Московский Государственный Технический Университет имени Н. Э. Баумана

Факультет «Информатика и системы управления»
Кафедра «Автоматизированные системы обработки информации и управления»

Отчёт по домашнему заданию № 2 по дисциплине «Проектирование интеллектуальных систем»

Исполнитель:
Саврасов П.А.
Группа ИУ5-24М
«__» _____ 2021 г.

Преподаватель:
Терехов В.И.
_____
«__» _____ 2021 г.

Москва, 2021 г.

1. Задание

Обучить нейронную сеть на собственном датасете, созданном в домашнем задании №1.

2. Предобработка текста

```
import json
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer

from sklearn.datasets import fetch_20newsgroups
```

В даненом домашнем задании сравним несколько моделей (MLP, RNN) в задаче классификации текста, так же сравним несколько векторизаторов текста.

Для начала загрузим датасет из домашнегно задания 1

```
with open('messages.json', 'r') as file:
    data = json.loads(file.read())

data = pd.json_normalize(data['messages'])
```

Для векторизации текстов применим 2 векторизатора: TfidfVectorizer, CountVectorizer

```
xData = []
inputDims = []


vectorizer = TfidfVectorizer()
xData.append(vectorizer.fit_transform(data['content']).toarray())
inputDims.append(len(vectorizer.get_feature_names()))

vectorizer = CountVectorizer()
xData.append(vectorizer.fit_transform(data['content']).toarray())
inputDims.append(len(vectorizer.get_feature_names()))
```

3. Обучение нейронной сети:

```
models = []
for inputDim in inputDims:

    model = tf.keras.Sequential(
        [
            layers.InputLayer(input_shape=(inputDim)),
            layers.Dense(300, activation='relu'),
            layers.Dense(200, activation='relu'),
            layers.Dense(100, activation='relu'),
            layers.Dense(2, activation='relu')
        ]
    )
    models.append(model)
```

```
for model in models:
    model.compile (
        loss=keras.losses.SparseCategoricalCrossentropy(from_logits=True),
        optimizer=keras.optimizers.Adam(),
        metrics=['accuracy']
    )
```

```
xTrain = []
xTest = []
yTrain = []
yTest = []


for x in xData:
    x_train, x_test, y_train, y_test = train_test_split(x, data['target'], test_size=0.3, random_state=42)
    xTrain.append(x_train)
    xTest.append(x_test)
    yTrain.append(y_train)
    yTest.append(y_test)

historys = []

for i in range(2):
    print(f'\n\nModel {i + 1}---------------')
    historys.append(models[i].fit(xTrain[i], yTrain[i],epochs=10, verbose=2))
    models[i].evaluate(xTest[i], yTest[i], verbose=2)
```

```
fig, ax = plt.subplots(figsize=(10,10))
plt.subplot(2, 1, 1)
plt.plot(historys[0].epoch, historys[0].history['accuracy'], '-o')
plt.plot(historys[1].epoch, historys[1].history['accuracy'], '-o')

plt.subplot(2, 1, 2)
plt.plot(historys[0].epoch, historys[0].history['loss'], '-o')
plt.plot(historys[1].epoch, historys[1].history['loss'], '-o')

plt.show()
```