuracy:
0.6893
Epoch 6/10
178/178 [==============================] - 4988s 28s/step - loss: 0.7295 - accuracy: 0.7392 - val_loss: 0.8109 - val_accuracy:
0.6893
Epoch 7/10
178/178 [==============================] - 5027s 28s/step - loss: 0.7399 - accuracy: 0.7342 - val_loss: 0.8197 - val_accuracy:
0.6893
Epoch 8/10
178/178 [==============================] - 5055s 28s/step - loss: 0.7188 - accuracy: 0.7481 - val_loss: 0.7969 - val_accuracy:
0.6893
Epoch 9/10
178/178 [==============================] - 5106s 29s/step - loss: 0.6662 - accuracy: 0.7766 - val_loss: 0.6250 - val_accuracy:
0.6893
Epoch 10/10
178/178 [==============================] - 5023s 28s/step - loss: 0.6567 - accuracy: 0.7853 - val_loss: 0.6236 - val_accuracy:
0.6893
```

Рис. 10. Процесс обучения сверточной нейронной сети

Сравнивая результаты на первой и последней эпохах обучения, можно сказать, что значение функции ошибки уменьшилось на 30 % (до 0,6567), а точность увеличилась на 23 % (до 0,7853).

На заключительном этапе на экран выводится значение точности на изображениях, входящих в тестовый набор (рис. 11).

```
Accuracy 0.75241184
```

Рис. 11. Значение точности на изображениях, входящих в тестовый набор

Таким образом, в результате обучения и тестирования сверточной нейронной сети была получена сеть, классифицирующая изображения с доброкачественными и злокачественными образованиями кожи, с точностью 75,2 %.

Представление результатов построения сверточных нейронных сетей

При представлении результатов построения сверточных нейронных сетей в научной статье или диссертации следует учитывать, что читателя будут интересовать, во-первых, структура сверточной нейронной сети, во-вторых, настройки обучения и, в-третьих, качество классификации.

Структура сверточной нейронной сети может быть представлена в текстовом виде путем описания слоев сверточной нейронной сети, в виде табличной структуры подобно той, которая приведена на рис. 8 или в графическом виде, иллюстрирующем структуру сети (рис. 12). В зарубежных статьях чаще всего используется именно третий вариант представления.

На рис. 12 представлены все слои сверточной нейронной сети, а под каждым слоем указана его характеристика: индекс «C» или «M» на данной схеме указывают на сверточный или максимизирущий слой, цифра после индекса слоя указывает на порядковый номер слоя, на второй строке указаны размер ядра и функции активации нейронов, значение, стоящее до знака «@», указывает на число масок, которое получается на выходе из данного слоя, а значения, стоящие после знака «@», указывают на размер масок, которые получаются на выходе из данного слоя. Представленной на рис. 12 информации достаточно для повторения структуры сверточной нейронной сети и понимания распространения по ней цифрового изображения.

При описании результатов построения сверточных нейронных сетей настройки обучения достаточно указать, какой использовался оптимизатор, функцию ошибки и метрику, по которой оценивалось качество обучения. Так как чаще всего в классификационных задачах в качестве метрики для оценки качества обучения используется точность (accuracy), то и для представления качества классификации, построенной нейронной сетью, приводятся значения данного показателя на последней эпохе обучения и на тестовом наборе изображений.

Несмотря на мощный математический аппарат сверточных нейронных сетей в отечественных медицинских исследованиях, их потенциал пока раскрыт недостаточно. Это связано с тем, что практическая реализация построенных сверточных нейронных сетей весьма затруднительна. Получив обученную сверточную нейронную сеть, воспользоваться ей для решения реальной практической задачи без привлечения технических специалистов невозможно, так как данная сеть должна быть встроена в существующее программное обеспечение либо должно быть написано новое, в которое будет включена обученная сверточная нейронная сеть.

Таким образом, в статье представлены возможности применения сверточных нейронных сетей в медицинских исследованиях, представлены методология их построения и пример медицинской задачи, в которой основу составляет анализ изображений, а также приведены пример построения сверточной нейронной сети и программный код, реализующий данную сеть.

Применение при распознавании медицинских изображений сверточных нейронных сетей позволяет получить инструмент для классификации результатов медицинских диагностических методов, которые дают визуальный результат в виде цифрового изображения, на различные классы. Однако практическое использование построенных сверточных нейронных сетей без привлечения дополнительных специалистов достаточно затруднительно.

Авторство

Наркевич А. Н. внес существенный вклад в концепцию и дизайн исследования, получение, анализ и интерпретацию данных, подготовил первый вариант статьи, окончательно утвердил присланную в редакцию рукопись; Виноградов К. А. внес существенный вклад в концепцию и дизайн

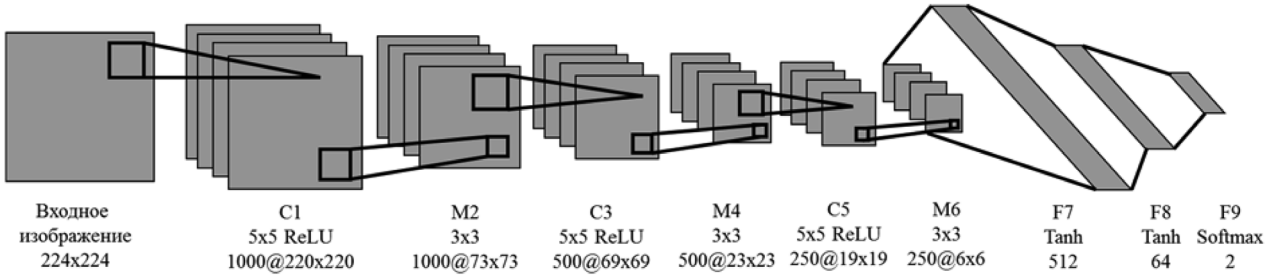


Рис. 12. Графическое представление структуры сверточной нейронной сети

исследования, получение, анализ и интерпретацию данных, существенно переработал первый вариант статьи на предмет важного интеллектуального содержания, окончательно утвердил присланную в редакцию рукопись; Параскевопуло К. М. внес существенный вклад в концепцию и дизайн исследования, получение, анализ и интерпретацию данных, существенно переработал первый вариант статьи на предмет важного интеллектуального содержания, окончательно утвердил присланную в редакцию рукопись; Мамедов Т. Х. внес существенный вклад в концепцию и дизайн исследования, получение, анализ и интерпретацию данных, существенно переработал первый вариант статьи на предмет важного интеллектуального содержания, окончательно утвердил присланную в редакцию рукопись.

Наркевич Артем Николаевич — ORCID 0000-0002-1489-5058; SPIN 9030-1493
Виноградов Константин Анатольевич — ORCID 0000-0001-6224-5618; SPIN 6924-0110
Параскевопуло Константин Михайлович — ORCID 0000-0002-4592-1386; SPIN 2320-3150
Мамедов Тимур Халигович — ORCID 0000-0003-2290-6439; SPIN 3393-0178

Список литературы / References

1. *Алексеева О. В., Россиев Д. А., Ильенкова Н. А.* Применение искусственных нейронных сетей в дифференциальной диагностике рецидивирующего бронхита у детей // Сибирское медицинское обозрение. 2010. Т. 66, № 6. С. 75–79.

Alekseeva O. V., Rossiev D. A., Il'enkova N. A. Application of artificial neural networks in the differential diagnosis of recurrent bronchitis in children. *Sibirskoe meditsinskoe obozrenie* [Siberian medical review]. 2010, 66 (6), pp. 75-79. [In Russian]

2. *Ахмед С. Х., Скородумов С. В.* Использование нейросетевых подходов в диагностировании заболеваний // Моделирование и анализ данных. 2020. Т. 10, № 2. С. 49–61.

Ahmed S. H., Skorodumov S. V. The use of neural network approaches in the diagnosis of diseases. *Modelirovanie i analiz dannyh* [Modeling and data analysis]. 2020, 10 (2), pp. 49-61. [In Russian]

3. *Волосова А. В.* Применение методов искусственного интеллекта для визуализации МРТ-изображений и ранней диагностики болезни Альцгеймера // Аспирант и соискатель. 2019. № 2. С. 124–126.

Volosova A. V. Application of artificial intelligence methods for visualization of MRI images and early diagnosis of Alzheimer›s. *Aspirant i soiskatel'* [The graduate student and the applicant]. 2019, 2, pp. 124-126. [In Russian]

4. *Гржибовский А. М., Иванов С. В., Горбатова М. А.* Однофакторный линейный регрессионный анализ с использованием программного обеспечения Statistica и SPSS // Наука и здравоохранение. 2017. № 2. С. 5–33.

Grjibovski A. M., Ivanov S. V., Gorbatova M. A. One-factor linear regression analysis using software Statistica and SPSS. *Nauka i zdravookhranenie* [Science and health]. 2017, 2, pp. 5-33. [In Russian]

5. *Гуляев Ю. В., Корженевский А. В., Черепенин В. А.* О возможности использования электроимпедансной компьютерной томографии для диагностики поражения легких вирусом COVID-19 // Журнал радиоэлектроники. 2020. № 5. С. 12.

Guljaev Ju. V., Korzhenevskij A. V., Cherepenin V. A. About the possibility of using electroimpedance computed tomography for the diagnosis of lung damage by the COVID-19 virus. *Zhurnal radioelektroniki* [Journal of Radioelectronics]. 2020, 5, p. 12. [In Russian]

6. *Дабагов А. Р., Томакова Р. А., Алексеев В. А., Кондрашов Д. С.* Метод автоматической классификации рентгеновских изображений на основе масок прозрачности // Высокопроизводительные вычислительные системы и технологии. 2020. Т. 4, № 1. С. 110–116.

Dabagov A. R., Tomakova R. A., Alekseev V. A., Kondrashov D. S. Method of automatic classification of X-ray images based on transparency masks. *Vysokoproizvoditel'nye vychislitel'nye sistemy i tekhnologii* [High-performance computing systems and technologies]. 2020, 4 (1), pp. 110-116. [In Russian]

7. *Далечина А. В., Беляев М. Г., Тюрина А. Н., Золотова С. В., Пронин И. Н., Голанов А. В.* Методы машинного обучения в сегментации глиом для планирования стереотоксической лучевой терапии // Лучевая диагностика и терапия. 2019. № 2. С. 24–31.

Dalechina A. V., Beljaev M. G., Tjurina A. N., Zolotova S. V., Pronin I. N., Golanov A. V. Methods of machine learning in glioma segmentation for planning stereotactic radiation therapy. *Luchevaya diagnostika i terapiya* [Radiation diagnostics and therapy]. 2019, 2, pp. 24-31. [In Russian]

8. *Ковалев В. А., Войнов Д. М., Малышев В. Д., Лапо Е. Д.* Компьютеризированная диагностика рака простаты на основе полнослайдовых гистологических изображений и методов глубокого обучения // Информатика. 2020. Т. 17, № 4. С. 48–60.

Kovalev V. A., Voinov D. M., Malyshev V. D., Lapo E. D. Computerized diagnosis of prostate cancer based on full-slide histological images and methods of deep learning. *Informatika* [Computer science]. 2020, 17 (4), pp. 48-60. [In Russian]

9. *Константинова Е. Д., Вараксин А. Н., Жовнер И. В.* Определение основных факторов риска развития неинфекционных заболеваний: метод деревьев классификации // Гигиена и санитария. 2013. Т. 92, № 5. С. 69–72.

Konstantinova E. D., Varaksin A. N., Zhovner I. V. Determination of the main risk factors for the development of non-communicable diseases: method of classification trees. *Gigiena i sanitariya*. 2013, 92 (5), pp. 69-72. [In Russian]

10. *Нагорнов Н. Н.* Моделирование вейвлет-обработки трехмерных изображений в медицине // Современная наука и инновации. 2019. № 3. С. 22–31.

Nagornov N. N. Modeling of three-dimensional image wavelet processing in medicine. *Sovremennaya nauka i innovatsii* [Modern science and innovation]. 2019, 3, pp. 22-31. [In Russian]

11. *Никитина М. А., Пчелкина В. А., Чернуха И. М.* Нейросетевые технологии в анализе гистологических препаратов // Контроль качества продукции. 2019. № 3. С. 17–24.

Nikitina M. A., Pchelkina V. A., Chernukha I. M. Neural network technologies in the analysis of histological preparations. *Kontrol' kachestva produktsii* [Product quality control]. 2019, 3, pp. 17-24. [In Russian]

12. *Харевич О. Н., Лаптева И. М., Лаптева Е. А., Королева Е. Г.* Клинические фенотипы тяжелой астмы (по результатам кластерного анализа) // Вестник Санкт-Петербургского университета. Медицина. 2015. № 2. С. 28–39.

Kharevich O. N., Lapteva I. M., Lapteva E. A., Koroleva E. G. Clinical phenotypes of severe asthma (based on the results of cluster analysis). *Vestnik Sankt-Peterburgskogo*

universiteta. Meditsina [Bulletin of the Saint Petersburg University. Medicine]. 2015, 2, pp. 28-39. [In Russian]

13. *Шилов О. А.* Сегментация изображений легких человека с использованием свёрточных нейронных сетей // Процессы управления и устойчивость. 2020. Т. 7, № 1. С. 178–182.

Shilov O. A. Segmentation of images of human lungs using convolutional neural networks. *Processy upravleniya i ustoichivost'* [Control processes and stability]. 2020, 7 (1), pp. 178-182. [In Russian]

14. Anwar S. M., Majid M., Qayyum A., Awais M., Alnowami M., Khan M. K. Medical image analysis using convolutional neural networks: A review. *Journal of Medical Systems*. 2018, 42 (11), p. 226. DOI: 10.1007/s10916-018-1088-1.

15. Bamber J. H., Evans S. A. The value of decision tree analysis in planning anaesthetic care in obstetrics. *International Journal of Obstetric Anesthesia*. 2016, 27, pp. 55-61. DOI: 10.1016/j.ijoa.2016.02.007.

16. Ben-Gal I., Dana A., Shkolnik N., Singer G. Efficient Construction of Decision Trees by the Dual Information Distance Method. *Quality Technology & Quantitative Management*. 2014, 11 (1), pp. 133-147.

17. Chereda H., Bleckmann A., Kramer F., Leha A., Beissbarth T. Utilizing molecular network information via graph convolutional neural networks to predict metastatic event in breast cancer. *Studies in Health Technology and Informatics*. 2019, 267, pp. 181-186. DOI: 10.3233/SHTI190824.

18. Cybenko G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*. 1989, 2 (4), pp. 303-314. DOI: 10.1007/BF02551274.

19. 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. 1 (4). pp. 541-551.

20. Nair V., Hinton G. E. Rectified Linear Units Improve Restricted Boltzmann Machines. *27th International Conference on International Conference on Machine Learning*. 2010. USA, Omnipress, pp. 807-814.

21. Sumida I., Magome T., Kitamori H., Das I. J., Yamaguchi H., Kizaki H., Aboshi K., Yamashita K., Yamada Y., Seo Y., Isohashi F., Ogawa K. Deep convolutional neural network for reduction of contrast-enhanced region on CT images. *Journal of Radiation Research*. 2019, 60 (5), pp. 586-594. DOI: 10.1093/jrr/rrz030.

22. Viebke A., Memeti S., Pllana S., Abraham A. CHAOS: a parallelization scheme for training convolutional neural networks on Intel Xeon Phi. *The Journal of Supercomputing*. 2019, 75 (1), pp. 197-227. DOI:10.1007/s11227-017-1994-x.

23. Wollmann T., Gunkel M., Chung I., Erfle H., Rippe K., Rohr K. GRUU-Net: Integrated convolutional and gated recurrent neural network for cell segmentation. *Medical Image Analysis*. 2019, 56, pp. 68-79. DOI: 10.1016/j.media.2019.04.011.

24. Wu Y., Lin L., Wang J., Wu S. Application of semantic segmentation based on convolutional neural network in medical images. *Sheng Wu Yi Xue Gong Cheng Xue Za Zhi*. 2020, 37 (3), pp. 533-540. DOI: 10.7507/1001- 5515.201906067.

25. Yasaka K., Akai H., Kunimatsu A., Kiryu S., Abe O. Deep learning with convolutional neural network in radiology. *Japanese Journal of Radiology*. 2018, 36 (4), pp. 257-272. DOI: 10.1007/s11604-018-0726-3.

26. Zhang W. Image processing of human corneal endothelium based on a learning network. *Applied Optics*. 1991, 30 (29), pp. 4211-4217. DOI: 10.1364/AO.30.004211.

27. Zhang Y., Lin H., Yang Z., Wang J., Sun Y., Xu B., Zhao Z. Neural network-based approaches for biomedical relation classification: A review. *Journal of Biomedical Informatics*. 2019, 99, e103294. DOI: 10.1016/j.jbi.2019.103294.

Контактная информация:

Наркевич Артем Николаевич – доктор медицинских наук, доцент, заведующий лабораторией медицинской кибернетики и управления в здравоохранении, заведующий кафедрой медицинской кибернетики и информатики ФГБОУ ВО «Красноярский государственный медицинский университет им. проф. В. Ф. Войно-Ясенецкого» Министерства здравоохранения Российской Федерации

Адрес: 660022, Красноярский край, г. Красноярск, ул. Партизана Железняка, д. 1 E-mail: narkevichart@gmail.com