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

ОТЧЕТ

по лабораторной работе №8

по дисциплине «Искусственные нейронные сети»

Тема: «Генерация текста на основе “Алисы в стране чудес”»

Студент гр. 7381 _____ Трушников А.П.

Преподаватель _____ Жукова Н.А..

Санкт-Петербург

2020

Цель работы.

Рекуррентные нейронные сети также могут быть использованы в качестве генеративных моделей.

Это означает, что в дополнение к тому, что они используются для прогнозных моделей (создания прогнозов), они могут изучать последовательности проблемы, а затем генерировать совершенно новые вероятные последовательности для проблемной области.

Подобные генеративные модели полезны не только для изучения того, насколько хорошо модель выявила проблему, но и для того, чтобы узнать больше о самой проблемной области.

Задачи.

- Ознакомиться с генерацией текста
- Ознакомиться с системой Callback в Keras

Ход работы.

1. Была реализована модель ИНС, которая умеет генерировать текст. Архитектура ИНС представлена на рис. 1.

```
callbacks_list = [checkpoint, MyCallback([0, 2, 5, 10, 15, 20])]

model = Sequential()
model.add(LSTM(256, input_shape=(X.shape[1], X.shape[2])))
model.add(Dropout(0.2))
model.add(Dense(y.shape[1], activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam')
model.fit(X, y, epochs=20, batch_size=128, callbacks=callbacks_list)
```

Рисунок 1 – Архитектура построенной сети

2. Был написан собственный CallBack – MyCallback, который показывает то, как генерируется текст во время обучения. Код представлен на рис. 2-3.

```python
class MyCallback(tensorflow.keras.callbacks.Callback):
    def __init__(self, epochs):
        super(MyCallback, self).__init__()
        self.epochs = epochs

    def on_epoch_end(self, epoch, logs=None):
        if epoch in self.epochs:
            print_text(self.model, epoch)
```

Рисунок 2 – код callback-а

```python
def print_text(model, epoch=0):
    start = numpy.random.randint(0, len(dataX) - 1)
    pattern = dataX[start]
    print("Seed:")

    print("\"", ''.join([int_to_char[value] for value in pattern]), "\"")
    text = []

    for i in range(1000):
        x = numpy.reshape(pattern, (1, len(pattern), 1))
        x = x / float(n_vocab)
        prediction = model.predict(x, verbose=0)
        index = numpy.argmax(prediction)
        result = int_to_char[index]
        text.append(result)
        pattern.append(index)
        pattern = pattern[1:len(pattern)]
    with open('text_{}.txt'.format(epoch), 'w') as file:
        file.write(''.join(text))
```

Рисунок 3 –вспомогательная функция callback-а

Приведем часть сгенерированных текстов в ходе обучения модели.

Текст, сгенерированный на 2 эпохе:

the toet the toet

Как видно из приведенного текста, наблюдается сплошные повторения из двух слов.

Текст, сгенерированный на 5 эпохе:

```
woen the woine to the toiee to the toeee to the toeee  the wouue the woiee to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the toeee  the woine tas to the toeee to the to
```

Как видно из приведенного текста, наблюдается сплошные повторения, но слов уже больше.

Текст, сгенерированный на 10 эпохе:

```
ano aadk to the tooee an thee  the was oo tie tooee oo the sooee of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the sooe of the soo
```

Как видно из приведенного текста, он все также плох. Вначале наблюдается что-то похожее на предложении, но потом идут повторения.

Текст, сгенерированный на 15 эпохе:

n a coeat ana mooee an inr to cens an inr to ce to head the rime of the rabbit, and the was no toe tiee and the had iov to be iuree to the thitg rabbit would be ioee to her an in and toe tal oo the toiee an inr to ce to head the rieee of the rabbit, and the was no toe tiee and the had iov to be iuree to the thitg rabbit would be ioee to her an in and toe tal oo the toiee an inr to ce to head the rieee of the rabbit, and the was no toe tiee and the had iov to be iuree to the thitg rabbit would be ioee to her an in and toe tal oo the toiee an inr to ce to head the rieee of the rabbit, and the was no toe tiee and the had iov to be iuree to the thitg rabbit would be ioee to her an in and toe tal oo the toiee an inr to ce to head the rieee of the rabbit, and the was no toe tiee and the had iov to be iuree to the thitg

Как видно из приведенного текста, повторений стало гораздо меньше. В тексте можно рассмотреть нормальные словосочетания.

Текст, сгенерированный на 19 эпохе:

edred in the sood of the sabbit sfde an the could her head to the thnt of the courd, and whi hert wer the wiite rabbit was aoi the wiile rabbit was a little oate and the whoeg harden sere the whot ofdel of the sabli bld the part wiin iar haad she cid no the sam so the sinee anoners the was so thyi thit way the white rabbit, and whs gel to the whrt on aelit in the wiile 'and whs ger tu mete to the thit oo the shrere '

'the dorss thet toe 'asd tou dinn the sore,' she said to herself, 'io would ba anl oo the simtee of the gitte th then io the seme to the whit oo here oo thet 'a

```
iott  she wish the seme to tou the whit to tote toued oo the sooe-'


'io she soee tureed to toue   shi mrck turtle sapd to hhrself, "that s the
shre then ' she seit on aelcr tonkeig. "
```

Начали появляться разборчивые предложения.

Выводы.

В ходе выполнения работы была построена и обучена сеть, генерирующая текст на основе «Алиса в стране чудес». Также написан callback, с помощью которого можно отследить процесс обучения сети.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД

```
import tensorflow.keras.callbacks
import numpy
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import LSTM
from keras.callbacks import ModelCheckpoint
from keras.utils import np_utils


filename = "wonderland.txt"
raw_text = open(filename).read()
raw_text = raw_text.lower()

chars = sorted(list(set(raw_text)))
char_to_int = dict((c, i) for i, c in enumerate(chars))
int_to_char = dict((i, c) for i, c in enumerate(chars))

n_chars = len(raw_text)
n_vocab = len(chars)

print("Total Characters: ", n_chars)
print("Total Vocab: ", n_vocab)

seq_length = 100
dataX = []
dataY = []

for i in range(0, n_chars - seq_length, 1):
    seq_in = raw_text[i:i + seq_length]
    seq_out = raw_text[i + seq_length]
    dataX.append([char_to_int[char] for char in seq_in])
    dataY.append(char_to_int[seq_out])


def print_text(model, epoch=0):
    start = numpy.random.randint(0, len(dataX) - 1)
    pattern = dataX[start]
    print("Seed:")

    print("\"", ''.join([int_to_char[value] for value in pattern]), "\"")
    text = []

    for i in range(1000):
        x = numpy.reshape(pattern, (1, len(pattern), 1))
        x = x / float(n_vocab)
        prediction = model.predict(x, verbose=0)
        index = numpy.argmax(prediction)
        result = int_to_char[index]
        text.append(result)
        pattern.append(index)
        pattern = pattern[1:len(pattern)]
```

```
    with open('text_{}.txt'.format(epoch), 'w') as file:
        file.write(''.join(text))


class MyCallback(tensorflow.keras.callbacks.Callback):
    def __init__(self, epochs):
        super(MyCallback, self).__init__()
        self.epochs = epochs

    def on_epoch_end(self, epoch, logs=None):
        if epoch in self.epochs:
            print_text(self.model, epoch)


n_patterns = len(dataX)
print("Total Patterns: ", n_patterns)

# reshape X to be [samples, time steps, features]
X = numpy.reshape(dataX, (n_patterns, seq_length, 1))
# normalize
X = X / float(n_vocab)
# one hot encode the output variable
y = np_utils.to_categorical(dataY)

filepath="weights-improvement-{epoch:02d}-{loss:.4f}.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='loss', verbose=1,
save_best_only=True, mode='min')
callbacks_list = [checkpoint, MyCallback([0, 2, 5, 10, 15, 19])]

model = Sequential()
model.add(LSTM(256, input_shape=(X.shape[1], X.shape[2])))
model.add(Dropout(0.2))
model.add(Dense(y.shape[1], activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam')
model.fit(X, y, epochs=20, batch_size=128, callbacks=callbacks_list)
```