МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №8

по дисциплине «Искуственные нейронные сети»

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

Студент гр. 8382 _____ Щемель Д.А.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2021

Цель

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

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 генирировать и выводить текст у необученной модели)
3. Отследить процесс обучения при помощи TensorFlowCallBack (TensorBoard), в отчете привести результаты и их анализ

Ход работы

Архитектура сети:

```
LSTM(256)
Dropout(0.2)
Dense(y.shape[1])
```

Генерация текста:

Модель обучалась в течение 20 эпох и на каждой нечётной эпохе выводился сгенерированный текст.

Структура вывода: Номер эпохи: Seed: “Шаблон для генерации” Сгенерированный текст

```
1:
Seed:
"d
away,' but they began running when they liked, and left off when they
liked, so that it was not ea"

 the the the the the the the the the the the the the the the the the the the th

3:
Seed:
"minded their own business,' the duchess said in a hoarse
growl, 'the world would go round a deal fas"
e toe  and the  and toe  and the  and toe  and the  and toe  and the  and toe  and

5:
Seed:
"all,' said the white rabbit; 'in fact, there's
nothing written on the outside.' he unfolded the pape"
e toet the woete to the toete th the tooee th the tooee th the tooee th the tooee t

7:
Seed:
" the
```

officers: but the hatter was out of sight before the officer could get
to the door.

'call the "
toie to toe toee to toe toee to the toiee ' shi gaterr aeren an anl aeren an an   '

9:
Seed:
"ould, for her neck kept getting entangled among the branches, and
every now and then she had to stop"
 the while tar io the toeee th the thne th the thne th the thne the was so tee to

11:
Seed:
"ou'd rather not.'

'we indeed!' cried the mouse, who was trembling down to the end of his
tail. 'as "
the was a little oi the cand '

13:
Seed:
"it didn't sound at all the
right word) '--but i shall have to ask them what the name of the country
"
'so tou to teat the moot of the sore-
the harter was iowt at the woide oo the courd she had not th

15:
Seed:
"en, among the bright flower-beds and the cool
fountains.

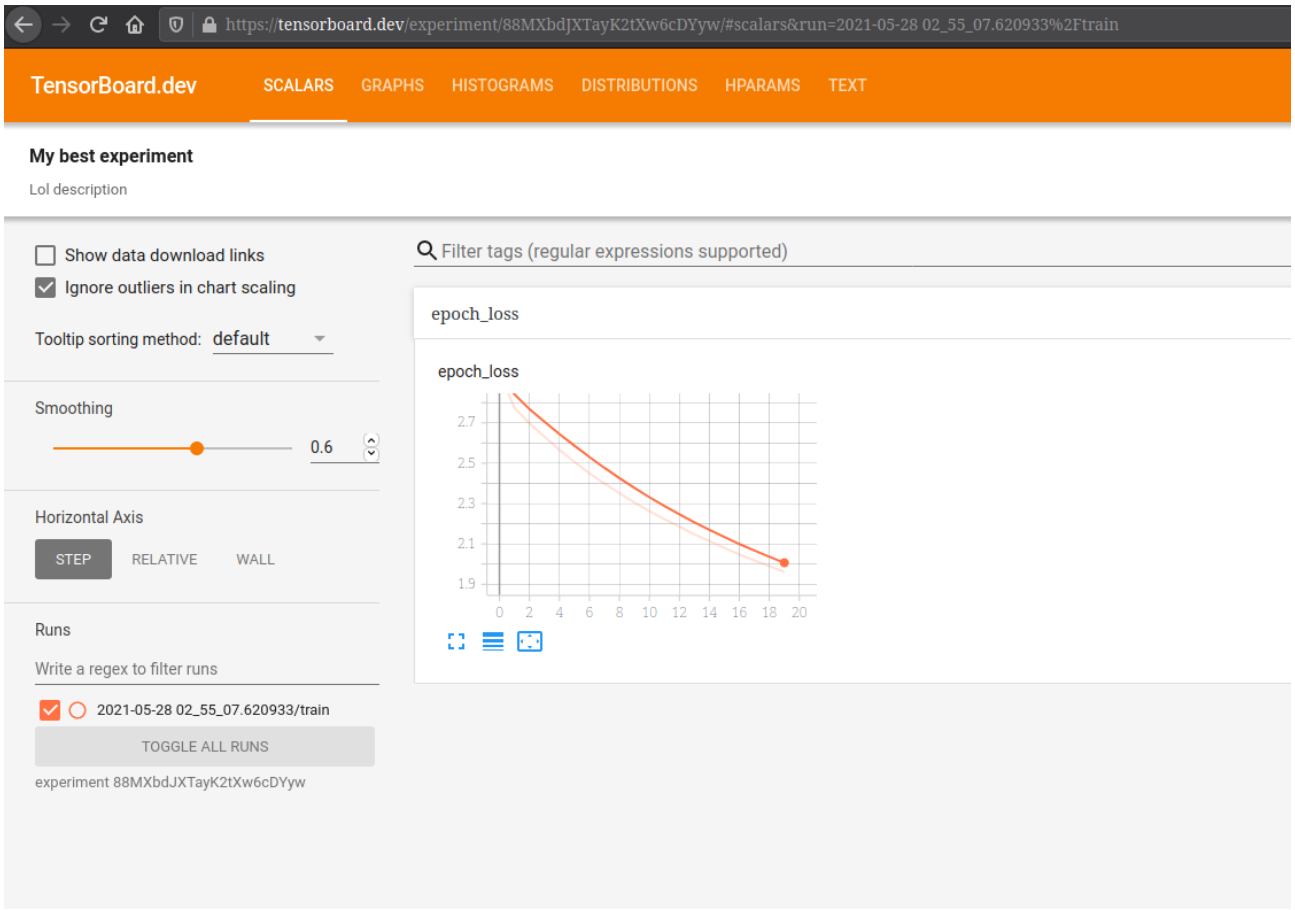
```
chapter viii. the queen's croquet-groun"



   '


17:
Seed:
" lie down upon their
faces, so that they couldn't see it?' so she stood still where she was,
and wai"
 so the wan so the sabbit  soe care an all toee a firtle  and she tai to tee it aal

19:
Seed:
"

'it goes on, you know,' the hatter continued, 'in this way:--

     "up above the world you fly,
 "
  ie thet wou donw thet toee th the
             whal the oook of the sooe

            wou  wow! w
```

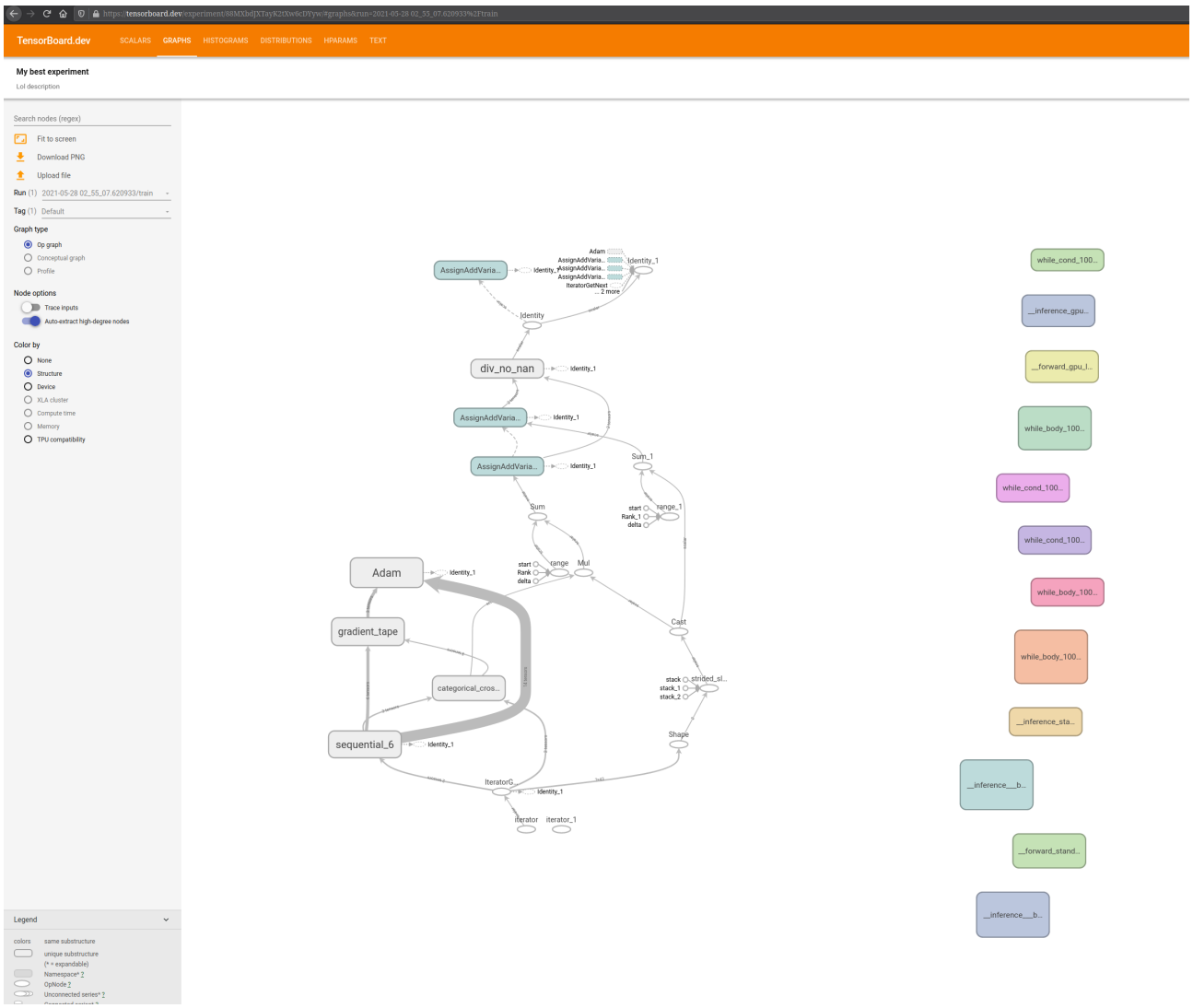
На 15ой эпохе видно, что генерируемый текст крайне зависим от форматирования шаблона для генерации, потому что из шаблона, содержащего 7 переносов строк сеть сгенерировала пустой текст.

На 19ой эпохе генерируются почти существующие слова, но с неправильными буквами.

TensorBoard:

На скриншотах ниже приведены график обучения сети и граф сети.




https://tensorboard.dev/experiment/88MXbdJXTayK2tXw6cDYyw/#graphs&run=2021-05-28 02_55_07.620933%2Ftrain
TensorBoard.dev
SCALARS
GRAPHS
HISTOGRAMS
DISTRIBUTIONS
HPARAMS
TEXT
My best experiment
Lol description
Search nodes (regex)
Fit to screen
Download PNG
Upload file
Run (1)
2021-05-28 02_55_07.620933/train
Tag (1)
Default
Graph type
Op graph
Conceptual graph
Profile
Node options
Trace inputs
Auto-extract high-degree nodes
Color by
None
Structure
Device
XLA cluster
Compute time
Memory
TPU compatibility
Legend
colors
same substructure
unique substructure
(* = expandable)
Namespace* ?
OpNode ?
Unconnected series* ?
AssignAddVaria...
Identity_1
Identity
div_no_nan
Sum_1
Sum
range_1
range
Mul
Adam
gradient_tape
Cast
categorical_cros...
strided_sl...
Shape
sequential_6
IteratorG...
iterator
iterator_1
while_cond_100...
__inference_gpu...
__forward_gpu_l...
while_body_100...
__inference_sta...
__forward_stand...

Вывод

В ходе лабораторной работы была написана программа, генерирующая текст, на основе произведения “Алиса в стране чудес”.

```
import sys
from datetime import datetime
from typing import Dict, List, Tuple

import numpy as np
from keras.callbacks import Callback, ModelCheckpoint
from keras.layers import LSTM, Dense, Dropout
from keras.models import Model, Sequential
from keras.utils import np_utils
from tensorflow.python.keras.callbacks import TensorBoard

FILENAME = "wonderland.txt"
FILENAME_PATTERN = "weights-improvement-{epoch:02d}-{loss:.4f}.hdf5"


class TextGeneratorCallback(Callback):
    def __init__(self, int_to_char: Dict[int, str], data_x: np.array, n_vocab) -> N
        super().__init__()
        self._int_to_char = int_to_char
        self._data_x = data_x
        self._n_vocab = n_vocab

    def on_epoch_end(self, epoch, logs):
        if epoch % 2 != 0:
            return
        start = np.random.randint(0, len(self._data_x) - 1)
        pattern = self._data_x[start]
        print("Seed:")
        print(f'"{"".join([self._int_to_char[value] for value in pattern])}"')
        for i in range(100):
            x = np.reshape(pattern, (1, len(pattern), 1))
            x = x / float(self._n_vocab)
```

```
            prediction = self.model.predict(x, verbose=0)
            index = np.argmax(prediction)
            result = self._int_to_char[index]
            sys.stdout.write(result)
            pattern.append(index)
            pattern = pattern[1 : len(pattern)]
        print()


def read_text(seq_length=100) -> Tuple[np.array, np.array, Dict, List[int], int]:
    raw_text = open(FILENAME).read().lower()
    chars = sorted(list(set(raw_text)))

    char_to_int = {c: i for i, c in enumerate(chars)}
    int_to_char = {i: c for i, c in enumerate(chars)}
    n_chars = len(raw_text)
    n_vocab = len(chars)

    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)

    x = np.reshape(dataX, (n_patterns, seq_length, 1))
    x = x / float(n_vocab)

    y = np_utils.to_categorical(dataY)
```

```
    return x, y, int_to_char, dataX, n_vocab


def create_model(x_shape, y_shape) -> 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 main() -> None:
    x, y, int_to_char, data_x, n_vocab = read_text()
    checkpoint = ModelCheckpoint(
        FILENAME_PATTERN, monitor="loss", verbose=1, save_best_only=True, mode="min
    )
    callbacks = [
        checkpoint,
        TextGeneratorCallback(int_to_char, data_x, n_vocab),
        TensorBoard(log_dir=f"logs/{datetime.now()}"),
    ]
    model = create_model(x.shape, y.shape)
    model.fit(x, y, epochs=20, batch_size=128, callbacks=callbacks)


if __name__ == "__main__":
    main()
```