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

ОТЧЕТ

по лабораторной работе №8

по дисциплине «Искусственные нейронные сети»

Тема: Генерация текста на основе «Алисы в Стране чудес»

Студент гр. 7383 Сычевский Р.А.

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.

Порядок выполнения работы.

1. Реализовать модель ИНС, которая будет генерировать текст
2. 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)
3. Отследить процесс обучения при помощи TensorFlowCallBack (TensorBoard), в отчете привести результаты и их анализ

Ход работы.

Для исследования была разработана и использована программа. Код программы приведен в приложении А.

Была построена рекуррентная сеть, модель сети представлена на рисунке 1.

```
model = Sequential()
model.add(LSTM(256, input_shape=(shapeIn[1], shapeIn[2])))
model.add(Dropout(0.2))
model.add(Dense(shapeOut[1], activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam')
```

Рисунок 1 – Модель сети

Для контроля обучения был написан callback, который выводит в консоль сгенерированный код после определенных эпох.

Рассмотрим сгенерированный текст:

1) первая эпоха:

Seed:
" und also, and
all of them bowed low.

'would you tell me,' said alice, a little timidly, 'why you ar "
e to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to
to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to
to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to
to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to
to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to
to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to
to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to
to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to
to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to to

2)шестая эпоха:

Seed:
" t one of the trees had a door
leading right into it. 'that's very curious!' she thought. 'but
everyt "
 to the woet toe toet to the toete the woete the woete the woete the woete the woete the woete the
woete the woete the woete the woete the woete the woete the woete the woete the woete the
woete the woete the woete the woete the woete the woete the woete the woete the woete the
woete the woete the woete the woete the woete the woete the woete the woete the woete the
woete the woete the woete the woete the woete the woete the woete the woete the woete the
woete the woete the woete the woete the woete the woete the woete the woete the woete the
woete the woete the woete the woete the woete the woete the woete the woete the woete the
woete the woete the woete the woete the woete the woete the woete the woete the woete the
woete the woete the woete the woete the woete the woete the woete the woete the woete the
woete the woete the woete the woete the woete the woete the woete the woete the woete the
woete the woete the woete the woete the woete the woete the woete the woete the woete the w

3)одиннадцатая эпоха:

Seed:

" ctronic works in formats readable by the widest variety of computers
including obsolete, old, middle "

to the sorde the soree tht sooe the wose and the woile tas io a lortee thre and the woile tas io the woree th the harte and the woile tas io a lortee thre and the woile tas io the woree th the harte and the woile tas io a lortee thre and the woile tas io the woree th the harte and the woile tas io a lortee thre and the woile tas io the woree th the harte and the woile tas io a lortee thre and the woile tas io the woree th the harte and the woile tas io a lortee thre and the woile tas io the woree th the harte and the woile tas io a lortee thre and the woile tas io the woree th the harte and the woile tas io a lortee thre and the woile tas io the woree th the harte and the woile tas io a lortee thre and the woile tas io the woree th the harte and the woile tas io a lortee thre and the woile tas io the woree th the harte and the woile tas io a lortee thre and the woile tas io the woree th the harte and the woile tas io a lortee thre and the woile tas io the woree

Можно заметить, что сгенерированный текст представляет повторяющиеся последовательности слов, в которых могу быть допущены ошибки. При небольшом количестве эпох, сгенерированные последовательности очень короткие, но с ростом количества эпох, растет и длина последовательности.

Выводы.

В ходе выполнения данной работы была разработана рекуррентная сеть для генерации текстов. Модель была обучена на основе книги «Алиса в Стране чудес». Был написан callback, с помощью которого осуществлялось отслеживание процесса обучения сети.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

```
import os
import requests
import numpy as np
import sys
from keras import Sequential
from keras.callbacks import ModelCheckpoint, EarlyStopping
from keras.layers import LSTM, Dropout, Dense
import tensorflow.keras.callbacks
from keras.utils import np_utils

BOOK_URL = "https://www.gutenberg.org/cache/epub/11/pg11.txt"
BOOK_PATH = "./alice's_adventures_in_wonderland.txt"


EPOCHS = 13
BATCH_SIZE = 256


if not os.path.exists(BOOK_PATH):
    f = requests.get(BOOK_URL)
    open(BOOK_PATH, 'wb').write(f.content)
raw_text = open(BOOK_PATH).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)
dataX = []
dataY = []
seq_length = 100
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)
X = np.reshape(dataX, (n_patterns, seq_length, 1))
X = X / float(n_vocab)
Y = np_utils.to_categorical(dataY)

def makeSequence(model):
    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]
        seq_in = [int_to_char[value] for value in pattern]
        sys.stdout.write(result)
        pattern.append(index)
        pattern = pattern[1:len(pattern)]

def buildModel(shapeIn, shapeOut):
    model = Sequential()
    model.add(LSTM(256, input_shape=(shapeIn[1], shapeIn[2])))
    model.add(Dropout(0.2))
    model.add(Dense(shapeOut[1], activation='softmax'))
    model.compile(loss='categorical_crossentropy', optimizer='adam')
    return model

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

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

filepath="weights-improvement-{epoch:02d}-{loss:.4f}.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='loss', verbose=1,
save_best_only=True, mode='min')
earlyStopping = EarlyStopping(monitor='loss', verbose=1, mode='min',
patience=2)
callbacks_list = [checkpoint, earlyStopping, myCallback([0, 5, 10])]
model = buildModel(X.shape, Y.shape)
model.fit(X, Y, epochs=EPOCHS, batch_size=BATCH_SIZE,
callbacks=callbacks_list)
```