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”»

Студентка гр. 7381 _____ Алясова А.Н.

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

Требования.

- Реализовать модель ИНС, которая будет генерировать текст
- Написать собственный CallBack, который будет показывать то как генерируется текст во время обучения (то есть раз в какое-то количество эпох генерировать и выводить текст у необученной модели)
- Отследить процесс обучения при помощи TensorFlowCallBack, в отчете привести результаты и их анализ

Ход работы.

В ходе работы была создана и обучена модель нейронной сети, весь код представлен в приложении.

В архитектуре сети определен один скрытый слой LSTM с 256 единицами памяти. Сеть использует выпадение с вероятностью 20. Выходной уровень – это плотный уровень, использующий функцию активации softmax для вывода прогнозирования вероятности для каждого из 47 символов в диапазоне от 0 до 1.

Для контроля обучения был написан callback, выводящий в консоль сгенерированный сетью текст после выбранных эпох.

Результат генерации текста в ходе обучения сети представлен в табл. 1.

Таблица 1 – Результат работы написанного callback

Номер эпохи	Результат
1	Seed: " who is dinah, if i might venture to ask the question?' said the lory. alice replied eagerly, for sh "e to

6	Seed: " the crumbs,' said the mock turtle: 'crumbs would all wash off in the sea. but they have their tails "to the toete so the toete and she woete tas io the woele to
10	Seed: " me,' thought alice, 'they're sure to kill it in a day or two: wouldn't it be murder to leave it beh " in ' 'io sou dn a san a lete ti teat ' said the mant, and the whit hn a dotr hf toete an aelen an anr an anr and the whit hn wouhd bo a lott of the care and the white rabbit so be anl an anr and the whit hn wouhd bo a lott of the care and the white rabbit so be anl an anr and the whit hn wouhd bo a lott of the care and the white rabbit so be anl an anr and the whit hn wouhd bo a lott of the care and the white rabbit so be anl an anr and the whit hn wouhd bo a lott of the care and the white rabbit so be anl an anr and the whit hn wouhd bo a lott of the care and the white rabbit so be anl an anr and the whit hn wouhd bo a lott of the care and the white rabbit so be anl an anr and the whit hn wouhd bo a lott of the care and the white rabbit so be anl an anr and the whit hn wouhd bo a lott of the care and the white rabbit so be anl an anr and the whit hn wouhd bo
15	Seed: " ying 'we beg your acceptance of this elegant thimble'; and, when it had finished this short speech, " and the white rabbit were to aeiin an once the mote tu tee sai so tae she mart was oo the tooe, and the white rabbit were to ani the past oa tee so tae the tas of the pooe of the courd, and the white rabbit were to ani the past oa tee so tae the tas of the pooe of the courd, and the white rabbit were to ani the past oa tee so tae the tas of the pooe of the courd, and the white rabbit were to ani

	the past oa tee so tae the tas of the pooe of the courd, and the white rabbit were to ani the past oa tee so tae the tas of the pooe of the courd, and the white rabbit were to ani the past oa tee so tae the tas of the pooe of the courd, and the white rabbit were to ani the past oa tee so tae the tas of the pooe of the courd, and the white rabbit were to ani the past oa tee so tae the tas of the pooe of the courd, and the white rabbit were to ani the past oa tee so tae the tas of he tas of the pooe of the courd, an
20	Seed: " d better take him his fan and gloves--that is, if i can find them.' as she said this, she came upon " a siry of geldsenn the had not the garter whsh all the soode ani the whst hor letter that she was not io the tine of the shoe afd no anice the white rabbit sirel alice was tor allwh to be a lenter of the goorh of the sart of the court, and the whst hare het head to fer that she was not in the tine, and saed to ferself the mucen' and the sored ' 'io wou dre't dlt oo whrh the sooss,' said the caterpillar. 'ie ionr the soeet ann the saad" said the gatter. "ne toune to tee the magter an tou dinl the mott oi the saad-' 'i sean toe car a ditd ' shiught alice. 'io so ae anledg to tay the mabte tas sa lott of then she wes aol the sioe afd no the taadit she was a little thate whsh a sille rabbe on the gorphon she was not in the tine of the thoe afdin, and she whst har aelin an inctt of the sooe. 'the dorst sat a sirslen ' she said to herself, 'io s soe cirtt tai soe toilee was a little so tal, 'it w

Как видно, в начале сеть просто генерирует повторяющуюся последовательность из 4 символов. Дальше генерирует повторяющуюся последовательность, только большей длины. Затем сначала сгенерировала одну последовательность, а потом начала повторять вторую.

Таким образом, сеть сгенерировала текст без постоянных повторений. В тексте можно разобрать некоторые слова, но он не имеет смысловой нагрузки.

Выводы.

В ходе работы были изучены задача генерации текста и система callback в keras нейронными сетями с использованием python и keras, был написан собственный callback, который в процессе обучения модели генерировал текст в конце определенной эпохи.

ПРИЛОЖЕНИЕ

Исходный код

```
import sys
import numpy
import tensorflow.keras.callbacks
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

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

    def on_epoch_end(self, epoch, logs={}):
        if epoch in self.epochs:
            generateSequence(model)


#загрузка текста ASCII в память и преобразование всех символов в
нижний регистр
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))

#суммирование набора данных
n_chars = len(raw_text)
n_vocab = len(chars)
print("Total Characters: ", n_chars)
print("Total Vocab: ", n_vocab)

#разделение книги на последовательности
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])
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)


def generateSequence(model):
# pick a random seed
    start = numpy.random.randint(0, len(dataX)-1)
    pattern = dataX[start]
    print("Seed:")
    print("\"", ''.join([int_to_char[value] for value in pattern]),
"\"")
    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]
        seq_in = [int_to_char[value] for value in pattern]
        sys.stdout.write(result)
        pattern.append(index)
        pattern = pattern[1:len(pattern)]


# define the LSTM model
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')
```

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


model.fit(X, y, epochs=20, batch_size=128, callbacks=callbacks_list)
```