ИНФОРМАТИКА

ИНФОРМАТИКА

INFORMATICS

DOI:10.26104/NNTIK.2022.25.65.006

Сабитов Б.Р., Сейтказиева Н.С., Кашкароева А.А.

ТЕРЕҢ ОКУТУУ ТЕХНОЛОГИЯЛАРЫН КОЛДОНУУ МЕНЕН ӨСҮМДҮКТӨРДҮН ООРУЛАРЫН КЛАССИФИКАЦИЯЛООНУН ЭКИЛИК МИЛДЕТИ

Сабитов Б.Р., Сейтказиева Н.С., Кашкароева А.А.

БИНАРНАЯ ЗАДАЧА КЛАССИФИКАЦИИ БОЛЕЗНИ РАСТЕНИЙ С ПРИМЕНЕНИЕМ ТЕХНОЛОГИЙ ГЛУБОКОГО ОБУЧЕНИЯ

B. Sabitov, N. Seitkazieva, A. Kashkaroeva

BINARY PROBLEM OF PLANT DISEASE CLASSIFICATION USING DEEP LEARNING TECHNOLOGIES

УДК: 004.93

Терең окутуу технологияларынын жардамы менен помидор оорусунун классификациясынын ар кандай моделдери курулган. Бул өсүмдүктүн эң кеңири тараган оорулары үчүн бинардык классификация маселеси изилденген. Макалада алдын ала аныкталган категориялар боюнча помидор өсүмдүктөрүнүн жалбырак ооруларын аныктоо жана классификациялоо үчүн төрт терең нейрондук тармак моделдерин куруу талкууланат. Өсүмдүктүн түсү, текстурасы жана жалбыректын четтери сыяктуу морфологиялык өзгөчөлүктөрү да эске алынган. Документте томат ооруларын классификациялоо милдети изилденет, бул биздин республиканын кайсы гана аймагында болбосун көптөгөн фермерлер жана жеке ишкерлер үчүн актуалдуу жана маанилүү жана, акырында, Кыргыз Республикасынын бүткүл калкы үчүн. Бул макалада ар кандай вариациялар менен стандарттуу терең үйрөнүү моделдери киргизилет. Биз помидор жалбырактарынын грибоктук жана бактериялык козгогучтарынан келип чыккан Кыргыз Республикасынын аймактары үчүн кеңири таралган биотикалык ооруларды изилдедик. Сунушталган моделдин тактыгы 99,4% түздү.

Негизги сөздөр: *бинардык тапшырма, модель, терең үйрөнүү, өсүмдүктөрдүн оорулары, помидор, нейрон тармактары, тесттик сүрөт.*

С помощью технологий глубокого обучения построено различные модели классификации болезни томатов. Изучена задача бинарной классификации для наиболее распространенных болезней данного растения. В статье рассмотрены построение четырех моделей глубокой нейронной сети для обнаружения и классификации болезней листьев растений томата по заранее определенным категориям. Также учитывались морфологические признаки, такие как цвет, текстура и края листьев растения. В работе изучается задача классификации болезней томатов, которая является актуальной и важной для многих фермеров и частных предпринимателей любого региона нашей республики и, в конечном счете, для всего населения Кыргызской Республики. В этой статье представлены стандартные модели глубокого обучения с различными вариантами. Изучалось общие распространенные для регионов КР, биотические заболевания, вызываемые грибковыми и бактериальными патогенами листьев томатов. Точность предложенной модели составила 99,4%.

Ключевые слова: *бинарная задача, модель, глубокое обучение, болезни растений, томаты, нейронные сети, тестовое изображение.*

With the help of deep learning technologies, various models of classification of tomato disease have been built. The problem of binary classification for the most common diseases of this plant is studied. The article discusses the construction of four models of a deep neural network for the detection and classification of diseases of tomato plant leaves by predefined categories. Morphological features such as the color, texture and edges of the leaves of the plant were also taken into account. The paper studies the problem of classification of tomato diseases, which is relevant and important for many farmers and private entrepreneurs in any region of our republic and, ultimately, for the entire population of the Kyrgyz Republic. This article presents standard deep learning models with various options. We studied the common biotic diseases common to the regions of the Kyrgyz Republic caused by fungal and bacterial pathogens of tomato leaves. The accuracy of the proposed model was 99.4%.

Key words: *binary problem, model, deep learning, plant diseases, tomatoes, neural networks, test image.*

Введение. В работе изучается задача классификации болезней томатов, которая является актуальной и важной для многих фермеров и частных предпринимателей любого региона нашей республики и, в конечном счете, для всего населения КР. В каждом регионе Кыргызской Республики, широко распространено выращивание данной культуры для различных целей. Данный продукт является стратегическим и играет большую роль для решения задач продовольственной программы. В связи с загрязнением окружающей среды, неправильной агротехнической обработки и аномально неблагоприятных для выращивания данной культуры периодов, многие фермеры ежегодно теряют из-за болезни растений прибыль на урожайности. Теряется при этом качество продукции. Среди болезней томатов, особенно в летний период распространено для Чуйской, Ошской и Жалал-Абадской областей следующие болезни томатов: Бактериальное

пятно, Ранняя гниль, Фитотороз, Листовая плесень, Септориоз листовой пятнистости, Паутинный клещ, пятнистый паутинный клещ, Пятнистость помидоров, Мозаичный вирус, Вирус желтой курчавости листьев томатов. В связи с этим ранее диагностирования болезней томатов и своевременное принятие соответствующих мер, улучшает урожайность и прибыль на урожайность. Весьма оптимальным при этом является использование новых технологий диагностирования и прогнозирование болезней растений. Особенно актуальной являются технологии, связанные с использованием элементов искусственного интеллекта для обнаружения болезней растений. В данной работе с использованием глубокого обучения, являющиеся элементом искусственного интеллекта изучается задача распознавания болезней томатов. Будем строит модели бинарной классификации, томатов является ли он больным или здоровым.

Обзор научных публикаций и литератур. Различные исследователи использовали передовые технологии, такие как машинное обучение и архитектуры нейронных сетей, такие как сеть Inception V3, сеть VGG 16 и SqueezeNet, для создания автоматизированных систем обнаружения заболеваний. В них используются высокоточные методы выявления болезней растений на листьях томатов. Кроме того, исследователи предложили множество основанных на глубоком обучении решений для выявления и классификации заболеваний, как описано ниже в [1, 2, 3, 4].

Методология построения нейронных сетей и обсуждение результатов. Для идентификации больных и здоровых листьев томатов мы построим модели с помощью технологий глубокого обучения. Будем использовать общую архитектуру сверточных нейронных сетей. Детали сверточных слоев для каждого слоя будем изучать отдельно. Сначала мы изучим пакеты изображений для ввода нейронную сеть.

Для предварительной обработки данных изображений мы загружаем только необходимый пакет программного обеспечения для классификации изображений. Определяем далее каталоги расположения данных.

Листинг 1

Каталог данных больных и здоровых листьев томатов.

```
image_dir = 'E:/Dataset_PlantVillage/Tomato_Healthy_Unhealthy/'

healthy_images_dir = image_dir + 'Healthy/'
unhealthy_images_dir = image_dir + 'UnHealthy/'
```

В каталоге Healthy и UnHealthy мы будем хранить больные и здоровые листья томатов.

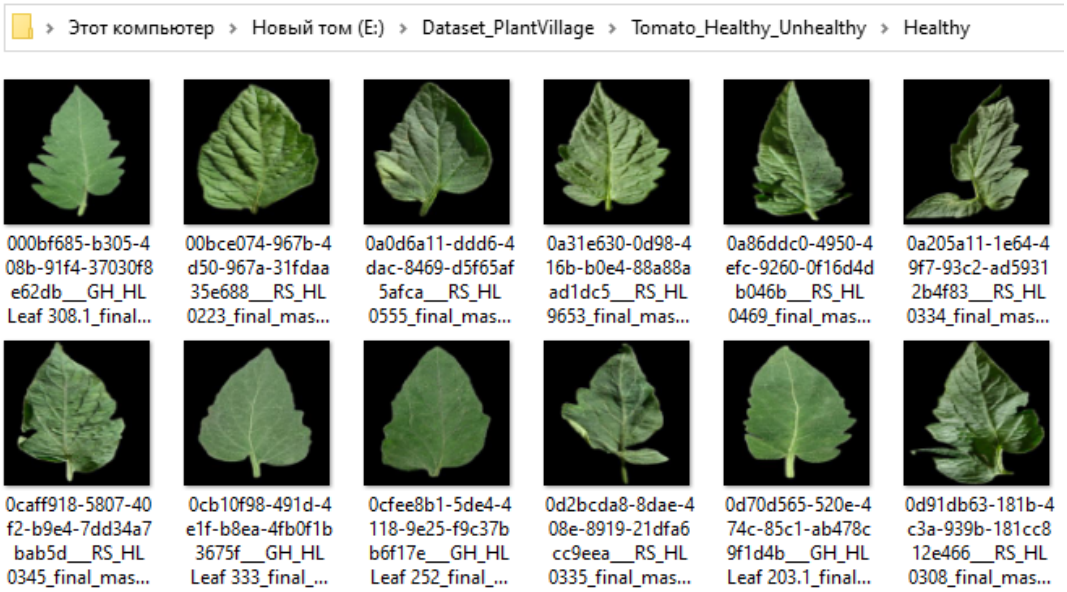


Рис. 1. Фрагмент из набора данных здоровых и больных томатов.

Для просмотра данных использованы специальные программы Opencv и пакеты Python, которые позволяют просмотр случайных изображений из набора данных.

```
plt.imshow( healthy_images[12])
```

```
<matplotlib.image.AxesImage at 0x1f7eb4851f0>
```

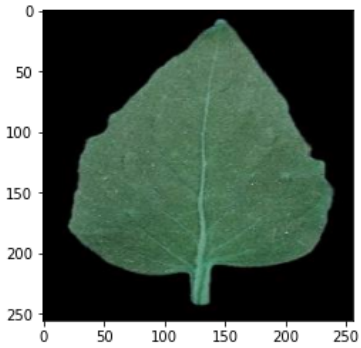


Рис. 2. Здоровые листья томатов.

После преобразования всех здоровых и больных изображений в массив numpy, после которого каждый из них преобразуется в четырехмерный тензор (806,256,256,3) и (2074, 256, 256, 3) соответственно. Теперь для каждого класса присвоим метку, здоровые листья 0 и больные листья 1.

Вот код

y_labels = [0] * len(healthy_images) + [1] * len(unhealthy_images)

Всего меток len(y_labels) = 3665. Сочетание здоровых и больных растений all_images_np = np. concatenate ((healthy_images_np, unhealthy_images_np), axis = 0) all_images_np.shape (3665, 256, 256, 3).

Таким образом, мы имеем четырёхмерный тензор. Первый элемент 3665это общее количество изображений,256x256 размер изображения в пикселях и число 3 означает, что изображение цветное. Другими словами, пакет со 3665 цветных изображений имеющих размер 256x256, можно сохранить в тензоре с формой (3665, 256, 256, 3).

Создание моделей для классификации здоровых или нездоровых листьев помидоров

import keras
import tensorflow **as** tf
from keras **import** backend **as** K
Импортируем необходимые строительные блоки нейронной сети
from keras.models **import** Sequential
from keras.layers **import** Conv2D, MaxPooling2D, Flatten, Dense, Activation, Dropout
from keras.layers.advanced_activations **import** LeakyReLU
Нормализация входных данных
X_train_norm = (X_train/255)-0.5
X_test_norm = (X_test/255)-0.5
Ниже приведены модели глубокой сети

Модель 1:
** Архитектура **
- Conv2D -> MaxPool -> Conv2D -> MaxPool -> Dense -> Dense -> Sigmoid

** Оптимизатор **
- Стохастический градиентный спуск
- Размер партии = 32
- Эпоха = 20
- model = Sequential ()

Model: "sequential"

_____

Layer (type) Output Shape Param #

==========

conv2d (Conv2D) (None, 254, 254, 8) 224

leaky_re_lu (LeakyReLU) (None, 254, 254, 8) 0

max_pooling2d (MaxPooling2D (None, 84, 84, 8) 0)

conv2d_1 (Conv2D) (None, 82, 82, 16) 1168

leaky_re_lu_1 (LeakyReLU) (None, 82, 82, 16) 0

max_pooling2d_1 (MaxPooling (None, 27, 27, 16) 0
2D)
flatten (Flatten) (None, 11664) 0

dense (Dense) (None, 16) 186640

leaky_re_lu_2 (LeakyReLU) (None, 16) 0

dense_1 (Dense) (None, 1) 17

activation (Activation) (None, 1) 0
Total params: 188,049
Trainable params: 188,049
Non-trainable params: 0

```
model.compile(optimizer='sgd',
              loss = 'binary_crossentropy',
              metrics = ['accuracy'])
BATCH_SIZE = 32
EPOCHS = 20
history = model.fit (
X_train_norm,
y_train,
batch_size=BATCH_SIZE,
epochs=EPOCHS,
validation_data= (X_test_norm, y_test),
verbose=1)
Epoch 1/20
72/72 [==============================] - 39s 506ms/step - loss: 0.6477 - accuracy: 0.6348 - val_loss:
0.5968 - val_accuracy: 0.7575
……..
Epoch 20/20
72/72 [==============================] - 20s 280ms/step - loss: 0.0841 - accuracy: 0.9690 - val_loss:
0.1238 - val_accuracy: 0.9592
```

```
plot_train_val_accuracy(history.history)
```

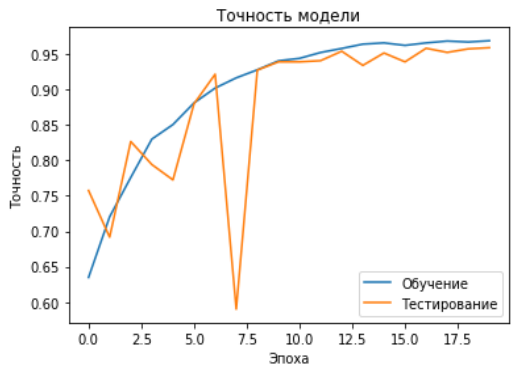


Рис. 3. График точности модели на данных обучения и валидации.

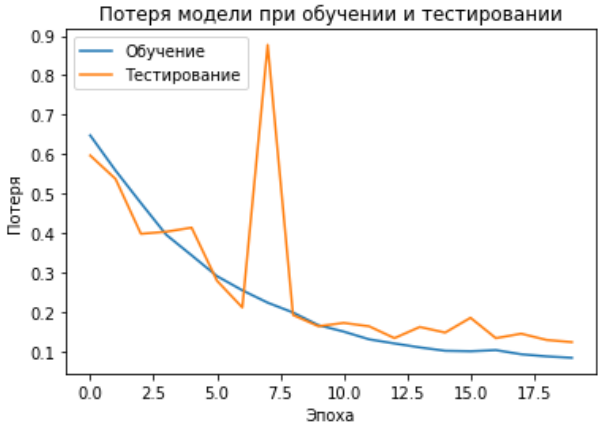


Рис. 4. График ошибки модели на данных обучения и валидации.

Создадим теперь другой модель и добавим для оптимизации модели оптимизатор Adam.
import pickle
pickle.dump(history.history, open('history_tomato_model_1.pkl', 'wb'))
Построим модель с некоторыми изменениями.

Модель 2: Работа с вариациями потери GD

** Архитектура **
Conv2D -> MaxPool -> Conv2D -> MaxPool -> Плотный -> Плотный -> Сигмоид
** Оптимизатор **-Adam, Размер партии = 32 Эпоха = 20. Компилируем модель и определяем параметры нейронной сети.

```
model.compile(optimizer='adam',
    loss = 'binary_crossentropy',
    metrics = ['accuracy'])
BATCH_SIZE = 32
EPOCHS = 20
```

Обучим модель с учетом введенных улучшенных параметров сети.

```
history = model.fit (
    X_train_norm,
    y_train,
    batch_size=BATCH_SIZE,
    epochs=EPOCHS,
    validation_data= (X_test_norm, y_test),
    verbose=1)
```

Epoch 1/20
72/72 [==============================] - 37s 504ms/step - loss: 0.5192 - accuracy: 0.7475 - val_loss: 0.3761 - val_accuracy: 0.8242

…….....

Epoch 20/20
72/72 [==============================] - 20s 280ms/step - loss: 0.0016 - accuracy: 1.0000 - val_loss: 0.1616 - val_accuracy: 0.9583

Создадим график точности и ошибки модели на данных обучения и валидации-проверки.

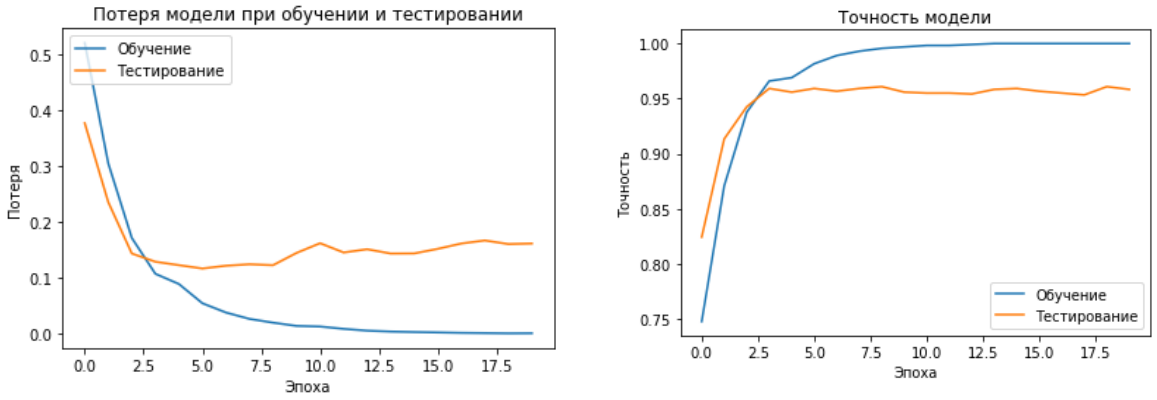


Рис. 5. Графики точности и ошибки модели на данных обучения и валидации-тестирования.

Следующая строка кода сохраняет наш модель.

pickle.dump(history.history, open('history_tomato_model_2.pkl', 'wb'))

Построенный модель 2, как видно из Рис.5 является переобученным и не обладает обобщающим свойствам. Точность и ошибка модели на данных обучения и валидации сильно различаются. В таких случаях применяется технологии регуляризации. В некоторых слоях нейронной сети мы деактивируем некоторые нейроны с помощью Dropout.

Модель 3: Работа с переобучением. Метод регуляризации.

** Архитектура **

Conv2D -> MaxPool -> Conv2D -> MaxPool -> Dense -> Dropout -> Dense -> Dropout -> (Dense -> Sigmoid)

** Optimizer **- Адам, Размер партии = 32, Эпоха = 20

- Adam
- Batch size = 32
- Epoch = 20
- model.compile(optimizer='adam',
  loss = 'binary_crossentropy',
  metrics = ['accuracy'])

```
BATCH_SIZE = 32
EPOCHS = 20
    history = model.fit (
    X_train_norm,
    y_train, # prepared data
    batch_size=BATCH_SIZE,
    epochs=EPOCHS,
    validation_data= (X_test_norm, y_test),
    verbose=1
)
```

Epoch 1/20

72/72 [==============================] - 36s 483ms/step - loss: 0.6391 - accuracy: 0.6514 - val_loss: 0.4499 - val_accuracy: 0.8075

……

Epoch 20/20

72/72 [==============================] - 20s 278ms/step - loss: 0.0281 - accuracy: 0.9913 - val_loss: 0.1887 - val_accuracy: 0.9700

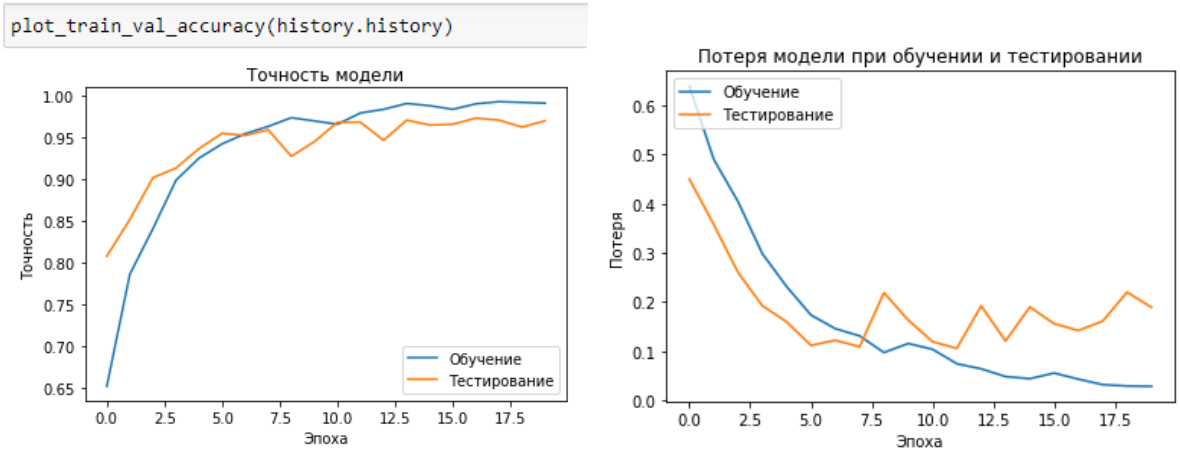


Рис. 5. Графики точности и ошибки модели на данных обучения и валидации-тестирования с прореживанием.

Мы улучшили точность модели с использованием прореживания с помощью деактивации некоторых нейронов промежуточных слоев нейронной сети. Для дальнейшего улучшения точности модели мы обучим нашу модель с более глубоким обучением нейронной сети.

Модель 4: Повышение точности за счет более глубокой сети

** Архитектура **

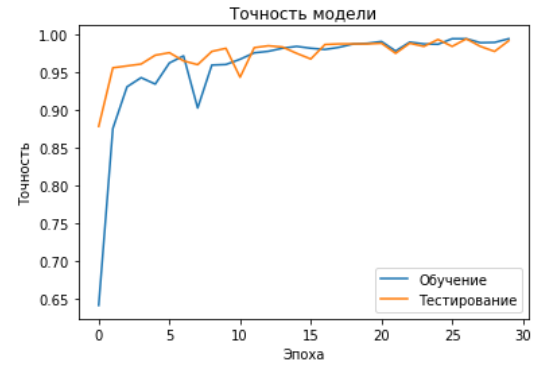
Conv2D -> MaxPool -> Conv2D -> MaxPool -> Conv2D -> MaxPool -> Dense -> Dropout -> Dense -> Dropout -> (Dense -> Sigmoid) ** Оптимизатор **

Адам Размер партии = 32 Эпоха = 30 (изменение)

```
K.clear_session() # clear default graph
model = Sequential ()
model.add(Conv2D(filters=8,              kernel_size=(3,3), strides=1,input_shape=image_shape))
model.add(LeakyReLU(0.1))
model.add(MaxPooling2D(pool_size=(3, 3)))
model.add(Conv2D(filters=16,             kernel_size=(3,3), strides=1, input_shape=image_shape))
model.add(LeakyReLU(0.1))
model.add(MaxPooling2D(pool_size=(3, 3)))
model.add(Conv2D(filters=32,             kernel_size=(3,3), strides=1, input_shape=image_shape))
model.add(LeakyReLU(0.1))
model.add(MaxPooling2D(pool_size=(3, 3)))
model.add(Flatten())
model.add(Dense(32))
model.add(LeakyReLU(0.1))
model.add(Dropout(0.5))
model.add(Dense(8))
model.add(LeakyReLU(0.1))
model.add(Dropout(0.5))model.add(Dense(1))
model.add(Activation('sigmoid'))
model.compile(optimizer='adam',
      loss = 'binary_crossentropy',
      metrics = ['accuracy'])
BATCH_SIZE = 32
EPOCHS = 30
history = model.fit(
   X_train_norm,
   y_train, # prepared data
   batch_size=BATCH_SIZE,
   epochs=EPOCHS,
   validation_data= (X_test_norm, y_test),
```

```
    verbose=1
)
Epoch 1/30
78/78 [==============================] - 21s 249ms/step - loss: 0.6083 - accuracy: 0.6414 - val_loss: 0.3437 - val_accuracy: 0.8783
……..……..
Epoch 30/30
78/78 [==============================] - 17s 220ms/step - loss: 0.0204 - accuracy: 0.9943 - val_loss: 0.0605 - val_accuracy: 0.9917
```

```
plot_train_val_accuracy(history.history)
```




```
plot_train_val_loss(history.history)
```

Рис. 5. Улучшение точности модели с глубоким обучением нейронной сети.

В результате мы получили высокую точность модели намного превышающий все предыдущие модели. Для дальнейшего развертывания модели в веб приложение, сохраним модель следующим образом.

```
import pickle
pickle.dump( history.history, open('history_tomato_model_4.pkl', 'wb') )
```

Таким образом, мы с улучшенной моделью с более глубоким обучением нейронной сети (30 эпох) достигли точности модели в 99.43%. Получилось неплохо. Теперь изучим скрытые слои нейронной сети. Визуализируем скрытые слои нейронных сетей. Рассмотрим очень важный вопрос, как нейронная сеть распознает элементы изображения. Возьмем тестовое изображение.

img_path = 'E:/Dataset_PlantVillage/color/Tomato___Tomato_Yellow_Leaf_Curl_Virus/0a14b65b-2e45-4bed-be45-c482a40a4f7c___UF.GRC_YLCV_Lab 02090.JPG'. Вот ее изображение.

```
import matplotlib.pyplot as plt

plt.imshow(img_tensor[0])
plt.show()
```

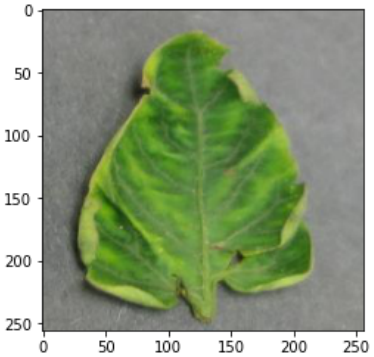


Рис. 6. Тестовое изображение.

Вот визуализация четвертого скрытого слоя нейронной сети

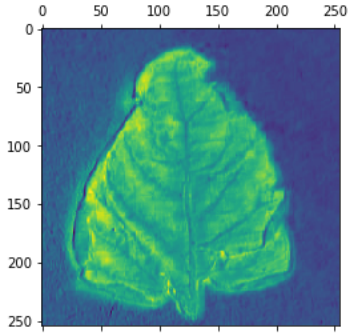


Рис. 7. Результат четвертого скрытого слоя сети.

Здесь мы представляем шестой канал скрытого слоя нейронной сети.

```
import matplotlib.pyplot as plt

plt.matshow(first_layer_activation[0, :, :, 6], cmap='viridis')
plt.show()
```

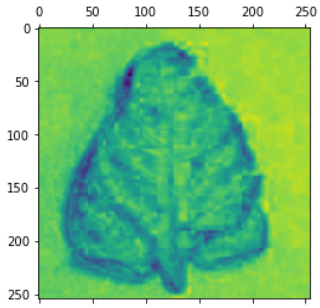


Рис. 8. Результат шестого скрытого слоя нейронной сети.

Восьмиканальное изображение всех скрытых слоев нейронной сети.

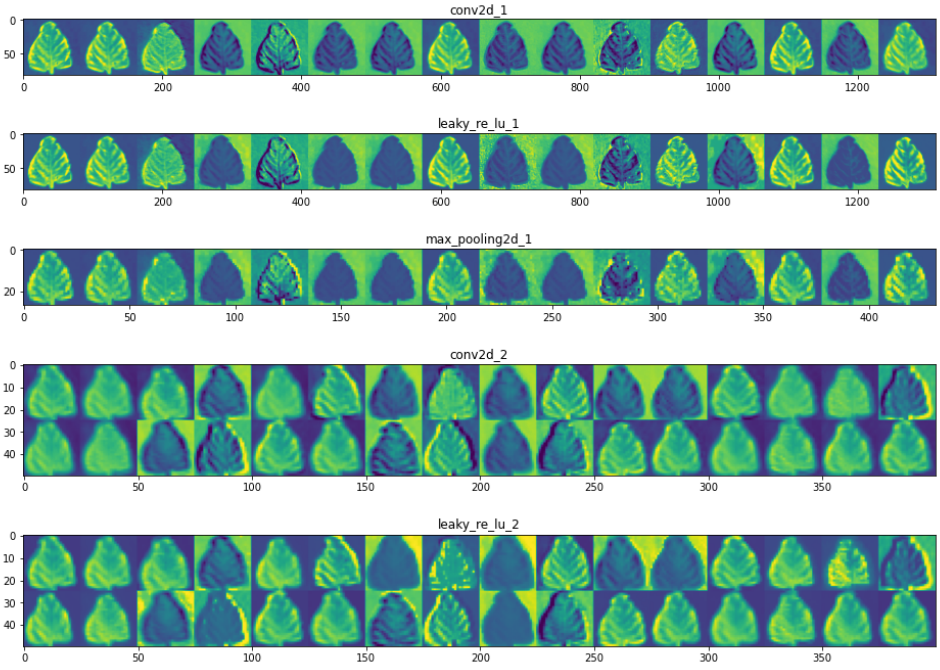


Рис. 9. Просмотр распознавания изображения в скрытых слоях нейронной сети.

В результате было получены следующие интересные факты промежуточных слоев нейронной сети. На первых слоях действует фактор распознавания границ как единый ансамбль. На этих слоях активации все еще сохраняются почти все информации, присутствующие в исходном изображении. Это видно из изображения листья томатов из тестового набора. При активации более глубоких скрытых слоев активации становятся все более абстрактными и менее интерпретируемыми визуально. Они начинают кодировать более высокий уровень такие понятия, как «края листьев», узоры или «пятна на листьях». Презентации более высокого уровня несут все меньше информации о визуальном содержании объекта изображение. Нейронная сеть данного уровня все больше содержит информации, относящейся не конкретному изображению, а к классу изображения, которому он относится.

Заключение. В статье обсуждались различные модели бинарной классификации болезней томатов с 3665 изображениями на основе технологий глубокой нейронной сети. Для обнаружения и классификации болезней листьев растений томата были известны категории болезней, которые наиболее охватывает регионы нашей страны. В работе были созданы 4 модели глубокого обучения с вариантами, обсуждались биотические заболевания, вызываемые грибковыми и бактериальными патогенами и распространенная болезнь ранняя гниль томатов. Повышение точности за счет более глубокой обучения сети модели составила более 99%.

Литература:

1. Hasan M., Tanawala B., Patel K.J. Deep learning precision farming: Tomato leaf disease detection by transfer learning; In Proceeding of the 2nd International Conference on Advanced Computing and Software Engineering (ICACSE); Sultanpur, Inida. 8–9 February 2019. [Google Scholar]
2. Adhikari S., Shrestha B., Baiju B., Kumar S. Tomato plant diseases detection system using image processing; Proceedings of the 1st KEC Conference on Engineering and Technology; Laliitpur, Nepal. 27 September 2018; pp. 81-86. [Google Scholar]
3. Sabrol H., Satish K. Tomato plant disease classification in digital images using classification tree; Proceedings of the International Conference on Communication and Signal Processing (ICCSP); Melmaruvathur, India. 6-8 April 2016; pp. 1242-1246. [Google Scholar]
4. Salih T.A. Deep Learning Convolution Neural Network to Detect and Classify Tomato Plant Leaf Diseases. *Open Access Libr.J.* 2020;**7**:12.doi: 10.4236/oalib.1106296.[CrossRef] [Google Scholar].
5. Сейтказиева Н.С., Токтогулова Г.А., Ибраева А.Т. Внедрение в образовательный процесс изучения информатики и информационно-коммуникационные технологии сервиса learningapps.org и google sites. / Наука, новые технологии и инновации Кыргызстана. 2020. №. 12. С. 239-243.