МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Искусственные нейронные сети»

Тема: Бинарная классификация сигналов радара

Студент гр. 8383 _____ Дейнега В. Е.

Преподаватель _____ Жангиров Т. Р.

Санкт-Петербург

2021

Цель работы

Реализовать классификацию между камнями (R) и металлическими цилиндрами (M) на основе данных об отражении сигналов радара от поверхностей.60 входных значений показывают силу отражаемого сигнала под определенным углом. Входные данные нормализованы и находятся в промежутке от 0 до 1.

Задание.

- Ознакомиться с задачей бинарной классификации
- Загрузить данные
- Создать модель ИНС в tf.Keras
- Настроить параметры обучения
- Обучить и оценить модель
- Изменить модель и провести сравнение. Объяснить результаты

Выполнение работы

Для выполнения лабораторной работы были импортированы все необходимые библиотеки, скачан и открыт файл с данными для анализа sonar.all-data(csv).

```
import pandas
import numpy
from tensorflow.keras.layers import Dense
from tensorflow.keras.models import Sequential
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt

dataframe = pandas.read_csv("sonar.csv", header=None)
dataset = dataframe.values
```

Была создана модель с 2 слоями, определена функция потерь, оптимизатор и метрика:

```
model = Sequential()
model.add(Dense(60,activation='relu',input_dim=60))
model.add(Dense(1, activation='sigmoid'))
```

```
    model.compile(optimizer='adam',loss='binary_crossentropy',
metrics=['accuracy'])
    hist = model.fit(X, encoded_Y, epochs=epochs, batch_size=10,
validation_split=0.1)
```

Для наглядности были построены графики ошибок и точности в ходе обучения:

```
    hist_dict = hist.history
    loss_values = hist_dict['loss']
    val_loss_values = hist_dict['val_loss']
    epochs = range(1, len(loss_values) + 1)
    plt.plot(epochs, loss_values, 'bo', label='Training loss')
    plt.plot(epochs,  val_loss_values,  'b',  label='Validation
loss')
    plt.title('Training and validation loss')
    plt.xlabel('Epochs')
    plt.ylabel('Loss')
    plt.legend()
    plt.show()


    plt.clf()
    acc_values = hist_dict['accuracy']
    val_acc_values = hist_dict['val_accuracy']
    plt.plot(epochs, acc_values, 'bo', label='Training acc')
    plt.plot(epochs, val_acc_values, 'b', label='Validation acc')
    plt.title('Training and validation accuracy')
    plt.xlabel('Epochs')
    plt.ylabel('Accuracy')
    plt.legend()
    plt.show()
```

Результаты тестирования ИНС1 (60 нейронов на входном слое, 1 на выходном слое) представлены на рис. 1 и 2.

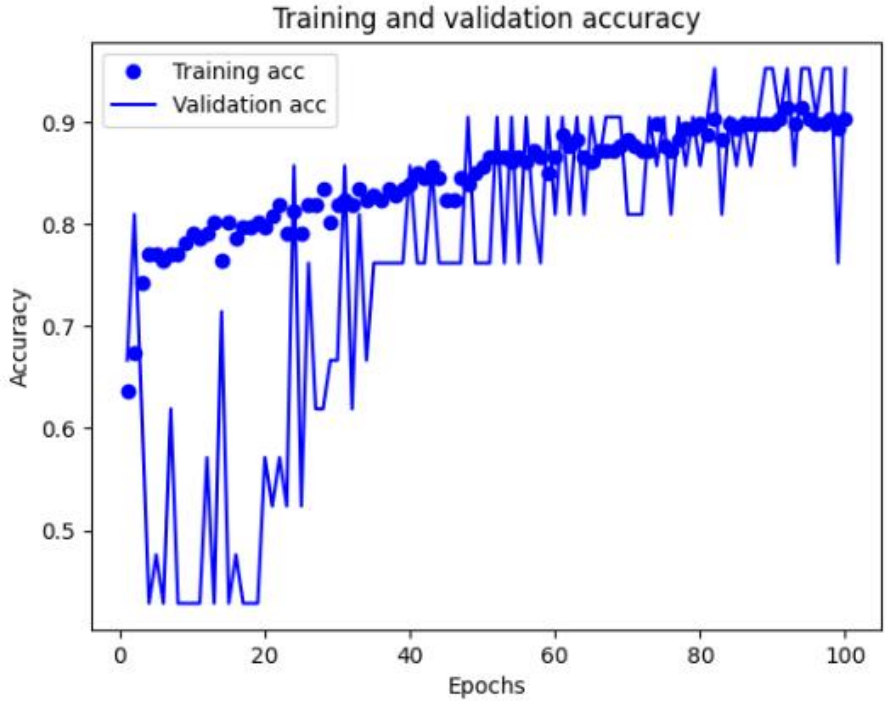

Рисунок 1 – График точности ИНС1.

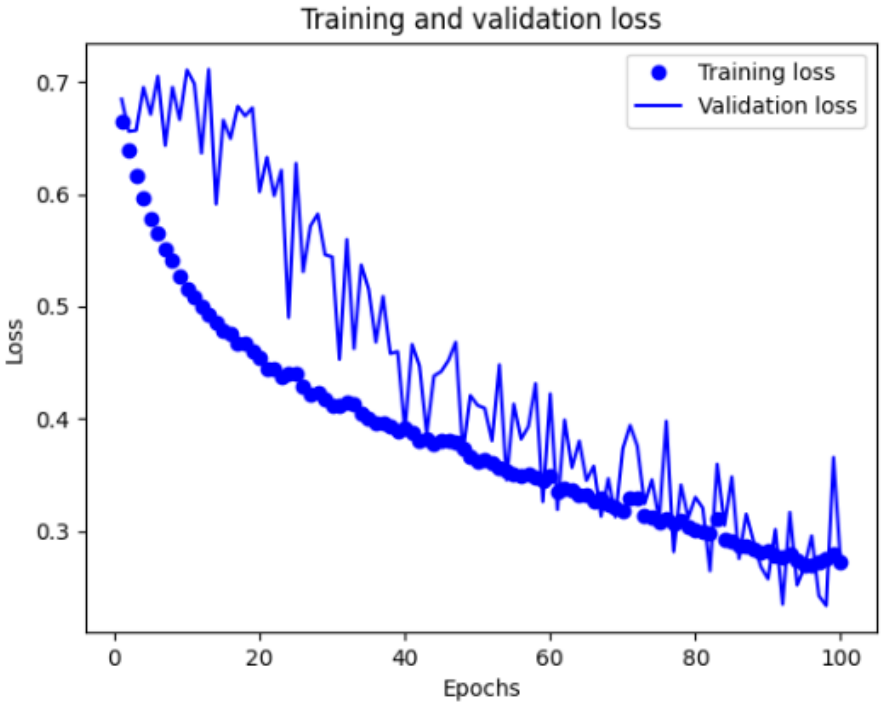

Рисунок 2 – График ошибок ИНС1.

Из графиков видно, что точность на данных для обучения достигает ~0.9, однако точность на тестовых данныхсильно колеблется и на 99 эпохе достигает значения ~0.75, что говорит о не самой оптимальной конфигурации нейросети.

Ошибки на обучающих данных достигают значения в 0.3, на тестовых так же наблюдаются сильные колебания.

В представленном наборе данных присутствует некоторая избыточность, т.к. с разных углов описывается один и тот же сигнал. Вероятно, что некоторые углы отражения сигнала имеют большую значимость, чем другие. Изменение количества нейронов во входном слое напрямую влияет на количество признаков, с которыми будет работать нейронная сеть.

Изменим структуру сети, уменьшим количество нейронов на входном слое до 30. Результаты тестирования ИНС2 представлены на рис. 3 и 4.

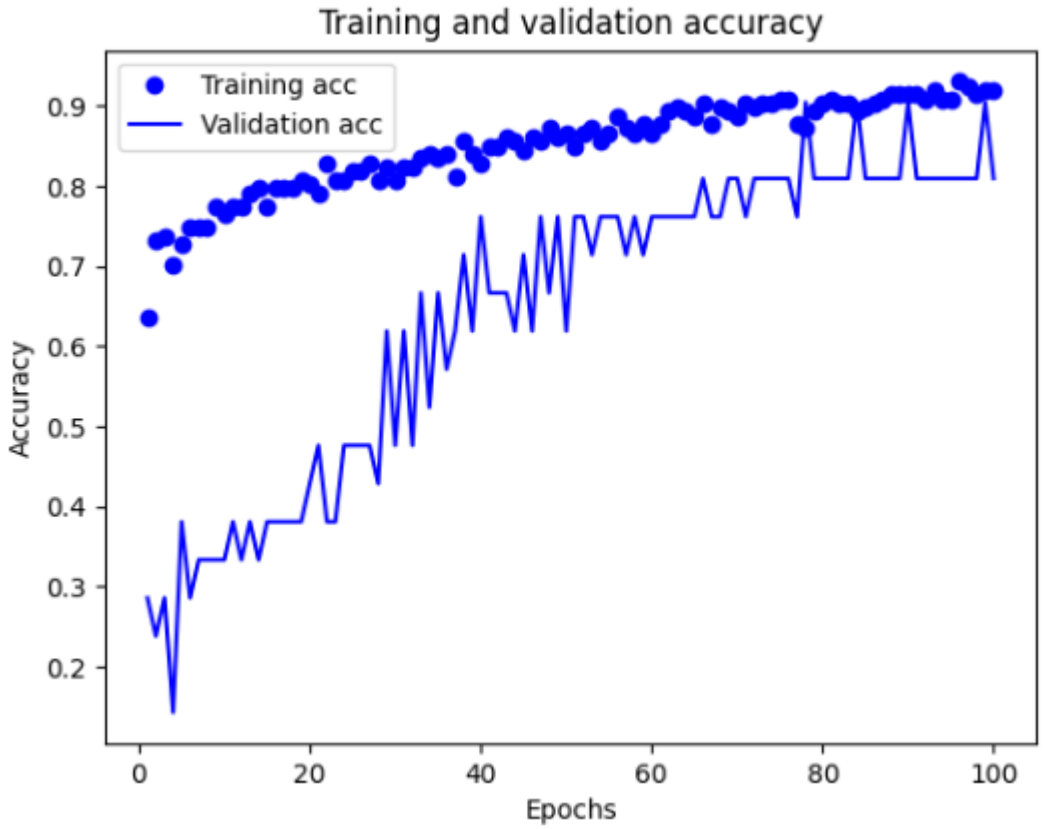

Рисунок 3 – График точности ИНС1.

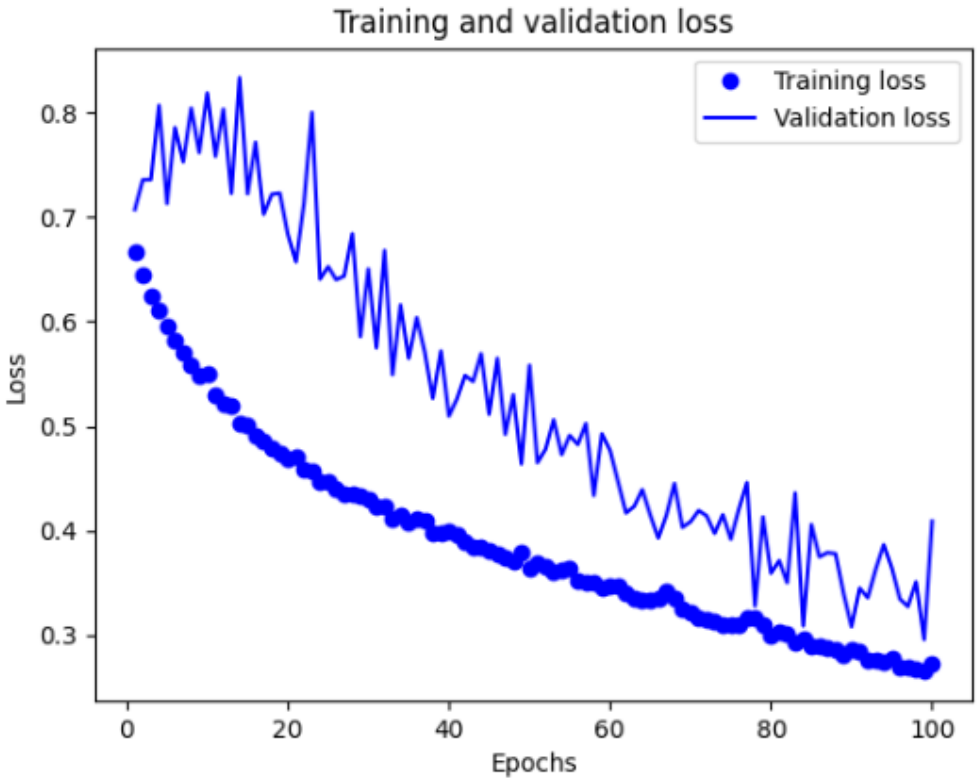

Рисунок 4 – График ошибок ИНС1.

Из графика точности видно, что точности на обучающих данных не изменилась, однако точность на тестовых данных значительно упала. Ошибки на обучающих данных остались прежними, на тестовых ошибка увеличилась. Из этого можно сделать вывод, что такая модель плохо справляется с поставленной задачей.

Нейронная сеть с несколькими слоями позволяет находить закономерности не только во входных данных, но и в их комбинации. Также, дополнительные слои позволяют ввести нелинейность в сеть, что позволяет получать более высокую точность.

Добавим скрытый слой на 15 нейронов, результаты работы ИНС3 приведены на рис. 5 и 6.

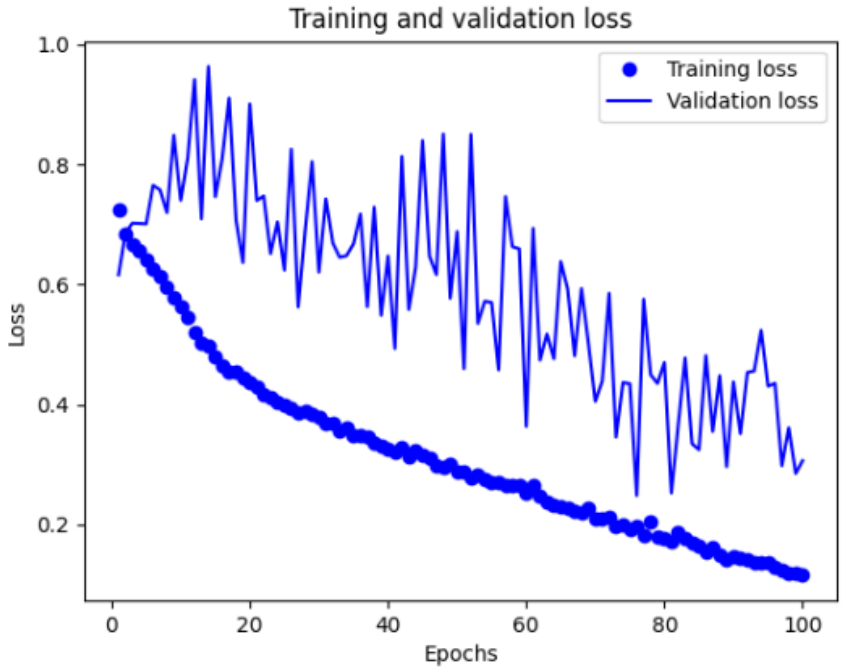

Рисунок 5 – График ошибок ИНС3.

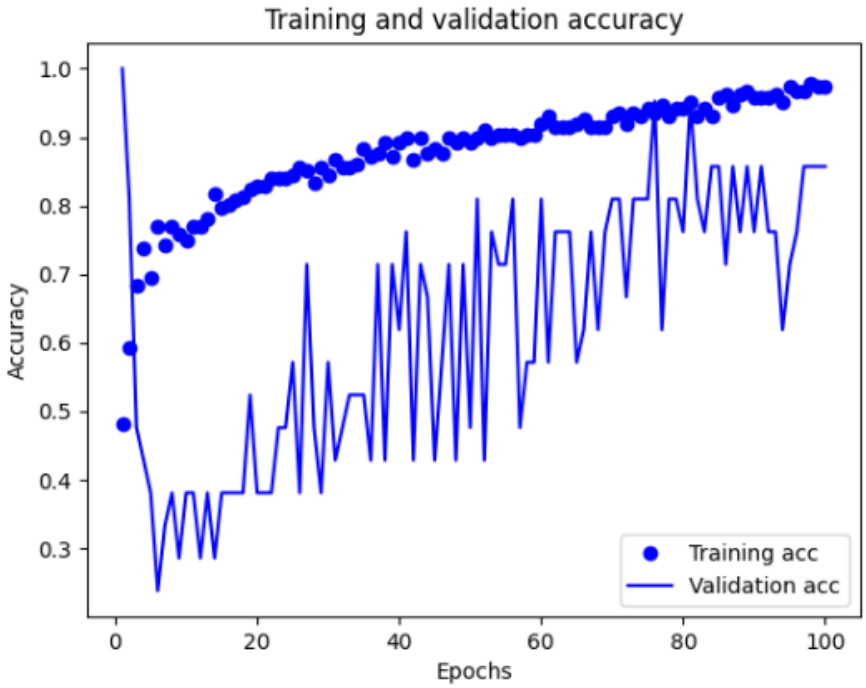

Рисунок 6 – График точности ИНС3.

Как видно на графиках, на обучающих данных точность возросла, а ошибки уменьшились, однако на тестовых данных точность значительно уменьшилась, а ошибки увеличились.

Изменим структуру сети. Добавим на входной слой еще 15 нейронов, а также добавим на скрытый слой еще 45 нейронов. Результаты тестирования ИНС4 представлены на рис. 7 и 8.

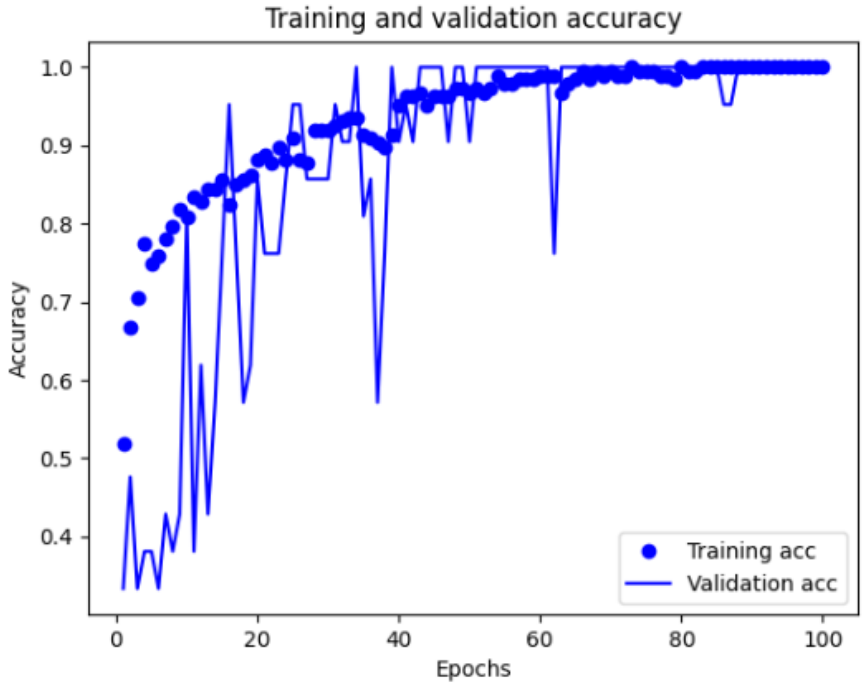

Рисунок 7 – График точности ИНС4.

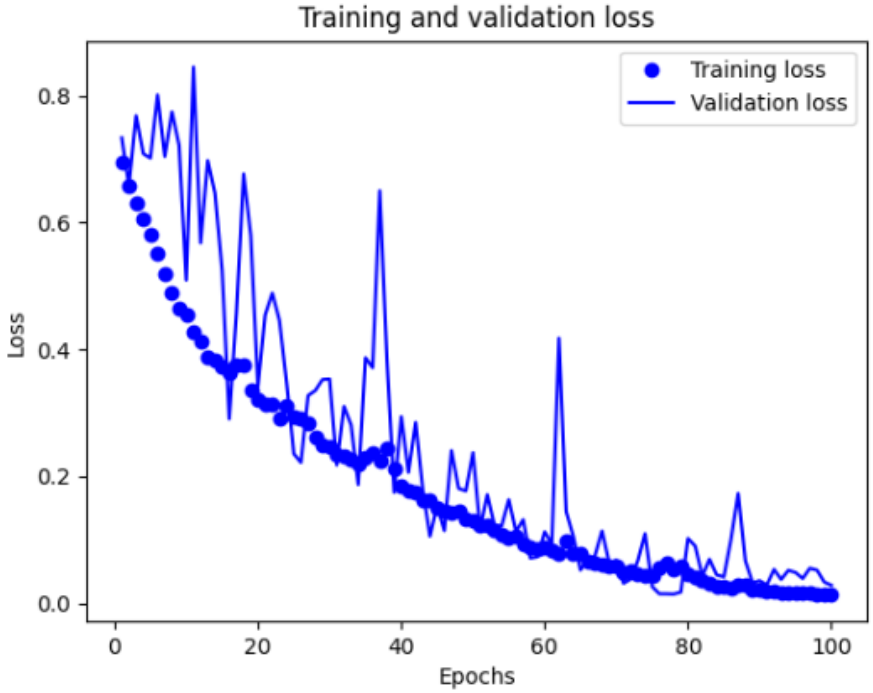

Рисунок 8 – График ошибок ИНС4.

На графике точности можно наблюдать значительные скачки точности на тестовых данных до 65-ой эпохи, далее графики сходятся в единице, что говорит о выской точности. На графике ошибок наблюдается резкий скачок на ~86 эпохе, однако в целом ИНС4 показывает результаты лучше, чем ее предшественники.

Выводы.

В ходе лабораторной работы была реализована классификация между камнями (R) и металлическими цилиндрами (M) на основе данных об отражении сигналов радара от поверхностей. Были рассмотрены различные модели ИНС.

Приложение А

```
import pandas
from tensorflow.keras.layers import Dense
from tensorflow.keras.models import Sequential
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt

dataframe = pandas.read_csv("sonar.csv", header=None)
dataset = dataframe.values
X = dataset[:, 0:60].astype(float)
Y = dataset[:, 60]

encoder = LabelEncoder()
encoder.fit(Y)
encoded_Y = encoder.transform(Y)

model = Sequential()
model.add(Dense(45, activation='relu', input_dim=60))
model.add(Dense(60, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',    loss='binary_crossentropy',
metrics=['accuracy'])
hist  =  model.fit(X,  encoded_Y,  epochs=100,  batch_size=10,
validation_split=0.1)

hist_dict = hist.history
loss_values = hist_dict['loss']
val_loss_values = hist_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs,   val_loss_values,   'b',   label='Validation
loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()


plt.clf()
acc_values = hist_dict['accuracy']
val_acc_values = hist_dict['val_accuracy']
plt.plot(epochs, acc_values, 'bo', label='Training acc')
plt.plot(epochs, val_acc_values, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```