Экономическая оценка внедрения систем искусственного интеллекта в сфере медицины

М.И. Русаков,
студент ИКБ, кафедра КБ-9 «Предметно-ориентированные информационные системы», РТУ МИРЭА (e-mail: kronologa@yahoo.com)
С.О. Потапов,
магистр, преподаватель ИКБ, кафедра КБ-9 «Предметно-ориентированные информационные системы», РТУ МИРЭА (e-mail: serpotapov2000@mail.ru)

Аннотация. *В современном мире растет интерес к исследованию искусственных нейронных сетей (ИНС). Одна из задач, решаемых нейросетями – выявление патологий на МРТ-снимках. В Европе и США нейронные сети уже используется, но в России такая практика только набирает обороты. Предмет. Способность системы обнаруживать и выделять искомые элементы на изображениях в экономических рамках повышения эффективности диагностики и лечения. Цели. Оценить экономический потенциал разработанной системы искусственного интеллекта выявления патологий на МРТ-снимках. Методология. Метод базируется на свёрточной нейронной сети сегментации изображений "U-Net", широко используемый в биомедицинской визуализации. Результаты. Настоящее исследование позволило оценить экономический и медицинский потенциал внедрения систем искусственного интеллекта на примере разработанной системы выявления патологий на МРТ-снимках, что является важной информацией для экономики и системы здравоохранения России. Область применения. Для принятия управленческих решений в здравоохранении.*

Abstract. *There is a growing interest in the study of artificial neural networks (ANN) now. The detection of pathologies on MRI images is one of the tasks that solved by neural networks. Neural networks are already used in Europe and the USA, but this practice is only gaining momentum in Russia. Subject. The ability of the system to detect and allocate the desired elements in images within the economic framework of improving the effectiveness of diagnosis and treatment. Goals. The main goal of research is evaluating the economic potential of the developed artificial intelligence system for detecting pathologies on MRI images. Methodology. The used method is based on the convolutional neural network of image segmentation "U-Net" which widely used in biomedical visualization. Results. This research allowed to assess the economic and medical potential of introducing artificial intelligence systems on the basis of the developed system for detecting pathologies on MRI images, which is an important information for the economy and the health system of Russia. Scope of application. The main guideline to making management decisions in healthcare.*

Ключевые слова: *распознавание элементов, U-Net, нейросети, искусственный интеллект в медицине, искусственный интеллект в здравоохранении.*

Keywords: *element recognition, U-Net, neural networks, artificial intelligence in medicine, artificial intelligence in healthcare.*

В современном мире растет интерес к исследованию искусственных нейронных сетей (ИНС). Одна из задач, решаемых нейросетями – выявление патологий на МРТ-снимках. В Европе и США нейронные сети уже используется, но в России такая практика только набирает обороты [1].

Основное преимущество автоматического распознавания аномалий позвоночника на МРТ-изображениях – возможность получения быстрых и точных результатов. Это позволит врачам быстрее и точнее ставить диагноз, вследствие чего увеличится качество медицинских услуг [1, 2].

ИНС могут диагностировать позвонки с высокой степенью точности и, поэтому такие методы могут быть использованы для постановки первичных диагнозов и наблюдений [3, 4].

Термин "нейронная сеть" происходит из исследований мозга и применяется к моделям с большим количеством параметров. Многие такие модели сохраняют родственные термины, однако были разработаны для небиологических задач. Нейронные сети распределены и способны хранить и использовать экспериментальные данные. Они напоминают мозг в том, что информация получается в процессе обучения и хранится через синаптические веса или нейронные связи разной силы. При этом нейронные сети не имеют строгих ограничений на структуру модели

U-Net – метод сегментации изображений, широко используемый в биомедицинской визуализации. Сеть U-Net состоит из двух частей: пути сокращения и экспансивного пути. В пути сокращения используются свертки 3x3 и слои максимального пула для сокращения размерности карт признаков. В экспансивном пути используются свертки и повышение частоты 2x2 для повышения дискретизации карт признаков. После объединения объектов с разных слоев применяются последовательные свертки 3x3. На выходе сети

применяется свертка 1x1 для получения сегментированного изображения. Края изображения обрезаются, чтобы удалить пиксели с малым количеством контекстной информации. Такая архитектура позволяет U-Net эффективно сегментировать объекты с использованием контекста из большей области [5].

В ходе этой части была создана разметка на МРТ-снимках, после чего был загружен архив таких снимков. Сегментация каждого участка на отдельном снимке проводилась вручную (Рисунок 1). После завершения сегментации был скачан JSON-файл с разметками.

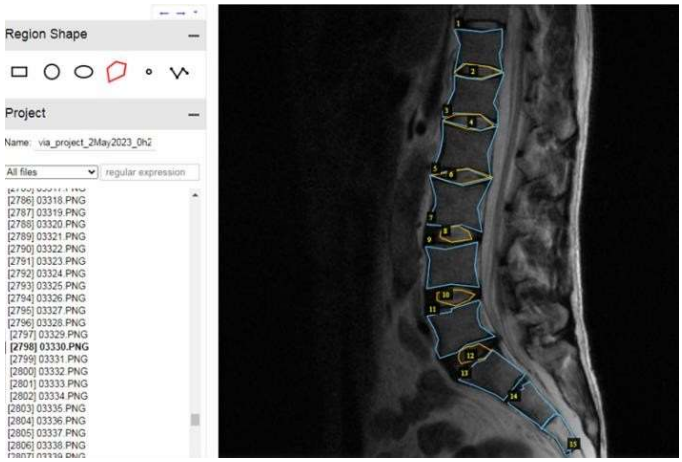

Рис. 1. **Сегментация участков**

Следующим шагом было рисование масок. Маски — это представление в форме изображения, которое показывает, к какому классу относятся пиксели. В данной работе такие маски будут бинарными для определения принадлежности пикселя к классу («0» или «1»). Для каждого из загруженных изображений подбирались необходимые данные из JSON-файла, подбор производился по имени файла, который соответствовал началу новой строки в файле с разметкой. Для этого создавались два массива, которые хранили изображения и регионы. Далее в эти массивы записывались адреса изображений, после чего с помощью записывались высота и ширина изображений. После этого на заранее созданную пустую (черную) маску рисовались контуры.

После блока рисования в системе следует блок демонстрации масок. С помощью функции были выгружены оригинальные изображения (МРТ-снимки) и их маски. Результаты представлены на Рисунке 2.

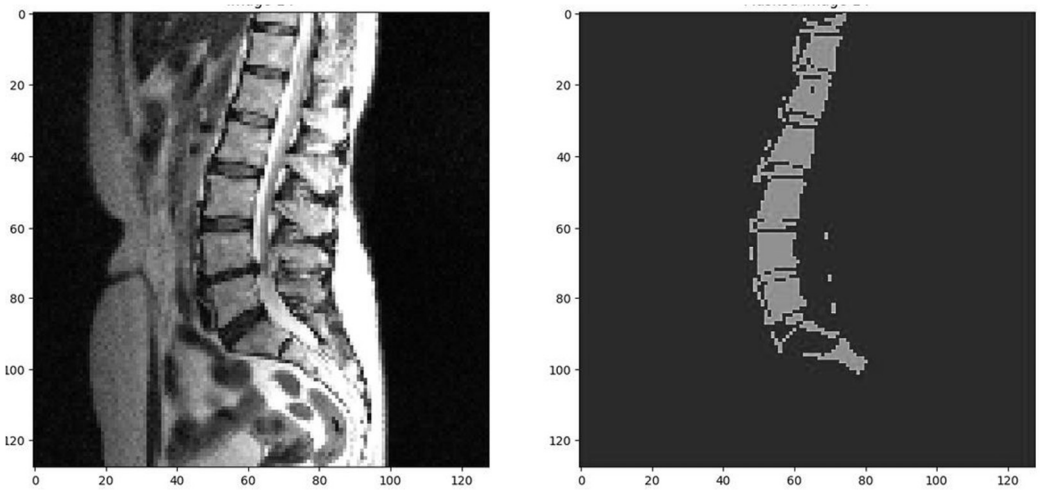

Рис. 2. **Маска участков**

После этого началась конструкция сети U-Net [5, 6]. В разделе *«Блок Декодера»* (см. Приложение А) понижаются пространственные размеры в два раза. Сам блок состоит из слоев сверток и субдискретизации. Этот блок принимает выход предыдущего слоя и выход заданного блока кодера. Блок реализует:

— увеличение в два раза пространственные размеры выхода предыдущего слоя за счет транспонированной свертки;

— сравнение результата с полученным выходом блока кодера;

— применение двух свёрточных слоёв, не изменяющих пространственные размеры.

Завершающей частью является сборка сети (блок «Собираем U-Net»). Сеть U-Net состоит из:

— 4 блоков кодера с понижением размеров карт признаков, но увеличением числа каналов;

— одного блока кодера без понижения размера;

— 4 блоков декодера с повышением размера карт признаков и понижением числа каналов;

— свёрточного слоя не изменяющего размеры;

— выходного свёрточного слоя с числом фильтров по количеству классов.

Более подробно создание сети U-Net, а также комментарии к частям кода показаны в Приложении А.

В результате получилась сеть, которая принимает на вход изображение, а в результате выдаёт нарисованную маску, пиксели на которой рассчитаны путём автоматического дешифрирования снимка [5, 6].

Перед обучением сети данные были разделены на обучающие и проверочные части (80% к 20%). Также добавлена метрика – “accurancy”, показывающая аккуратность (точность) обучения.

После этого началось обучение сети (блок «Обучение сети» в Приложении А). Для качественного обучения нужно было подобрать параметры “batch_size” (количество входных данных). и “epochs” (количество циклов обучения, эпох).

При малом количестве эпох (20) сеть не могла достаточно обучиться и выдавала пустой результат (Рисунок 3).

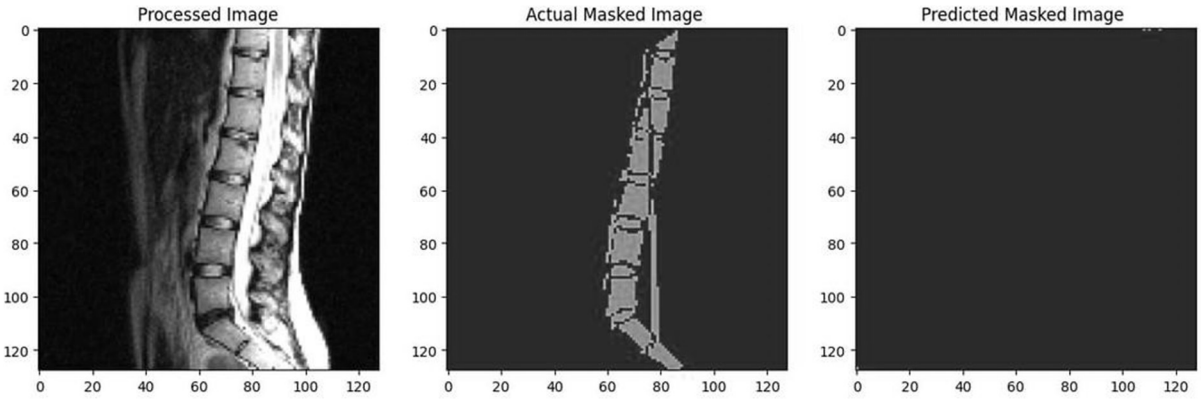

Рис. 3. **Недостаточное обучение сети**

При чрезмерном количестве циклов обучения (100) система выдавала чрезмерное количество распознанных пикселей (Рисунок 4), то есть происходило переобучение сети.

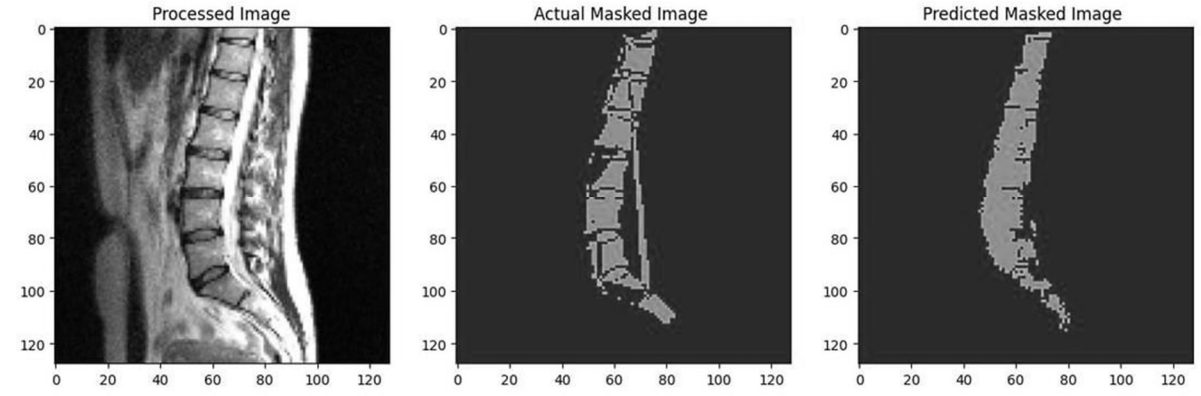

Рис. 4. **Переобучение сети**

Экспериментальным путём были подобраны оптимальные значения параметров обучения (40 эпох и размер данных «2»), Скриншот результатов при таких параметрах показан на Рисунке 5.

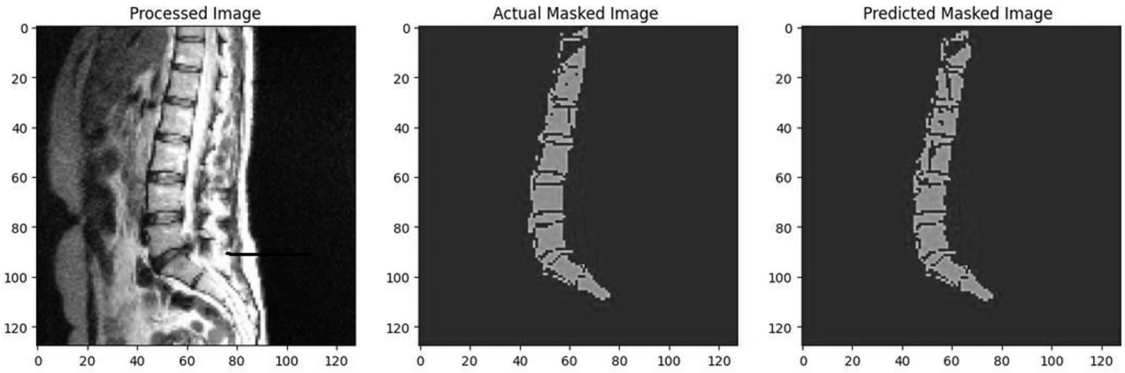

Рис. 5. **Результаты при 40 эпохах и размере «2».**

При таких условиях достигается наилучший вариант сегментации частей позвоночника на загружаемых снимках. На Рисунке 6 Представлены графики обучения в зависимости от эпох.

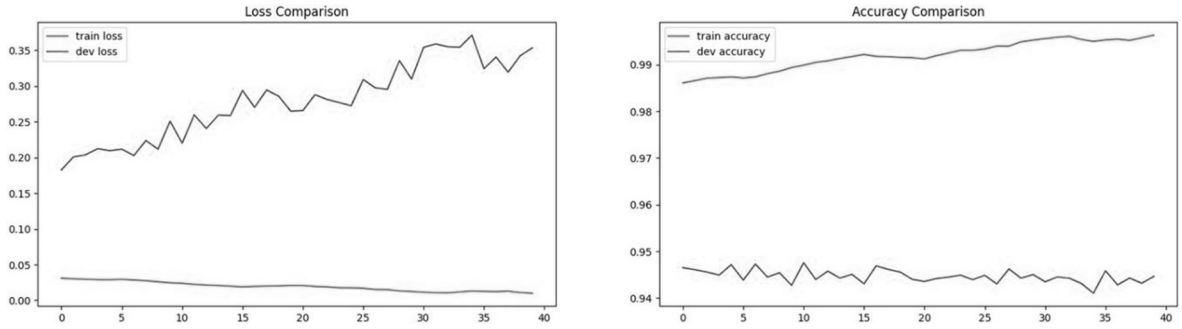

Рис. 6. **Графики обучения сети**

Как видно из графиков, с количеством эпох возрастает точность изучения (слева) и падает оценка ошибок (справа). Это говорит о том, что созданная сеть способна самообучаться на основе загружаемых данных.

Использование искусственного интеллекта (ИИ) в медицине имеет огромный потенциал для экономии затрат на здравоохранение и улучшения результатов лечения пациентов.

Одним из основных способов оптимизации медицинских услуг с использованием ИИ в медицине является повышение эффективности диагностики и лечения. Благодаря анализу больших объемов данных о симптоматике, ИИ позволяют выявлять заболевания на более ранних стадиях, что позволит предотвратить развитие осложнений. Это существенно снизит государственные или частные расходы на лечение пациентов за счет уменьшения продолжительности госпитализации и использования дорогостоящих операций.

Персонализированный подход к лечению на основе генетического профиля пациента, сформированный с помощью ИИ, сократит расходы на назначение неэффективных или неподходящих лекарственных препаратов. Это также поможет снизить шанс появления побочных эффектов, что предотвратит расходы на лечение таких. Помимо этого, такой генетический профиль может использоваться как инструмент предсказания заболеваний у пациентов на основе истории болезни его родителей и семьи [7].

Примером успешного применения таких сетей может служить использование их в целях ускорения определения симптомов, постановки диагноза, а также назначения лечения в условиях пандемии COVID-19 [8]. Мониторинг лечения ИИ помог создать интеллектуальную платформу для автоматического мониторинга и прогнозирование распространения этого вируса. Кроме того, он смог предсказать количество положительных случаев заболевания и смертей в разных регионах. Это помогло определить наиболее уязвимые для распространения регионы и предотвратить очаговые вспышки. Вдобавок искусственный интеллект использовался как инструмент для разработки вакцин, а также помогал снизить нагрузку на медицинский персонал за счёт первичной диагностики заболеваний и постановки диагноза. Благодаря использованию искусственного интеллекта для решения проблем во время эпидемии было накоплено много информации, которая может быть использована как для решения последующих задач такого плана, так и для предотвращения их появления.

Вопрос уменьшения расходов на медицинские услуги становится всё острее и использование ИИ имеет потенциал уменьшить такие траты.

Американская компания Accenture оценила на основе существующих ИИ-решений, что к 2026 году они могут сохранить до 150 миллиардов долларов ежегодной экономии для системы здравоохранения США. Для человечества станет большим достижением стабилизация расходов на медицину за счёт искусственного интеллекта.

В России перспективность использования искусственного интеллекта вопрос времени, но вместе с этим возникает вопрос выгоды такого внедрения, в том числе и в медицинскую отрасль. В соответствии с данными из базы данных расходов на систему здравоохранения ВОЗ (Всемирной организации здравоохранения). Расходы росли с 7,6 трлн долларов в 2016 году до 7,8 трлн долларов в 2017 и до 8,3 трлн долларов в 2018 году, что в конечном итоге определило расходы в 1110 долларов США в 2018 году на 1 жителя Земли. Такие темпы расходов опередили темпы роста экономики [9].

Американский поставщик медицинских услуг OptumIQ на основе опроса более 500 руководителей в области здравоохранения опубликовал отчет «Annual Survey on AI in Health Care» [10], в котором определил увеличение числа удачных внедрений ИИ в медицине почти на 88% по сравнению с предыдущим годом. В отчёте также говорится о повышении уверенности лидеров здравоохранения в ИИ и рост инвестиций в это направление. Более того, 9 из 10 лидеров уверены, что затраты таких внедрений окупятся, причем около половины считает, что менее чем за 3 года.

Российская ассоциация электронных коммуникаций (РАЭК) совместно с Высшей Школой Экономики при поддержке Microsoft провели исследование использования технологии ИИ российским бизнесом [11], в отчёт о котором также вошёл отдельный раздел по перспективам ИИ в сфере здравоохранения. В отчёте прогнозируется повышение прибыли с 2,2 % до 3,4 % к 2035 году, а также рост доходов до 60% от использования таких систем. Рынок ПО, оборудования и обслуживания ИИ для отрасли здравоохранения к 2025 году превысит $ 34 млрд. Среди перспективных технологий — анализ медицинских изображений (теме которых и посвящена данная работа), разработка виртуальных помощников, создание новых лекарств, предложение рекомендаций по медицинскому лечению и обработка персональных данных пациентов. Эксперты исследования называют сферу здравоохранения в России одной из наиболее перспективных для использования искусственного интеллекта. Уже сейчас здравоохранение активно внедряет инструменты оптимизации принятия решений. В области искусственного интеллекта здравоохранение обладает большим потенциалом, особенно учитывая бюджеты на разработку и ожидаемый экономический эффект от использования оптимизированных алгоритмов.

Приложение А. Листинг программы.

```
#@title Загрузка библиотек import os
import json
import numpy as np import skimage.draw
import cv2#работа с масками import gdown #скачивание архива import shutil #распаковка архива

# обработка и рисование изображений import imageio
from PIL import Image

# рисование графиков
import matplotlib.pyplot as plt

# для создание модели import tensorflow as tf
# Слои, которые потребуются
from tensorflow.keras.layers import Input from tensorflow.keras.layers import Conv2D
from tensorflow.keras.layers import MaxPooling2D from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import BatchNormalization
from tensorflow.keras.layers import Conv2DTranspose # транспонированная свертка
from tensorflow.keras.layers import concatenate # объединение тензоров

from tensorflow.keras.losses import binary_crossentropy # функция ошибки from sklearn.model_selection import train_test_split # разделение данных на обучающее и проверочное множества

#Загрузка датасета url =
'https://drive.google.com/uc?export=download&confirm=y&id=1jQ5IROvRCphNRJ_PJK UiOd-mEUGn66xOD'
output = 'data.zip'
gdown.download(url, output, quiet=False)

shutil.unpack_archive('data.zip', "/content/")

#Обозначение испольуемых данных (картинки + json) IMAGE_FOLDER = "/content/images" PATH_ANNOTATION_JSON = 'sagittals.json'
annotations = json.load(open(PATH_ANNOTATION_JSON, 'r')) imgs = annotations["_via_img_metadata"]
```

```
#Рисование масок mask_ = []
img_ = []
for imgId in imgs:
filename = imgs[imgId]['filename'] regions = imgs[imgId]['regions'] if len(regions) <= 0:
continue points= []
for ln in range(0,len(regions)):
polygons = regions[ln]['shape_attributes']

image_path = os.path.join(IMAGE_FOLDER, filename)

#чтение высоты+ширины картинки image = cv2.imread(image_path) height, width = image.shape[:2]

#создание пустой (чёрной) маски
maskImage = np.zeros((height,width), dtype=np.uint8) countOfPoints = len(polygons['all_points_x'])
for i in range(countOfPoints):
x = int(polygons['all_points_x'][i]) y = int(polygons['all_points_y'][i]) points.append((x, y))

contours = np.array(points)
contours = contours.reshape((-1, 1, 2))

#рисование
for i in range(width):
for j in range(height):
if cv2.pointPolygonTest(contours, (i, j), False) > 0: maskImage[j,i] = 1
image = cv2.resize(image, (128, 128)) img_.append(image/255.0)

#сохранение результата
maskImage = cv2.resize(maskImage, (128, 128)) mask_.append(np.reshape(maskImage,(128,128,1)))

# Создание сети Unet # Блок кодера
def EncoderMiniBlock(inputs, # вход
n_filters=32, # количество фильтров dropout_prob=0.3, # вероятность отброса max_pooling=True): #
использовать ли субдискретизацию

# Два сверточных слоя (выходы) с инициализацией.
# Набивка 'Same' не изменяет пространственные размеры. conv = Conv2D(n_filters,
3, # размер ядра activation='relu', padding='same',
kernel_initializer='HeNormal')(inputs) # инициализация (только при создании)
conv = Conv2D(n_filters,
3, # Kernel size activation='relu', padding='same',
kernel_initializer='HeNormal')(conv)

# Нормализация
conv = BatchNormalization()(conv, training=False)

# dropout, если задан if dropout_prob > 0:
conv = tf.keras.layers.Dropout(dropout_prob)(conv)
# Субдискретизация, если задано. Конкретно MaxPooling, уменьшает пространственные размеры в
два раза
if max_pooling:
next_layer = tf.keras.layers.MaxPooling2D(pool_size = (2,2))(conv)
else:
next_layer = conv

# выход слоев ДО субдискретизации, будет передаваться в другие блоки.
skip_connection = conv

return next_layer, skip_connection # возвращаем выход блока и выход слоев до субдискретизации


# Блок декодера
def DecoderMiniBlock(prev_layer_input, # выход предыдущего слоя (блока) skip_layer_input, # выход
соответствующего блока
кодера
n_filters=32): # число фильтров
```

```
# Транспонированная свертка увеличивает пространственный размер карты признаков в два раза
up = Conv2DTranspose(
n_filters,
(3,3),        # размер ядра strides=(2,2),
padding='same')(prev_layer_input) # набивка, чтобы не уменьшался размер при выполнении свертки

# Конкатенируем по каналам (измерение 3) выход транспонированной свертки и выход блока кодера
merge = concatenate([up, skip_layer_input], axis=3)

# Две свертки не изменяющие размеры, с инициализацией conv = Conv2D(n_filters,
3,             # Kernel size activation='relu', padding='same',
kernel_initializer='HeNormal')(merge) conv = Conv2D(n_filters,
3, # Kernel size activation='relu', padding='same',
kernel_initializer='HeNormal')(conv)
return conv

#Сборка Unet
def UNetCompiled(input_size=(128, 128, 3), # размер изображения-входа
n_filters=32, # базовое число фильтров n_classes=3): # число классов

# Вход в сеть заданного размера inputs = Input(input_size)

# Кодер
# блок 1 принимает вход в сеть, число фильтров базовое, dropout нет, понижает размеры карты
cblock1 = EncoderMiniBlock(inputs, n_filters,dropout_prob=0, max_pooling=True)
# блок 2 принимает выход блока 1 (обратите внимание что блоки кодера возвращают два выхода),
# число фильтров в два раза больше, dropout нет, понижает размеры карты cblock2 =
EncoderMiniBlock(cblock1[0],n_filters*2,dropout_prob=0,
max_pooling=True)
# блок 3 принимает выход блока 2, число фильтров еще в два раза больше, dropout нет, понижает
размеры карты
cblock3 = EncoderMiniBlock(cblock2[0], n_filters*4,dropout_prob=0, max_pooling=True)
# блок 4 принимает выход блока 3, число фильтров еще в два раза больше, dropout есть, понижает
размеры карты
cblock4 = EncoderMiniBlock(cblock3[0], n_filters*8,dropout_prob=0.3, max_pooling=True)
# блок 5 принимает выход блока 4, число фильтров еще в два раза больше, dropout есть, НЕ пони-
жает размеры карты
cblock5 = EncoderMiniBlock(cblock4[0], n_filters*16, dropout_prob=0.3, max_pooling=False)

# Декодер
# блок 6 принимает выход блока 5, и второй выход (т.е. до слоя субдискретизации) блока 4,
# число фильтров в два раза меньше чем у блока 5, повышает размеры карты ublock6 =
DecoderMiniBlock(cblock5[0], cblock4[1],  n_filters * 8)
# блок 7 принимает выход блока 6, и второй выход блока 3, число фильтров в два раза меньше,
повышает размеры карты
ublock7 = DecoderMiniBlock(ublock6, cblock3[1], n_filters * 4)
# блок 8 принимает выход блока 7, и второй выход блока 2, число фильтров в два раза меньше,
повышает размеры карты
ublock8 = DecoderMiniBlock(ublock7, cblock2[1], n_filters * 2)
# блок 9 принимает выход блока 8, и второй выход блока 1, число фильтров в два раза меньше,
повышает размеры карты
ublock9 = DecoderMiniBlock(ublock8, cblock1[1], n_filters)

# слой свертки без изменения размеров карты, число фильтров как у предыдущего блока
conv9 = Conv2D(n_filters,
3,
activation='relu', padding='same',
kernel_initializer='he_normal')(ublock9)
# слой свертки без изменения размеров карты, число фильтров по количеству классов
conv10 = Conv2D(n_classes, 1, padding='same')(conv9)

# Создаем модель из слоев
model = tf.keras.Model(inputs=inputs, outputs=conv10) return model
for i in range(len(mask_)):
img_view = img_[i] # путь к изображению mask_view = mask_[i] # путь к его маске
```

```
fig, arr = plt.subplots(1, 2, figsize=(15, 15)) arr[0].imshow(img_view) # рисуем изображение arr[0].set_title('Image '+ str(i)) arr[1].imshow(mask_view) # рисуем маску arr[1].set_title('Masked Image '+ str(i))

#Обработка
X = np.array(img_) y = np.array(mask_)

print("X Shape:", X.shape) print("Y shape:", y.shape) #Классы : фон, объект, контур print(np.unique(y)) #Демонстрация
image_index = 0
fig, arr = plt.subplots(1, 2, figsize=(15, 15)) arr[0].imshow(X[image_index]) arr[0].set_title('Processed Image') arr[1].imshow(y[image_index,:,:]) arr[1].set_title('Processed Masked Image ')

#Разделяем данные на обучающие проверочные (80/20%)
X_train, X_valid, y_train, y_valid = train_test_split(X, y, test_size=0.2, random_state=123)
#Создаем сеть, 3 класса, 32 базовых каналов
unet = UNetCompiled(input_size=(128,128,3), n_filters=32, n_classes=3) unet.summary()
# оптимизатор Adam, функция ошибки - кроссэнтропия которая применяет softmax к выходам сети, метрика - аккуратность unet.compile(optimizer=tf.keras.optimizers.Adam(),
loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=T
rue),
metrics=['accuracy'])

# Обучение сети (40 эпох, время ~ 12 минут)
results = unet.fit(X_train, y_train, batch_size=4, epochs=40, validation_data=(X_valid, y_valid))

fig, axis = plt.subplots(1, 2, figsize=(20, 5)) axis[0].plot(results.history["loss"], color='r', label = 'train loss') axis[0].plot(results.history["val_loss"], color='b', label = 'dev loss') axis[0].set_title('Loss Comparison')
axis[0].legend()
axis[1].plot(results.history["accuracy"], color='r', label = 'train accuracy')
axis[1].plot(results.history["val_accuracy"], color='b', label = 'dev accuracy')
axis[1].set_title('Accuracy Comparison') axis[1].legend()

#Качество обучения unet.evaluate(X_valid, y_valid)

#Демонстрация работы сети def VisualizeResults(index):
img = X_valid[index] # изображение
img = img[np.newaxis, ...] # добавляем измерение примеров pred_y = unet.predict(img) # расчитываем маску
pred_mask = tf.argmax(pred_y[0], axis=-1) # выбираем максимальный класс (канал - последнее измерение)
pred_mask = pred_mask[..., tf.newaxis] # канальное измерение добавляем
# рисуем
fig, arr = plt.subplots(1, 3, figsize=(15, 15)) arr[0].imshow(X_valid[index]) arr[0].set_title('Processed Image') arr[1].imshow(y_valid[index,:,:]) arr[1].set_title('Actual Masked Image ') arr[2].imshow(pred_mask[:,:,0]) arr[2].set_title('Predicted Masked Image ')
index = 1 VisualizeResults(index).
```

Библиографический список:

1.Щукина Н.А. Нейросетевые модели в задаче классификации медицинских изображений. Моделирование, оптимизация и информационные технологии – 2021 – № 9(4).

2.Кравченко В. О. Методы использования искусственных нейронных сетей в медицине //Устойчивое развитие науки и образования. – 2018. – №. 6. – С. 266-270.

3.Дороничева А. В., Савин С. З. Метод сегментации медицинских изображений //Фундаментальные исследования. – 2015. – №. 5-2. – С. 294- 298.

4.Интернет-портал «IBM Документация». Раздел «Что такое нейронная сеть?» [Электронный ресурс]. URL: www.ibm.com/docs/ru/spss- statistics/saas?topic=networks-what-is-neural-network

5.Siddique N. et al. U-net and its variants for medical image segmentation: A review of theory and applications //Ieee Access. – 2021. – Т. 9. – С. 82031-82057.

6.Бредихин - А. И. Алгоритмы обучения сверточных нейронных сетей //Вестник Югорского государственного университета. – 2019. – №. 1 (52). – С. 41-54.

7.Esteva A. et al. A guide to deep learning in healthcare //Nature medicine. – 2019. – Т. 25. – №. 1. – С. 24-29.

8.Vaishya R. et al. Artificial Intelligence (AI) applications for COVID-19 pandemic //Diabetes & Metabolic Syndrome: Clinical Research & Reviews. – 2020. – Т. 14. – №. 4. – С. 337-339

9. Лукичев, П. М. Экономика искусственного интеллекта: возможности и проблемы использования в здравоохранении / П. М. Лукичев, О. П. Чекмарев // Вопросы инновационной экономики. – 2022. – Т. 12, № 2. – С. 1111-1130.

10. Ежегодный отчёт «Наблюдение за искусственным интеллектом в здравоохранении» американского поставщика медицинских услуг OPTUM. [Электронный ресурс]. URL: cdn-aem.optum.com/content/dam/optum3/optum/en/resources/ebooks/wf1628666-iq-ai-special-report-2.pdf

11. Сайт РАЭК. Раздел «Цифровая экономика от теории к практике: как российский бизнес использует искусственный интеллект». [Электронный ресурс]. URL: https://raec.ru/activity/analytics/11002/.