УДК 004.032.26

Использование нейронных сетей для определения результатов тестов на Covid-19 с использованием библиотеки Keras

Ульянов Егор Андреевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
Целью данной статьи было, применяя машинное обучение, создать в программе Jupyter Notebook (Anaconda) нейронную сеть по определению вердикта SARS-CoV-2 по анализам, обучив её на тестовом наборе данных с точными результатами заболевания. Для этого использовались метод эксперимента по созданию нейронной сети и сравнение полученных результатов с реальными. Итогом исследования стала полностью рабочая обученная нейронная сеть, определяющая по результатам анализов заражён человек или нет с точностью 79,49%.
Ключевые слова: нейронная сеть, SARS-CoV-2, Anaconda, Keras

Using neural networks to determine test results for Covid-19 using Keras library

Ulianov Egor Andreevich
Sholom-Aleichem Priamursky State University
Student

Abstract
The purpose of this article was, using machine learning, to create a Jupyter Notebook (Anaconda) neural network for determining the SARS-CoV-2 verdict by analysis, training it on a test dataset with accurate disease results. For this, we used the method of experiment to create a neural network and compare the results with real ones. The result of the study was a fully trained neural network that determines whether a person is infected or not based on the results of analyzes with an accuracy of 79.49%.
Keywords: neural network, SARS-CoV-2, Anaconda, Keras

1 Введение

1.1 Актуальность исследования

SARS-CoV-2 в 2020 году стал главной проблемой для всего человечества. Из-за его стремительного распространения не осталось ни одной страны, где он не был бы выявлен, но хотя и на момент написания данной статьи лекарство всё ещё разрабатывается, общие симптомы уже поддаются лечению. Но возникает другая проблема – выявить наличие инфекции у человека, специализированные клиники проводят множество

тестов каждый день, но моментально дать результат не могут из-за обилия данных. Которые с каждым днём из-за всё увеличивающегося количества зараженных неуклонно растут.

Помочь в этом могла бы специально обученная нейронная сеть, которая собирая информацию о результате теста могла бы точно сказать болен человек или результат отрицательный. А также насколько точной может быть такая нейронная сеть, ведь от этого будут зависеть жизни?

1.2 Обзор исследований

Исследованиями в данной теме занимались следующие авторы. В.В. Евсюков, Т.В. Свиридова, и Е.Р. Богатенко в своей работе «Искусственным интеллект и коронавирус Covid-19» [1] рассмотрели, как противодействовать его распространению. «Разработка нейросетевой модели для мониторинга заболеваемости и прогнозирования эффективности противоэпидемических мер» была показана Н.В. Сухановой [2], модель позволяет провести прогноз эпидемической обстановки на перспективу и оценку эффективности противоэпидемических мер. А.А. Арбузова выявила способ «Диагностики легочных заболеваний с помощью нейронных сетей» [3], как один из симптомов Covid-19. «Нейросетевая модель детекции признаков поражения легких, ассоциированных с Covid-19, на аксиальных срезах нативной компьютерной томографии грудной клетки» [4] была разработана целой группой ученых из таких людей как П.В. Гаврилов, К.С. Щеткин, Р.М. Залялов, У.А. Смольникова, А.В. Бельских, Д.С. Блинов, А.А. Азаров, Е.В. Блинова, П.К. Яблонский. Также В.В. Цветков, И.И. Токин, Д.А. Лиознов, Е.В. Венев и А.Н. Куликов с помощью машинного обучения провели «Прогнозирование длительности стационарного лечения пациентов с Covid-19» [5]. Как «Свёрточная нейронная сеть использует изображения рентгенографии грудной клетки для идентификации COVID-19» было продемонстрировано D.Muralia, E.Bhuvaneswarib, S.Parvathic и A.N.Sanjeev Kumard [6]. Sergio Varela-Santos и Patricia Melin показали другой подход в своей работе «Новый подход к классификации коронавируса COVID-19 на основе его проявления на рентгеновских снимках грудной клетки с использованием текстурных особенностей и нейронных сетей» [7]. С помощью «Обнаружения пациентов с COVID-19 на основе нечеткого механизма вывода и глубокой нейронной сети» [8] смогли добиться результатов ученые Warda M. Shabana, Asmaa H. Rabieb, Ahmed I. Salehb и M.A. Abo-Elsoud. Группа исследователей Chen-Xin Wu, Min-Hui Liao, Mumtaz Karatas, Sheng-Yong Chen, Yu-Jun Zheng занималась «Планированием нейронной сети в реальном времени производства масок для оказания неотложной медицинской помощи во время COVID-19» [9]. «Прогнозирование числа случаев COVID-19 в Индии на основе рекуррентной нейронной сети» [10] исследовали K. Shyam Sunder Reddy, Y.C.A. Padmanabha Reddy, Ch. Mallikarjuna Rao.

1.3 Цель исследования

Цель исследования – применяя машинное обучение, создать в программе Jupyter Notebook (Anaconda) нейронную сеть по определению вердикта SARS-CoV-2 по анализам, обучив её на тестовом наборе данных с точными результатами заболевания.

2. Материалы и методы

2.1 Данные

Данные были взяты с сайта Kaggle [11]. Этот набор данных содержит анонимные данные о пациентах из больницы Israelita Albert Einstein в Сан-Паулу, Бразилия, и у которых были собраны образцы для проведения ОТ-ПЦР SARS-CoV-2 и дополнительных лабораторных тестов во время посещения больницы. Все данные были обезличены. Все клинические данные были стандартизированы, чтобы иметь нулевое среднее значение и стандартное отклонение.

2.2 Методы исследования

Для создания нейронной сети будем использовать программное обеспечение Anaconda [12], так как она обладает огромным числом настраиваемых модулей для машинного обучения. К тому же основывается на языке Python, зарекомендовавшем себя в области нейронных сетей.

Для создания использовалась Keras [13] – открытая нейросетевая библиотека на языке Python, создана на быструю и простую работу с сетями машинного обучения, легко расширяема и модульная, содержит в себе многочисленные реализации для создания нейронных сетей.

3 Результаты и дискуссия

Начнём создание нейронной сети. После того как были скачаны данные, в окне Anaconda подгрузим необходимые библиотеки и выгрузим из Excel всю информацию в переменную Dataset (рис. 1).

```
import pandas as pd
import numpy as np

dataset = pd.read_excel('C:/AnacondProject/dataset.xlsx')
```

Рисунок 1. Импорт библиотек и выгрузка данных

Набор данных содержит 5644 строки и 111 столбцов, но при просмотре файла было выявлено, что большая часть данных отсутствует. Таким образом, было найдено оптимальное количество из 191 строки и 42 столбцов, при этом большинство значений были заполнены (хотя некоторые значения все еще отсутствовали). Поэтому выгружаем только данные с которыми можно работать (рис.2).

```
dataset = dataset.dropna(axis=0, thresh=50)
dataset = dataset.dropna(axis=1, thresh=160)

X = dataset.iloc[:, 1:42]
X = X.drop(X.columns[1], axis =1)
y = dataset.iloc[:, 2]
```

```
from sklearn.preprocessing import LabelEncoder
labelencoder_y = LabelEncoder()
y = labelencoder_y.fit_transform(y)
```

Рисунок 2. Процедура очистки набора данных от малоинформативных

Так как даже у малого числа выбранных данных отсутствовали некоторые численные значения, было решено ввести медианные значения для каждого столбца, хотя это могло привести к искажению прогнозируемых значений, но отсутствовало не более 20% значений, поэтому это не сильно скажется на результате (рис.3).

```
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(missing_values = np.nan, strategy = "median")
imputer = imputer.fit(X.iloc[:, 4:17])
X.iloc[:, 4:17] = imputer.transform(X.iloc[:, 4:17])
imputer = imputer.fit(X.iloc[:, 35:])
X.iloc[:, 35:] = imputer.transform(X.iloc[:, 35:])
detection = X.iloc[:, 18:35]
```

Рисунок 3. Заполнение пропущенных значений медианными

Было также и много столбцов с пропущенными значениями, которые состояли из “detected” или “not_detected”. Анализируя их, было обнаружено, что большинство проверок были помечены как “not_detected”, независимо от того, была ли диагностика Covid-19 положительной или нет, поэтому это значение было решено сделать по умолчанию (рис.4).

```
negDetectedneg, negDetectedpos, posDetectedneg, posDetectedpos = 0, 0, 0, 0
for line in range(191):
    initial_row = 18
    for row in range(17):
        if X.iloc[line,(initial_row + row)] == 'not_detected':
            if y[line] == 0:
                negDetectedneg = negDetectedneg + 1
            else:
                negDetectedpos = negDetectedpos + 1
        elif X.iloc[line,(initial_row + row)] == 'detected':
            if y[line] == 0:
                posDetectedneg = posDetectedneg + 1
            else:
                posDetectedpos = posDetectedpos + 1

print('Отрицательно отмеченный как "not_detected": ', negDetectedneg)
print('Положительно отмеченный как "not_detected": ', negDetectedpos)
print('Отрицательно отмеченный как "detected": ', posDetectedneg)
print('Положительно отмеченный как "detected": ', posDetectedpos)
```

Рисунок 4. Выявление значения по умолчания

Таким образом из 3043 ячейки часто встречается действительно “not_detected” (рис. 5).

```
Отрицательно отмеченный как "not_detected":  2374
Положительно отмеченный как "not_detected":  576
Отрицательно отмеченный как "detected":  91
Положительно отмеченный как "detected":  2
```

Рисунок 5. Результат проверки ячеек

Поэтому заполняем все пустые ячейки этим значением, чтобы доработать данные перед передачей их нейронной сети (рис.6).

```
import math
for col in range(17):
    for line in range(191):
        if type(detection.iloc[line,col]) == float:
            if math.isnan(detection.iloc[line,col]):
                detection.iloc[line,col] = 'not_detected'

for col in range(17):
    X.iloc[:, (18 + col)] = detection.iloc[:, col]
```

Рисунок 6. Замена пустых ячеек на “not_detected”

Для того, чтобы полностью привести данные к готовым для обработки нейронной сетью, необходимо преобразовать текстовые поля, поэтому с помощью команды LabelEncoder() они преобразуются (рис.7).

```
for col in range(17):
    labelencoder_X = LabelEncoder()
    X.iloc[:, (18 + col)] = labelencoder_X.fit_transform(X.iloc[:, (18 + col)])
```

Рисунок 7. Форматирование оставшихся категориальных данных

Поскольку существует большое число библиотек для работы с искусственными нейронными сетями, для получения наилучших результатов было решено использовать Keras, ведь помимо прочего он также является и самым простым в использовании (рис.8). В Keras одним из способов для обучения нейронной сети и построения её модели используются слои (layers), сама модель в итоге – это граф слоёв. Функция Sequntial() создаёт пустую модель для обучения.

Сам слой настраивается следующими параметрами:

- Kernel_initializer – схема инициализация, создающие веса слоя. «uniform» - случайное распределение.
- Activation – установка функции активации для слоя. Здесь указывается имя функции или объект. Она может принимать значения различные значения. Relu – расшифровывается как Rectified Linear Unit (выпрямленный линейный блок). Sigmoid - Сигмоидно-взвешенная линейная функция.

Функция «compile» позволяет настроить процесс обучения модели и принимает на вход:

- Optimizer – определяет процедуру обучения, в него передаются оптимизаторы из модуля tf.keras.optimizers. Adam - adaptive moment estimation (метод Адаптивной Оценки Моментов).
- Loss – функция, которая будет минимизироваться при обучении. binary_crossentropy – двоичная кросс-энтропия.
- Metrics – функция для мониторинга обучения. accuracy — доля правильных ответов алгоритма

Наконец, функция «fit» сигнализирует о начале обучения модели, происходит это для фиксированного количества эпох (итераций в наборе данных).

```
#Разделение набора данных на обучающий набор и тестовый набор
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)

import keras
from keras.models import Sequential
from keras.layers import Dense

#Инициализация нейронной сети
classifier = Sequential()

#Добавление входного слоя и первого скрытого слоя
classifier.add(Dense(units = 20, kernel_initializer = 'uniform', activation = 'relu', input_dim = 40))

#Добавление второго скрытого слоя
classifier.add(Dense(units = 20, kernel_initializer = 'uniform', activation = 'relu'))

#Добавление выходного слоя
classifier.add(Dense(units = 1, kernel_initializer = 'uniform', activation = 'sigmoid'))

#Компиляция перестроенного нейрона
classifier.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'])

#Подгонка нейронной сети к обучающей выборке
classifier.fit(X_train, y_train, batch_size = 10, epochs = 200)
```

```
16/16 [==============================] - 0s 750us/step - loss: 5.3804e-04 - accuracy: 1.0000
Epoch 193/200
16/16 [==============================] - 0s 688us/step - loss: 5.3203e-04 - accuracy: 1.0000
Epoch 194/200
16/16 [==============================] - 0s 750us/step - loss: 5.1755e-04 - accuracy: 1.0000
Epoch 195/200
16/16 [==============================] - 0s 688us/step - loss: 5.1426e-04 - accuracy: 1.0000
Epoch 196/200
16/16 [==============================] - 0s 750us/step - loss: 4.9792e-04 - accuracy: 1.0000
Epoch 197/200
16/16 [==============================] - 0s 688us/step - loss: 4.9291e-04 - accuracy: 1.0000
Epoch 198/200
16/16 [==============================] - 0s 688us/step - loss: 4.8170e-04 - accuracy: 1.0000
Epoch 199/200
16/16 [==============================] - 0s 750us/step - loss: 4.7324e-04 - accuracy: 1.0000
Epoch 200/200
16/16 [==============================] - 0s 750us/step - loss: 4.6379e-04 - accuracy: 1.0000

<tensorflow.python.keras.callbacks.History at 0x1bee4ca9940>
```

Рисунок 8. Подключение библиотека Keras и обучение нейронной сети

Так как нейронная сеть обучена, теперь требуется проверить её возможности для создания предсказываемых значений (рис.9).

```
#Прогнозирование результатов набора тестов
y_pred = classifier.predict(X_test)
y_pred = (y_pred > 0.5)
```

Рисунок 9. Подгонка прогноза тестового набора данных в модель нейронной сети

В последнюю очередь для проверки необходимо сравнить предсказываемые значения нейронной сети с настоящими данными (рис.10). Это осуществляется с помощью команды accuracy_score модуля Sklearn.Mentrics. Это классификационная оценка точности, функция вычисляет точность соответствия предсказанных наборов результатов действительному.

```
from sklearn.metrics import accuracy_score
print('Точность: %.2f%%' % (accuracy_score(y_test, y_pred)*100))
```

```
Точность: 79.49%
```

Рисунок 10. Точность предсказаний нейронной сети

Таким образом, созданная и обученная нейронная сеть оказалась точной в 79.49% значениях и в них точно предсказывает результат теста на SARS-CoV-2 у людей.

4 Выводы

Этот эксперимент вносит вклад в растущий корпус исследований по SARS-CoV-2, показывающих, что использование нейронных сетей в данной области необходимо, так как может значительно уменьшить нагрузку с людей при проверке результатов анализов, ведь всё будет осуществляться автоматически.

Ко всему прочему дальнейшая разработка созданной модели нейронной сети может привести к ещё большей точности, так как существует большая вероятность того, что неточности возникли только из-за отсутствующих данных, замененных на медианные числа и значения по умолчанию.

Библиографический список

1. Евсюков В.В., Свиридова Т.В., Богатенко Е.Р. Искусственным интеллект и коронавирус Covid-19 // Вестник Тульского филиала Финуниверситета. 2020. № 1. С. 295-297.
2. Суханова Н.В. Разработка нейросетевой модели для мониторинга заболеваемости и прогнозирования эффективности противоэпидемических мер // Вестник Брянского государственного технического университета. 2020. № 10 (95). С. 42-50.
3. Арбузова А.А. Диагностика легочных заболеваний с помощью нейронных сетей // В сборнике: Математическое и компьютерное моделирование естественно-научных и социальных проблем. Материалы XIV Международной научно-технической конференции молодых специалистов, аспирантов и студентов. Под редакцией И.В. Бойкова. 2020. С. 185-189.
4. Гаврилов П.В., Щеткин К.С., Залялов Р.М., Смольникова У.А., Бельских А.В., Блинов Д.С., Азаров А.А., Блинова Е.В., Яблонский П.К. Нейросетевая модель детекции признаков поражения легких, ассоциированных с Covid-19, на аксиальных срезах нативной компьютерной томографии грудной клетки // Медицинский альянс. 2020. Т. 8. № 2. С. 6-13.
5. Цветков В. В., Токин И. И., Лиознов Д. А., Венев Е. В., Куликов А. Н.

Прогнозирование длительности стационарного лечения пациентов с Covid-19 // Медицинский совет. 2020. № 17. С. 82-90.
6. D. Murali, E. Bhuvaneswari, S. Parvathi, A.N. Sanjeev Kumar Convolutional neural network use chest radiography images for identification of COVID-19 // Materials Today: Proceedings, 2020
7. Sergio Varela-Santos, Patricia Melin A new approach for classifying coronavirus COVID-19 based on its manifestation on chest X-rays using texture features and neural networks // Information Sciences, Volume 545, 2021, Pages 403-414
8. Warda M. Shaban, Asmaa H. Rabie, Ahmed I. Saleh, M.A. Abo-Elsoud Detecting COVID-19 patients based on fuzzy inference engine and Deep Neural Network // Applied Soft Computing, 2020
9. Chen-Xin Wu, Min-Hui Liao, Mumtaz Karatas, Sheng-Yong Chen, Yu-Jun Zheng Real-time neural network scheduling of emergency medical mask production during COVID-19 // Applied Soft Computing, Volume 97, Part A, 2020
10. Shyam Sunder Reddy K., Y.C.A. Padmanabha Reddy, Ch. Mallikarjuna Rao Recurrent neural network based prediction of number of COVID-19 cases in India // Materials Today: Proceedings, 2020
11. Diagnosis of COVID-19 and its clinical spectrum // Kaggle URL: https://www.kaggle.com/einsteindata4u/covid19/ (дата обращения: 28.11.2020).
12. The World's Most Popular Data Science Platform // Anaconda URL: https://www.anaconda.com/ (дата обращения: 28.11.2020).
13. Keras: the Python deep learning API // URL: https://keras.io/ (дата обращения: 28.11.2020).