

ГОСУДАРСТВЕННОЕ УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «ВЫСОКОВСКАЯ
СРЕДНЯЯ ШКОЛА» КАМЕНЕЦКОГО РАЙОНА

Секция «Компьютерные науки и программирование»

**«HEART ASSIST – ДИАГНОСТИЧЕСКАЯ СИСТЕМА
АНАЛИЗА ЗВУКОВ СЕРДЦЕБИЕНИЯ»**

Кондратюк Мария Владимировна, 10 класс
Сацута Дмитрий Владимирович, 10 класс

Лешкевич Александр Николаевич, учитель
математики и информатики

г. Высокое, 2020 г.

Оглавление

Введение.....	3
Основная часть ..	4
1. Методы классификации звуковых файлов	4
2. Программирование нейронной сети	5
3. Обучение нейронной сети.....	7
4. Тестирование нейронной сети.....	9
5. Разработка и сборка оборудования для проведения записи и диагностики звуков сердцебиения.....	10
6. Тестирование собранной системы.....	11
Заключение	13
Список использованных источников	14

Введение

Одной из самых актуальных проблем научной медицины и практического здравоохранения начала XXI века являются болезни системы кровообращения. По данным Всемирной организации здравоохранения основной причиной смертности в мире являются сердечно-сосудистые заболевания. Эти болезни занимают ведущее место среди всех причин смерти в большинстве экономически развитых странах мира. В нашей стране болезни системы кровообращения также занимают первое место среди причин смертности и инвалидности. Часто человек даже не подозревает, что у него есть проблемы с сердечно-сосудистой системой. Важным звеном в лечении и профилактике заболеваний сердца является ранняя диагностика его работы.

В настоящее время на помощь врачам приходят компьютерные программы, помогающие проводить первичные диагностики заболеваний. Для диагностики работы сердца было создано программное обеспечение «IHeart Assist – диагностическая система анализа звуков сердцебиения» (ИНА).

Цель работы: исследование различных методов и подходов анализа звуков сердцебиения и создание диагностической системы ИНА.

Задачи:

- 1) изучить актуальные подходы к обработке звуковых файлов на компьютере;
- 2) применить различные методы классификация для анализа звуков сердцебиения;
- 3) разработать диагностическую системы ИНА для быстрой и эффективной диагностики отклонений работы сердца;
- 4) протестировать диагностическую системы ИНА

Объект исследования: звуковые файла с записью сердцебиения.

Предмет исследования: классификация результатов на основе анализа звуковых файлов.

Методы исследования: анализ, моделирование, программирование.

Основная часть

1. Методы классификации звуковых файлов

Первый этап исследования включает в себя подготовку аудиоданных для дальнейшего их анализа.

Как и во всех неструктурированных форматах данных, аудиоданные имеют несколько этапов предварительной обработки, которые необходимо выполнить, прежде чем они будут представлены для анализа. Для решения задачи классификации сердечбиения были выделены два основных подхода к обработке звука:

- 1) частотно-спектральные преобразования. В основе этого подхода лежит представление данных в частотную область для извлечения набора мелчастотно-кепстральных коэффициентов.
- 2) амплитудные преобразования. В основе этого подхода лежит извлечение амплитуд аудиофайла с последующим применением сглаживающего фильтра для выделения необходимого набора данных.

Для решения задачи классификации существуют множество методов, из которых были выбраны четыре наиболее подходящие для решения данной задачи: такие как kNN (k – ближайших соседей), SVM (метод опорных векторов), CNN (сверточные нейронные сети), BLSTM (двунаправленная LSTM сеть).

Метод kNN – метрический классификатор, основанный на оценивании сходства объектов. Классифицируемый объект относится к тому классу, которому принадлежат ближайшие к нему объекты обучающей выборки.

Метод SVM – метод опорных векторов, основная идея которого перевод исходных векторов в пространство более высокой размерности и поиск разделяющей гиперплоскости с максимальным зазором в этом пространстве.

Метод CNN основан на использовании операции поэтапной свертки.

Метод BLSTM основан на использовании как входных данных, так и данных из предыдущих шагов.

Были выделены три основных класса для нейронной сети:

- 1) Artifact – не удалось записать сердечбиение;
- 2) Murmur – обнаружены нарушения в работе сердца;
- 3) Normal – не выявлено проблем с работой сердца.

2. Программирование нейронной сети

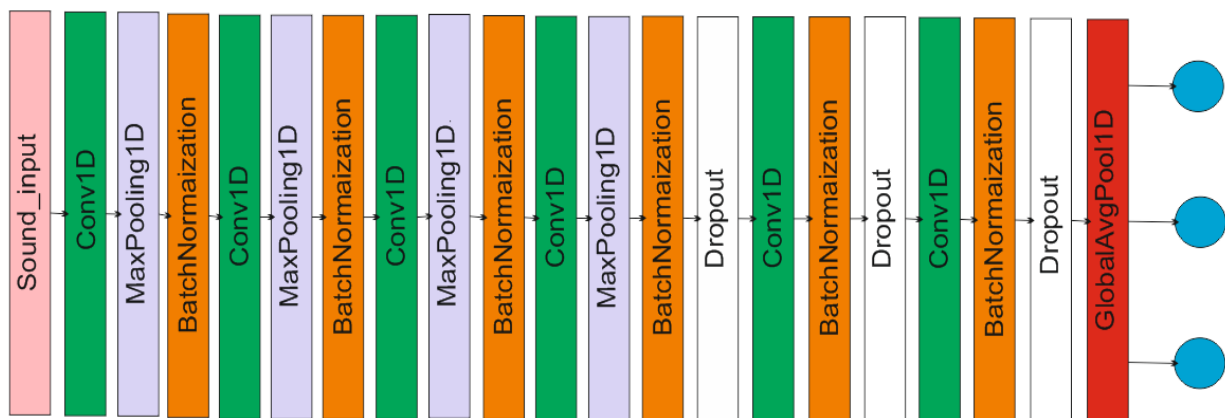
Вторым этапом стало реализация методов и исследование их эффективности при классификации звуковых файлов. Для частотного и амплитудного подходов были реализованы четыре метода классификации, т.е. написаны четыре модели нейронной сети и два простейших классификатора. Для написания нейронных сетей был выбран язык программирования Python, так как он обладает высокой производительностью при обработке данных. Ещё одним важным достоинством этого языка является большое количество библиотек для работы с машинным обучением. В основе работы нейронных сетей лежит принцип аускультации сердца. Нейронные сети были обучены и протестированы на тестовой выборке, на основе которой были получены следующие результаты точности для каждого из методов.

Как видно из таблицы (Таблица 1) наибольшую точность показала модель сверточной нейронной сети с амплитудным преобразованием.

	kNN	SVM	CNN	BLSTM
Частотно-спектральные преобразования	0.9091(k=3)	0.7792	0.8312	0.9134
Амплитудные преобразование	0.625 (k=4)	0.4583	0.9583	0.7222

Таблица 1

Модель сверточной нейронной сети с амплитудным преобразованием состоит из 4 слоев свертки с последующим объединением и нормализацией, 2 слоев с нормализацией, на 4, 5, 6 слоях применяется отсев входных данных. Затем применяется операция глобального среднего объединения временных данных. На выходе имеем Dense слой с тремя нейронами, отвечающими за каждый класс.



Прежде чем приступить к реализации нейронной сети необходимо подготовить данные для её обучения. Реализация загрузки данных происходит с помощью функции `load_file_data ()`, которая импортирует аудиофайл и

возвращает набор *mfccs*, в последствии с которым работает нейронная сеть, после преобразования в список

```
def load_file_data (folder, file_names, duration=12, sr=16000):
    input_length=sr*duration
    data = []
    for file_name in file_names:
        try:
            sound_file=folder+file_name
            print ("Загрузка файла: ",sound_file)
            X, sr = librosa.load ( sound_file, sr=sr, duration=duration,res_type='kaiser_fast')
            dur = librosa.get_duration (y=X, sr=sr)
            if (round (dur) < duration):
                print ("Фиксированная длина аудиофайла: ", file_name)
                y = librosa.util.fix_length(X, input_length)
                mfccs = np.mean (librosa.feature.mfcc (y=X, sr=sr, n_mfcc=40).T, axis=0)
        except Exception as e:
            print("Произошла ошибка при разборе файла: ", file)
            feature = np.array (mfccs).reshape([-1,1])
            data.append (feature)
    return data
```

3. Обучение нейронной сети

Используя базы данных были созданы тестовая и тренировочная выборки. На основе тренировочной выборки было реализовано обучение нейронной сети.

Обучение нейронной сети – это процесс, в котором параметры нейронной сети настраиваются посредством моделирования среды, в которую эта сеть встроена. Тип обучения определяется способом подстройки параметров. Различают алгоритмы обучения с учителем и без учителя. В работе использовался алгоритм обучения с учителем. Процесс обучения с учителем представляет собой предъявление сети выборки обучающих примеров. Каждый образец подается на входы сети, затем проходит обработку внутри структуры НС, вычисляется выходной сигнал сети, который сравнивается с соответствующим значением целевого вектора, представляющего собой требуемый выход сети.

Для того, чтобы нейронная сеть была способна выполнить поставленную задачу, ее необходимо обучить (Рис. 1).

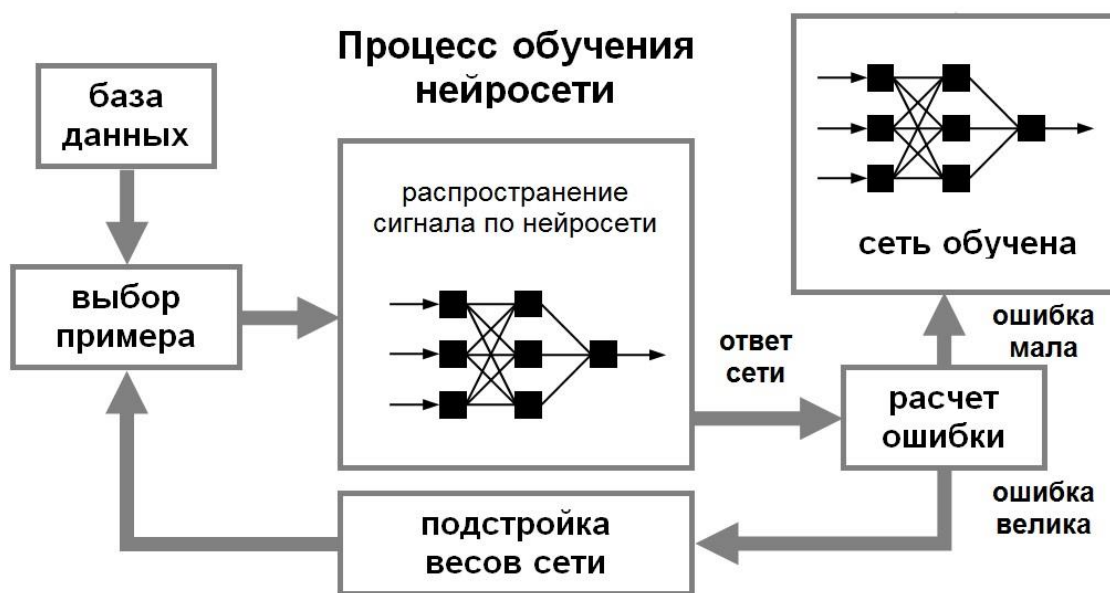


Рис. 1

Затем по определенному правилу вычисляется ошибка, и происходит изменение весовых коэффициентов связей внутри сети в зависимости от выбранного алгоритма. Векторы обучающего множества предъявляются последовательно, вычисляются ошибки и веса подстраиваются для каждого вектора до тех пор, пока ошибка по всему обучающему массиву не достигнет приемлемо низкого уровня.

Математически процесс обучения можно описать следующим образом. В процессе функционирования нейронная сеть формирует выходной сигнал Y , реализуя некоторую функцию $Y = G(X)$. Если архитектура сети задана, то вид функции G определяется значениями синаптических весов и смещенной сети.

Пусть решением некоторой задачи является функция $Y = F(X)$, заданная параметрами входных-выходных данных $(X^1, Y^1), (X^2, Y^2), \dots, (X^N, Y^N)$, для которых $Y^k = F(X^k)$ ($k = 1, 2, \dots, N$).

Обучение состоит в поиске (синтезе) функции G , близкой к F в смысле некоторой функции ошибки E .

Если выбрано множество обучающих примеров – пар (X^N, Y^N) (где $k = 1, 2, \dots, N$) и способ вычисления функции ошибки E , то обучение нейронной сети превращается в задачу многомерной оптимизации, имеющую очень большую размерность, при этом, поскольку функция E может иметь произвольный вид обучение в общем случае – многоэкстремальная невыпуклая задача оптимизации.

Вывод процесса обучения нейронной сети на экран имеет следующий вид:

Epoch 00001: loss improved from inf to 1.00303, saving model to ./best_model_trained.hdf5

Epoch 00002: loss improved from 1.00303 to 0.81297, saving model to ./best_model_trained.hdf5

Epoch 00003: loss improved from 0.81297 to 0.72901, saving model to ./best_model_trained.hdf5

Epoch 00004: loss improved from 0.72901 to 0.71217, saving model to ./best_model_trained.hdf5

Epoch 00005: loss improved from 0.71217 to 0.68847, saving model to ./best_model_trained.hdf5

...

Процесс обучения данной модели можно увидеть на графике (Рис. 2).

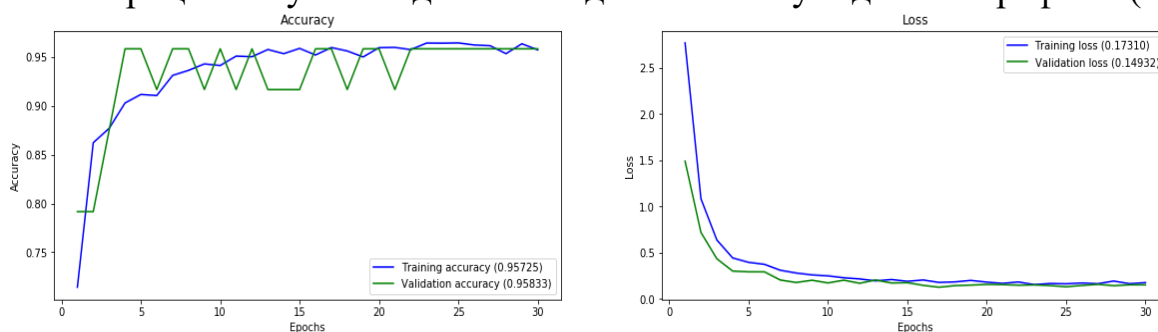


Рис. 2

После обучения было выполнено сохранение модели с весами, полученными в процессе обучения для дальнейшего использования.

4. Тестирование нейронной сети

Третий этап разработки включал в себя проведение тестирования, используя тестовую выборку.

```
y_pred = model.predict_classes(x_test, batch_size = 32)
```

```
scores = model.evaluate(x_test, y_test, verbose = 0)
```

```
print("Точность обученной модели: ", round(scores[1]*100,"%"))
```

Для каждой модели нейронной сети были получены определенных значения точности.

5. Разработка и сборка оборудования для проведения записи и диагностики звуков сердцебиения

Для эффективной записи сердечного ритма была разработана и собрана следующая конструкция микрофона (Рис. 11):

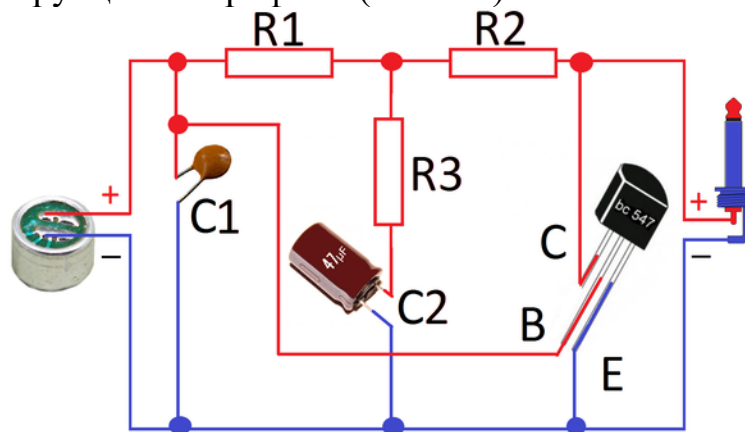


Рис. 11

Для сборки усилителя для микрофон использовались следующие элементы:

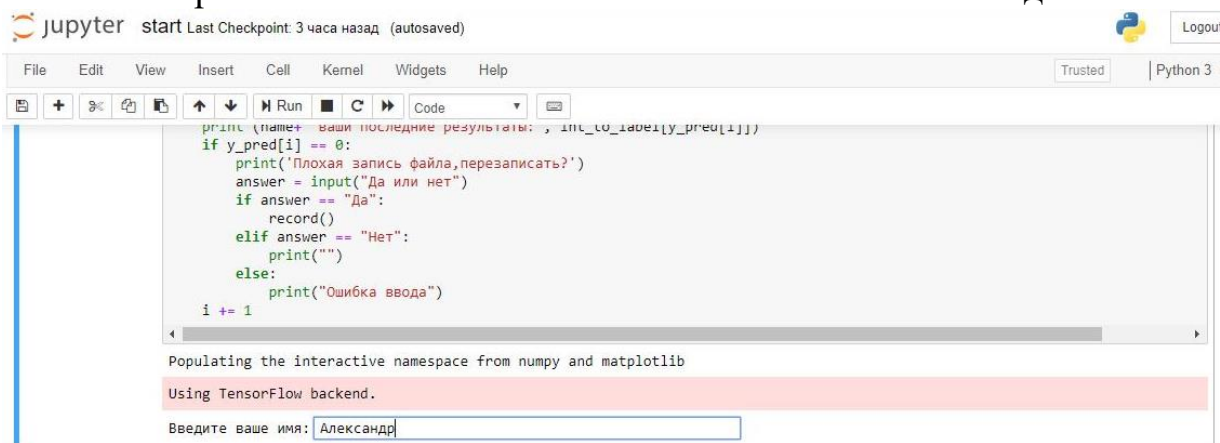
- 1) микрофон (1 шт.);
- 2) штекер 3,5 миллиметра (1 шт.);
- 3) транзистор BC547 (1 шт.);
- 4) резисторы номиналом 1кОм (R1 и R2) (2 шт.);
- 5) резистор с возможностью регулировать сопротивление для точной настройки (R3) (1 шт.);
- 6) дисковый керамический конденсатор номиналом 300 пФ (C1) (1 шт.);
- 7) электролитический конденсатор 47 мкФ (C2) (1 шт.).

Для записи низкочастотных звуков, какими являются ритмы сердца, использовался электретный микрофон.

Для усиления громкости и точности звукозаписи использовали рабочую головку стетоскопа.

6. Тестирование собранной системы

Тестирование системы пользователем начинается с ввода имени.



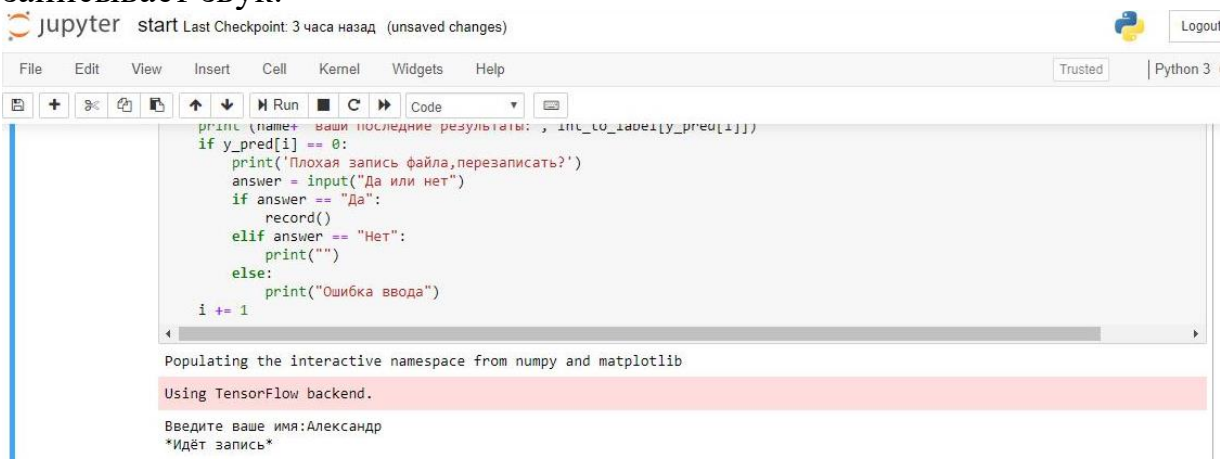
```
print (name+ " ваши последние результаты: ", int_to_label[y_pred[i]])
if y_pred[i] == 0:
    print('Плохая запись файла,перезаписать?')
    answer = input("Да или нет")
    if answer == "Да":
        record()
    elif answer == "Нет":
        print("")
    else:
        print("Ошибка ввода")
i += 1
```

Populating the interactive namespace from numpy and matplotlib

Using TensorFlow backend.

Введите ваше имя: Александр

В область сердца на груди крепится микрофон, после чего программа записывает звук.



```
print (name+ " ваши последние результаты: ", int_to_label[y_pred[i]])
if y_pred[i] == 0:
    print('Плохая запись файла,перезаписать?')
    answer = input("Да или нет")
    if answer == "Да":
        record()
    elif answer == "Нет":
        print("")
    else:
        print("Ошибка ввода")
i += 1
```

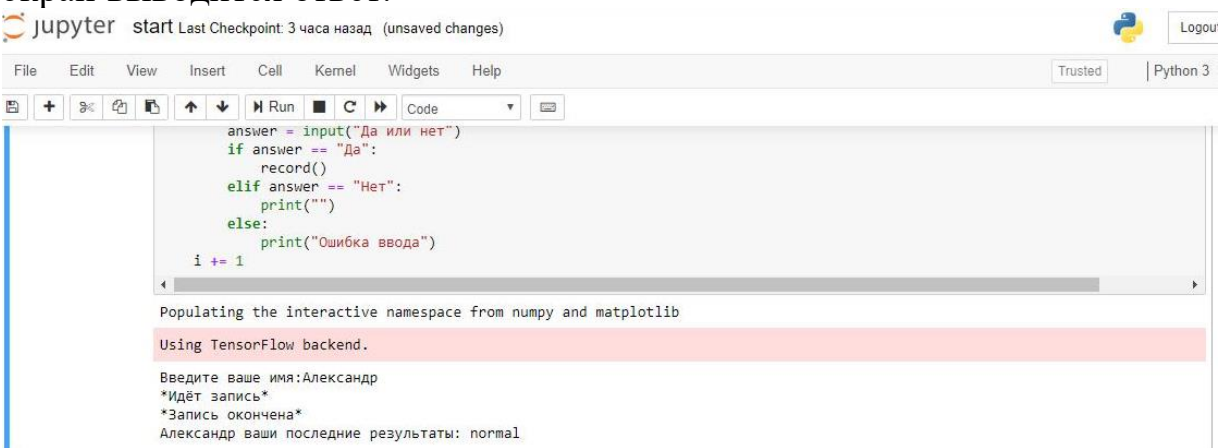
Populating the interactive namespace from numpy and matplotlib

Using TensorFlow backend.

Введите ваше имя:Александр

Идёт запись

После окончания записи звуковой файл обрабатывается программой и на экран выводится ответ.



```
answer = input("Да или нет")
if answer == "Да":
    record()
elif answer == "Нет":
    print("")
else:
    print("Ошибка ввода")
i += 1
```

Populating the interactive namespace from numpy and matplotlib

Using TensorFlow backend.

Введите ваше имя:Александр

Идёт запись

Запись окончена

Александр ваши последние результаты: normal

20.11.2019 года была проведена консультация с врачом общей практики УЗ «Высоковская городская больница» Аврамчук А. И. Был изложен принцип работы программы и получено разрешение на проведение дополнительного исследования пациентов, направленных на ЭКГ.

Тестирования программы ИНА проводилось в учреждении здравоохранения «Высоковская городская больница» в период с 25.11.2019 по

27.11.2019. Пациентам, делающим ЭКГ, было предложено сделать запись сердцебиения системой ИНА. Результаты сравнения анализа записей программой и результаты ЭКГ представлены в таблице.

Пациент	ИНА	ЭКГ
Пациент 1	normal	патологии не выявлено
Пациент 2	normal	патологии не выявлено
Пациент 3	normal	патологии не выявлено
Пациент 4	normal	патологии не выявлено
Пациент 5	murmur	патологии не выявлено
Пациент 6	normal	патологии не выявлено
Пациент 7	normal	отклонения в работе сердца
Пациент 8	normal	патологии не выявлено
Пациент 9	artifact	отклонения в работе сердца
Пациент 10	normal	патологии не выявлено
Пациент 11	normal	патологии не выявлено
Пациент 12	normal	патологии не выявлено
Пациент 13	normal	патологии не выявлено
Пациент 14	murmur	отклонения в работе сердца
Пациент 15	artifact	патологии не выявлено
Пациент 16	normal	патологии не выявлено
Пациент 17	murmur	патологии не выявлено
Пациент 18	normal	отклонения в работе сердца
Пациент 19	artifact	патологии не выявлено
Пациент 20	normal	патологии не выявлено

Результаты диагностики программой ИНА и результаты ЭКГ совпали на 65 %, при заявленной точности нейронной сети 81% (на 27.11.2019 г.).

Результаты обследований ЭКГ и ИНА были обсуждены на повторной консультации с врачом Аврамчук А. И. Врач прокомментировала несовпадения результатов исследования. В частности, было отмечено, что результат, определенный как artifact (т.е. не удалось записать сердцебиение) имеет место быть, т.к. определить проблему даже врачу, особенно у пациентов, склонных к полноте, очень сложно, диагностировать заболевание можно только с помощью УЗИ сердца. Результат murmur (т.е. обнаружены нарушения в работе сердца) при результате «патологии не выявлено» также требует дополнительного исследования и подтверждения результатов на УЗИ.

Заключение

Таким образом, было проведено исследование разных подходов и методов анализа звуков сердцебиения. Наилучшие результаты показал амплитудный подход при использовании сверточных нейронных сетей. Используя данную модель была разработана система «IHeart Assist – диагностическая система анализа звуков сердцебиения» и проведено ее тестирование.

Данную систему можно использовать для первичной диагностики отклонений в работе сердца.

Список использованных источников

1. 7 архитектур нейронных сетей для решения задач NLP [Электронный ресурс]. – Режим доступа: <https://neurohive.io/ru/osnovy-data-science/7-arhitektur-nejronnyh-setej-nlp/>. – Дата доступа: 20.10.2019.
2. Классификация звуков с помощью TensorFlow [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/company/dataart/blog/343464/>. – Дата доступа: 20.10.2018.
3. Обучение нейросети с учителем, без учителя, с подкреплением — в чем отличие? Какой алгоритм лучше? [Электронный ресурс]. – Режим доступа: <https://neurohive.io/ru/osnovy-data-science/obuchenie-s-uchitelem-bez-uchitelja-s-podkrepleniem/>. – Дата доступа: 15.10.2019.
4. Распознавание [Электронный ресурс]. – Режим доступа: <http://www.machinelearning.ru/wiki/index.php>. – Дата доступа: 10.10.2019.
5. Classifying Heart Sounds Challenge [Электронный ресурс]. – Режим доступа: <http://www.peterjbentley.com/heartchallenge/>. – Дата доступа: 20.10.2019.
6. LibROSA [Электронный ресурс]. – Режим доступа: <https://librosa.github.io/librosa/>. – Дата доступа: 14.10.2019.
7. LSTM — нейронная сеть с долгой краткосрочной памятью [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/company/wunderfund/blog/331310/>. – Дата доступа: 12.10.2019.
8. Matplotlib [Электронный ресурс]. – Режим доступа: <https://matplotlib.org/#>. – Дата доступа: 15.10.2019.
9. NumPy [Электронный ресурс]. – Режим доступа: <https://numpy.org/#>. – Дата доступа: 12.10.2019.
10. Python Data Analysis Library [Электронный ресурс]. – Режим доступа: <https://pandas.pydata.org/>. – Дата доступа: 12.10.2019.
11. Softmax [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Softmax>. – Дата доступа: 13.10.2019.