

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра технічної кібернетики

Звіт до комп'ютерного практикуму 8 з дисципліни:
**“Програмні засоби проектування та реалізації
нейромережових систем”**

Виконав
ІП-01 Черпак А.В.

Перевірив:
Шимкович В.М.

Комп'ютерний практикум 4

Тема: Нейронні мережі CNN-bi-LSTM для розпізнавання звуку

Завдання:

Написати програму, що реалізує нейронну мережу типу CNN-bi-LSTM для розпізнавання мови в тексті. Використати датасет `librispeech`: <https://www.tensorflow.org/datasets/catalog/librispeech>

Виконання:

Створимо методи для побудови моделі та створення функції втрат CTC:

```
from keras.layers import ReLU, Dense, LSTM, Dropout, BatchNormalization, Conv2D, Reshape, Input, Bidirectional
import tensorflow as tf

def buildModel(input_dim, output_dim, numOfRNN, numOfRNNUnits):
    input = Input((None, input_dim))
    x = Reshape((-1, input_dim, 1))(input)
    x = Conv2D(filters=32, kernel_size=[11, 41], strides=[2, 2], padding="same", use_bias=False)(x)
    x = BatchNormalization()(x)
    x = ReLU()(x)
    x = Conv2D(filters=32, kernel_size=[11, 21], strides=[1, 2], padding="same", use_bias=False)(x)
    x = BatchNormalization()(x)
    x = ReLU()(x)
    x = Reshape((-1, x.shape[-2] * x.shape[-1]))(x)
    for i in range(numOfRNN):
        recurrent = LSTM(units=numOfRNNUnits, activation="tanh", recurrent_activation="sigmoid", use_bias=True, return_sequences=True)
        x = Bidirectional(recurrent, merge_mode="concat")(x)
        if i < numOfRNN - 1:
            x = Dropout(rate=0.5)(x)

    x = Dense(numOfRNNUnits * 2)(x)
    x = ReLU()(x)
    x = Dropout(rate=0.5)(x)
    output = Dense(output_dim + 1, activation="softmax")(x)
    model = tf.keras.Model(input, output)
    model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=1e-4), loss=CTCLoss)
    return model

def CTCLoss(y_true, y_pred):
    batchLen = tf.cast(tf.shape(y_true)[0], "int64")
    inputLen = tf.cast(tf.shape(y_pred)[1], "int64") * tf.ones(shape=(batchLen, 1), dtype="int64")
    labelLen = tf.cast(tf.shape(y_true)[1], "int64") * tf.ones(shape=(batchLen, 1), dtype="int64")
    loss = tf.keras.backend.ctc_batch_cost(y_true, y_pred, inputLen, labelLen)
    return loss
```

Створимо метод для декодування передбачень, а також колбек, який наприкінці кожної епохи візуалізуватиме результат тестування:

```
import tensorflow as tf
import numpy as np
```

```

from jiwer import wer

characters = [x for x in "abcdefghijklmnopqrstuvwxyz'?! "]
charToNum = tf.keras.layers.StringLookup(vocabulary=characters, oov_token='')
numToChar = tf.keras.layers.StringLookup(vocabulary=charToNum.get_vocabulary(),
oov_token='', invert=True)

def decodePredictions(pred):
    inputLen = np.ones(pred.shape[0]) * pred.shape[1]
    results = tf.keras.backend.ctc_decode(pred, input_length=inputLen,
greedy=True)[0][0]
    outputText = [tf.strings.reduce_join(numToChar(result)).numpy().decode('utf-
8') for result in results]
    return outputText

class CallbackEval(tf.keras.callbacks.Callback):
    def __init__(self, dataset, model):
        super().__init__()
        self.dataset = dataset
        self.model = model

    def on_epoch_end(self, epoch: int, logs=None):
        predictions = []
        targets = []
        for spectrograms, labels in self.dataset:
            batch_predictions = self.model.predict(spectrograms, verbose=0)
            batch_predictions = decodePredictions(batch_predictions)
            predictions.extend(batch_predictions)
            for label in labels:
                label =
(tf.strings.reduce_join(numToChar(label)).numpy().decode('utf-8'))
                targets.append(label)
            wer_score = wer(targets, predictions)
            print('-' * 100)
            print(f'Word Error Rate: {wer_score:.4f}')
            print('-' * 100)
            for i in np.random.randint(0, len(predictions), 2):
                print(f'Target : {targets[i]}')
                print(f'Prediction: {predictions[i]}')
                print('-' * 100)

```

У головному файлі також створимо функцію для обробки кожної точки даних, тобто переведення аудіо у спектрограму, а текст – у набір міток:

```

import tensorflow_datasets as tfds
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt

from Lab8.CallbackEval import CallbackEval
from nn_model import buildModel

def processSample(label, audio):
    audio = tf.cast(audio, tf.float32)
    spectr = tf.signal.stft(audio, frame_length=256, frame_step=160,
fft_length=fft_length)
    spectr = tf.math.pow(tf.math.abs(spectr), 0.5)
    mean = tf.math.reduce_mean(spectr, 1, keepdims=True)
    std = tf.math.reduce_std(spectr, 1, keepdims=True)
    spectr = (spectr - mean) / (std + 1e-10)
    label = charToNum(tf.strings.unicode_split(tf.strings.lower(label),
input_encoding='UTF-8'))
    return spectr, label

```

Тепер у розділі `__main__` завантажимо необхідний датасет, одразу розділимо

його на тренувальну та тесту вальну вибірки. Також визначимо набір символів, які розпізнаватимемо. Виведемо текст одного з семплів, а також візуалізуємо його спектрограму. Потім створимо модель, виведемо дані про неї та проведемо 5 епох навчання.

```
if __name__ == '__main__':
    trainData, testData = tfds.load("ljspeech", split=["train[:90%]",
"train[90%:]"], as_supervised=True)
    characters = [x for x in "abcdefghijklmnopqrstuvwxyz'?! "]
    charToNum = tf.keras.layers.StringLookup(vocabulary=characters, oov_token='')
    numToChar =
tf.keras.layers.StringLookup(vocabulary=charToNum.get_vocabulary(), oov_token='',
invert=True)
    SAMPLE_RATE = 22050
    fft_length = 384

    trainPreprocessed = (trainData.map(processSample,
num_parallel_calls=tf.data.AUTOTUNE).padded_batch(32).prefetch(buffer_size=tf.data
.AUTOTUNE))
    testPreprocessed = (testData.map(processSample,
num_parallel_calls=tf.data.AUTOTUNE).padded_batch(32).prefetch(buffer_size=tf.data
.AUTOTUNE))
    for spectrograms, labels in trainPreprocessed.take(1):
        spectrogram = spectrograms[1].numpy()
        spectrogram = np.array([np.trim_zeros(x) for x in
np.transpose(spectrogram)])
        label = labels[1]
        label = tf.strings.reduce_join(numToChar(label)).numpy().decode('utf-8')
        plt.imshow(spectrogram, vmax=1)
        print(label)

    model = buildModel(fft_length // 2 + 1, charToNum.vocabulary_size(), 5, 512)
    model.summary()




    callback = CallbackEval(testPreprocessed, model)
    history = model.fit(trainPreprocessed, validation_data=testPreprocessed,
epochs=5, callbacks=[callback])
    callback.on_epoch_end(5)

    model.save('/model.h5')
```

Для тренування нейромережі було використано Google Colab, оскільки на моєму пристрої тренування проходило нереально повільно.

Тепер наведемо результати виконання даного коду.

Результати завантаження датасету:

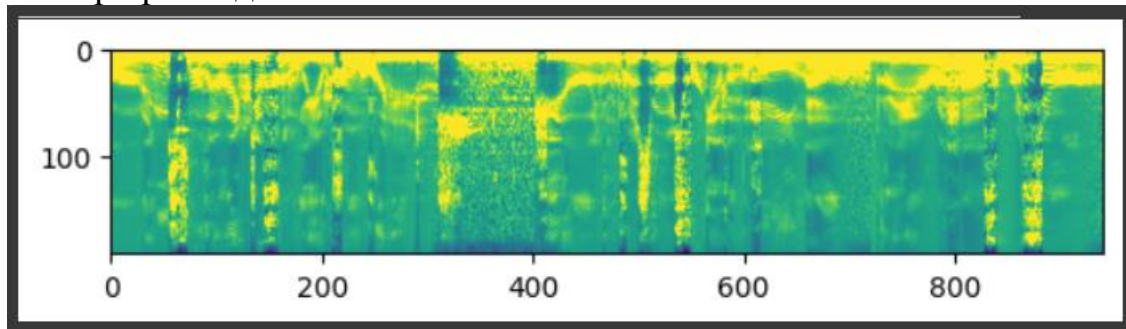
```
WARNING:absl:You use TensorFlow DType <dtype: 'int16'> in tfds.features This will soon be deprecated in favor of NumPy DTypes
Downloading and preparing dataset 2.56 GiB (download: 2.56 GiB, generated: 10.73 GiB, total: 13.29 GiB) to /root/tensorflow_
DI Completed...: 100%  1/1 [07:16<00:00, 76.66s/ url]
DI Size...: 100%  2621/2621 [07:16<00:00, 34.83 MiB/s]
Extraction completed...: 100%  13102/13102 [07:16<00:00, 1114.71 file/s]
WARNING:absl:`FeatureConnector.dtype` is deprecated. Please change your code to use NumPy with the field `FeatureConnector.np
Dataset ljspeech downloaded and prepared to /root/tensorflow_datasets/ljspeech/1.1.1. Subsequent calls will reuse this data.
```

Текст одного з семплів та дані про модель:

earnings were very differently appropriated here the prisoners were given the whole amount there a half or a third
Model: "model"

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, None, 193)]	0
reshape (Reshape)	(None, None, 193, 1)	0
conv2d (Conv2D)	(None, None, 97, 32)	14432
batch_normalization (Batch Normalization)	(None, None, 97, 32)	128
re_lu (ReLU)	(None, None, 97, 32)	0
conv2d_1 (Conv2D)	(None, None, 49, 32)	236544
batch_normalization_1 (Batch Normalization)	(None, None, 49, 32)	128
re_lu_1 (ReLU)	(None, None, 49, 32)	0
reshape_1 (Reshape)	(None, None, 1568)	0
bidirectional (Bidirectional)	(None, None, 1024)	8523776
dropout (Dropout)	(None, None, 1024)	0
bidirectional_1 (Bidirectional)	(None, None, 1024)	6295552
dropout_1 (Dropout)	(None, None, 1024)	0
bidirectional_2 (Bidirectional)	(None, None, 1024)	6295552
dropout_2 (Dropout)	(None, None, 1024)	0
bidirectional_3 (Bidirectional)	(None, None, 1024)	6295552
dropout_3 (Dropout)	(None, None, 1024)	0
bidirectional_4 (Bidirectional)	(None, None, 1024)	6295552
dense (Dense)	(None, None, 1024)	1049600
re_lu_2 (ReLU)	(None, None, 1024)	0
dropout_4 (Dropout)	(None, None, 1024)	0
dense_1 (Dense)	(None, None, 32)	32800
=====		
Total params: 35,039,616		
Trainable params: 35,039,488		
Non-trainable params: 128		

Спектрограма одного з семплів:



Результати тренування моделі:

```
Epoch 1/3
369/369 [=====] - ETA: 0s - loss: 314.3067-----
Word Error Rate: 1.0000

Target : the secret service had no standard procedure for the systematic review of its requests for and receipt of information from other federal agencies
Prediction:

Target : they saw and heard brennan describing what he had seen
Prediction:

369/369 [=====] - 2206s 6s/step - loss: 314.3067 - val_loss: 309.5638
Epoch 2/3
369/369 [=====] - ETA: 0s - loss: 275.1676-----
Word Error Rate: 0.9997

Target : all that the hangman whoever he may be does under the new regime is to unhook the halter and remove the pinioning straps
Prediction: r

Target : but worse than the bankruptcy was the confession made by the partners in the court
Prediction:

369/369 [=====] - 1731s 5s/step - loss: 275.1676 - val_loss: 227.0847
Epoch 3/3
369/369 [=====] - ETA: 0s - loss: 175.3123-----
Word Error Rate: 0.7909

Target : the roof of the female prison says the grand jury in their presentment in eighteen thirteen let in the rain
Prediction: the rofof the feml prison ss the ranr in therprisen men tin eten thertein litin the rin

Target : thus a sound public policy has been defeated
Prediction: u a sn ou bwa pal sa as bn difaed

369/369 [=====] - 1720s 5s/step - loss: 175.3123 - val_loss: 119.9174
Epoch 1/2
369/369 [=====] - ETA: 0s - loss: 113.8995-----
Word Error Rate: 0.6323
```

Про всяк випадок після трьох епох тренування я зберіг модень, а потім провів ще 2 епохи навчання.

```
Epoch 1/2
369/369 [=====] - ETA: 0s - loss: 113.8995-----
Word Error Rate: 0.6323

Target : he sought for himself a place in history a role as the great man who would be recognized as having been in advance of his times
Prediction: he saut for hamself u plac ion hisstry ar ro as the great man who ad be recanis is having binin inadvens of his time

Target : the fbi which was established within the department of justice in nineteen oh eight has had in recent years an increasingly important role to play
Prediction: the fb which was a stablich within the parent of justics an nineteno egt has hain reanon ars and increionly mporttan torwald to ga

369/369 [=====] - 1679s 5s/step - loss: 113.8995 - val_loss: 86.7535
Epoch 2/2
369/369 [=====] - ETA: 0s - loss: 87.5658-----
Word Error Rate: 0.5279

Target : from where several witnesses last saw oswald running west on jefferson boulevard shortly after the tippit murder
Prediction: from where several wicesis las sa oswald runing west unjefer and bulievard shortly after the tipit murder

Target : and when the inspectors visited newgate they found the three certain to die in a dayroom by themselves
Prediction: and when the ispec as bisited newgat they found the thre sertin to die in ada rom by themsels

369/369 [=====] - 1703s 5s/step - loss: 87.5658 - val_loss: 71.3065
Word Error Rate: 0.5279

Target : and then only went round to count the number
Prediction: and then only whent round to count the nubur

Target : subsistence and health of large numbers in the community then
Prediction: soub sistence sand hel of lrge numbers in thit cimmunity ven
```

Як бачимо, вже після 5 епох точність досягла майже 50%, що насправді непогано для такої складної задачі. І, як бачимо з першого семплу з останнього тестування, насправді помилок геть небагато, і більшість з них незначні. А з останнім семплом пощастило менше.

Висновок:

Під час виконання комп'ютерного практикуму ми реалізували гібридну двонаправлену CNN-biLSTM нейромережу для розпізнавання тексту з уривків звукозаписів із датасету ljspeech. Після 5 епох тренувань ми отримали точність 47,21%, при чому, деякі уривки були розпізнані з похибкою буквально у три

літери, що насправді дуже навіть непогано. Щоправда, її тренування на моєму ноутбукі проходило вкрай повільно, тому довелося використовувати google colab з підтримкою GPU. У такому випадку тренування тривало близько 4 годин.