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. 7383 _____ Левкович Д. В.

Преподаватель _____ Жукова Н.А.

Санкт-Петербург

2020

Цель работы:

Реализовать генерацию текста на основе текста из сказки «Алиса в стране чудес»

Задачи.

1. Ознакомиться с генерацией текста
2. Ознакомиться с системой Callback в Keras

Ход работы.

1. Была создана модель рекуррентной нейронной сети. Архитектура сети представлена на рис. 1.

```
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')
```

Рисунок 1 - Архитектура сети

2. Для отслеживания прогресса генерации текста во время обучения нейронной сети был написан класс `CustomCB`, который по окончании определенных эпох выводит в файл текст, сгенерированный нейронной сетью.
3. Посмотрим результат работы сети на 2, 6, 16 и 20 эпохах.

Вторая эпоха:

Seed:" with a t!' said the king sharply. 'do you

take me for a dunce? go on!'

'i'm a poor man,' the hatte"

toe toe tee toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe toe

…

Можно увидеть, что повторяются многократно слова.

Шестая эпоха:

Seed:" did not at all like the tone of this remark, and thought it would

be as well to introduce some othe"

the har ho the cad to the woree and the woree th the cout of the cad to the woree and the woree th the cout of the cad to the woree and the woree th the cout of the cad to the woree and the woree th the cout of the cad to the woree and the woree th the cout of the cad to the woree and the woree th the cout of the cad to the woree and the woree th the cout of the cad to the woree …

В данном случае сеть уже сгенерировала тоже повторяющуюся последовательность, но уже из нескольких слов.

Двенадцатая эпоха:

'that's nothing to what i could say if i chose,' the duchess replied, in

a pleased ton"

e.

'and thet i vo her she marter aage an an anllonsnnn tone. "the would saed to tee toeee to the soeer '

'i wool t so hen the marter ' said the monk turtle an an anllonsnnn tone. "the would saed to tee toeee to the soeer '

'i wool t so hen to toe toent ' said the monk turtle an an anllonsnnn tone. "the would saed to the toeee to the soeer '

'i wool t so hen the marter ' said the monk turtle an an anllonsnnn tone. "the would saed to tee toeee to the soeer '

В данном случае получились длинные предложения, которые начали повторяться.

Двадцатая эпоха:

Seed:"r and left foot, so as to prevent its undoing itself,) she carried

it out into the open air. 'if i d"

an toe toids tf tead to teye '

'io tou dlene" said the kock turtle. "thl would ' said the muck turtle,

'iede you mene th that i weal to tou,' thi gatter went on, 'and then the made tf think to thi goo toee a art of the sohee 'she mad to ten whet i wes lo tie rooe tf toine the roeer th then toen a toie of tointe the goule sfe white would be no eerirt thi was so fend an inr aacin an anlle the reater to the karter whuh the woodd 'bid then the mert aater in toe toiet the was in the rore tf toile to the then she was oo the riaee tfe was so tong thet whsh the lirtle garter and the…

В данном случае видно, что повторений получается мало, можно узнать некоторые слова, а также словосочетания.

Видно, что с увеличение количества эпох результат улучшается: повторяющиеся последовательности постепенно сокращаются, появляются какие-то знакомые слова и словосочетания.

Выводы:

В ходе выполнения данной лабораторной работы была построена модель, генерирующая текст на основе текста сказки «Алиса в стране чудес». Также была изучена система callback в Keras, благодаря которой были сохранены модели на каждой из эпох, и был отслежен прогресс обучения путем генерации текста на определенных эпохах.

ПРИЛОЖЕНИЕ А

```
import numpy as np
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
import keras.callbacks
from keras.utils import np_utils
import sys


filename = "wonderland.txt"
file_with_gen_text = "generated.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])
n_patterns = len(dataX)
print("Total Patterns: ", n_patterns)
X = np.reshape(dataX, (n_patterns, seq_length, 1))
X = X / float(n_vocab)
y = np_utils.to_categorical(dataY)
```

```
class MyCustomCallback(keras.callbacks.Callback):

    def __init__(self):
        super(MyCustomCallback, self).__init__()

    def on_epoch_end(self, epoch, logs=None):
        if (epoch+1) % 4 or (epoch+1) == 1:
            generate_symbols(self.model, epoch+1)



def generate_symbols(model_, epoch):
    g = open(file_with_gen_text, 'a')
    gen_symbols = []
    start = np.random.randint(0, len(dataX) - 1)
    pattern = dataX[start]
    g.write("On epoch " + str(epoch) + " seed is:\n" + "\"" +
''.join([int_to_char[value] for value in pattern]) + "\"\n")
    for i in range(1000):
        x = np.reshape(pattern, (1, len(pattern), 1))
        x = x / float(n_vocab)
        prediction = model_.predict(x, verbose=0)
        index = np.argmax(prediction)
        result = int_to_char[index]
        gen_symbols.append(result)
        pattern.append(index)
        pattern = pattern[1:len(pattern)]
    g.write("generated                                          text
is:\n"+"\""+''.join(gen_symbols)+"\"\n")
    g.close()



def build_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')
    return model



def run_fit():
    model = build_model()
    filepath="weights-improvement-{epoch:02d}-{loss:.4f}.hdf5"
    checkpoint    =    ModelCheckpoint(filepath,    monitor='loss',
verbose=1, save_best_only=True, mode='min')
    callbacks_list = [checkpoint, MyCustomCallback()]
    model.fit(X,        y,        epochs=20,        batch_size=128,
callbacks=callbacks_list)



def run_best_model():
    model = build_model()
    model.load_weights("weights-improvement-20-1.9430.hdf5")


    gen_symbols = []
    start = np.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 = np.reshape(pattern, (1, len(pattern), 1))
        x = x / float(n_vocab)
        prediction = model.predict(x, verbose=0)
        index = np.argmax(prediction)
        result = int_to_char[index]
        gen_symbols.append(result)
        pattern.append(index)
        pattern = pattern[1:len(pattern)]
    print("generated text is:\n"+"\""+''.join(gen_symbols)+"\"\n")



run_best_model()
```