ДОДАТОК А

Програмний код

Лістинг А.1 – Реалізація візуального зчитування по губам на основі салієнтних ознак

```
from keras.models import Sequential # To initialise the nn as a sequence of layers
from keras.layers import Convolution2D # To make the convolution layer for 2D images
from keras.layers import MaxPooling2D #
from keras.layers import GlobalAveragePooling2D
from keras.layers import Flatten
from keras.layers import Dense
from keras.layers import Dropout
from keras.callbacks import CSVLogger
from keras.optimizers import RMSprop
from keras.layers import BatchNormalization
from keras.optimizers import Adam
from keras.models import load_model
from keras.callbacks import ModelCheckpoint
model = load_model('1_rmsprop_new.h5')
model.compile(optimizer = 'rmsprop', loss = 'categorical_crossentropy', metrics = ['accuracy'])
model.summary()
W = model.get_weights()
import numpy as np
W = np.array(W)
W.shape
W[0].shape
lay = []
for l in model.layers:
    lay.append(l.name)
    print(l.weights)
```

```
i = 0
```

Продовження лістингу А.1 – Реалізація візуального зчитування по губам на основі салієнтних ознак

```
for w in W:
    print(i,'-->{}'.format(w.shape))
    i += 1
import matplotlib.pyplot as plt
%matplotlib inline
import cv2
img = cv2.imread('F01_phrases02_01.jpg',0)
plt.imshow(img)
from keras.models import Model
import math
def plot_conv_weights(weights, input_channel=0):
    # Get the lowest and highest values for the weights.
    # This is used to correct the colour intensity across
    # the images so they can be compared with each other.
    w_min = np.min(weights)
    w_max = np.max(weights)

    # Number of filters used in the conv. layer.
    num_filters = weights.shape[3]

    # Number of grids to plot.
    # Rounded-up, square-root of the number of filters.
    num_grids = math.ceil(math.sqrt(num_filters))

    # Create figure with a grid of sub-plots.
    fig, axes = plt.subplots(num_grids, num_grids)

    # Plot all the filter-weights.
    for i, ax in enumerate(axes.flat):
        # Only plot the valid filter-weights.
        if i<num_filters:
```

Продовження лістингу А.1 – Реалізація візуального зчитування по губам на основі салієнтних ознак

```
    # Get the weights for the i'th filter of the input channel.
                # See new_conv_layer() for details on the format
                # of this 4-dim tensor.
                img = weights[:, :, input_channel, i]

                # Plot image.
                ax.imshow(img, vmin=w_min, vmax=w_max,
                          interpolation='nearest', cmap='seismic')

            # Remove ticks from the plot.
            ax.set_xticks([])
            ax.set_yticks([])

        # Ensure the plot is shown correctly with multiple plots
        # in a single Notebook cell.
        plt.show()
    weights_convo1 = layers[0].get_weights()[0]
    weights_convo1.shape
    plot_conv_weights(weights=weights_convo1, input_channel=0)
    weights_convo2 = layers[1].get_weights()[0]
    weights_convo2.shape
    plot_conv_weights(weights=weights_convo2, input_channel=0)
    def plot_conv_output(values):
        # Number of filters used in the conv. layer.
        num_filters = values.shape[3]

        # Number of grids to plot.
        # Rounded-up, square-root of the number of filters.
        num_grids = math.ceil(math.sqrt(num_filters))

        # Create figure with a grid of sub-plots.
```

Продовження лістингу А.1 – Реалізація візуального зчитування по губам на основі салієнтних ознак

```
        fig, axes = plt.subplots(num_grids, num_grids, figsize = (20,20))

        # Plot the output images of all the filters.
        for i, ax in enumerate(axes.flat):
            # Only plot the images for valid filters.
            if i<num_filters:
                # Get the output image of using the i'th filter.
                img = values[0, :, :, i]

                # Plot image.
                ax.imshow(img,              interpolation='nearest', cmap='binary')

            # Remove ticks from the plot.
            ax.set_xticks([])
            ax.set_yticks([])

        # Ensure the plot is shown correctly with multiple plots
        # in a single Notebook cell.
        plt.show()
    from keras import backend as K
    img = np.reshape(img,(1,224,224,1))
    img.shape
    output_conv1 = K.function(inputs= model.inputs,
                              outputs=[layers[0].output])
    layer_output1 = output_conv1([0+img])
    layer_output1[0].shape
    plot_conv_output(layer_output1[0])
    output_conv2 = K.function(inputs=model.inputs,
                              outputs=[layers[1].output])
    layer_output2 = output_conv2([img])
```

Продовження лістингу А.1 – Реалізація візуального зчитування по губам на основі салієнтних ознак

```
plot_conv_output(layer_output2[0])
output_conv3 = K.function(inputs=model.inputs,
                          outputs=[layers[2].output])
layer_output3 = output_conv3([img])
plot_conv_output(layer_output3[0])
output_conv4 = K.function(inputs=model.inputs,
                          outputs=[layers[3].output])
layer_output4 = output_conv4([img])
plot_conv_output(layer_output4[0])
output_conv5 = K.function(inputs=model.inputs,
                          outputs=[layers[4].output])
layer_output5 = output_conv5([img])
plot_conv_output(layer_output5[0])
```

ДОДАТОК Б

Відомість кваліфікаційної роботи

Позначення	Найменування	Дод. відомості
	Текстові документи	
1.	Пояснювальна записка	66 с.
	Інші документи	
2.	Презентаційні матеріали	__ плакатів

.	.	Прізвище та ініціали.	Підп.	Дата		
Розробив		Онищенко А.Г.			Глибинна рекурентна нейронна мережа та її навчання в завданні розпізнавання мови по губах	Шифр групи: СШІм-20-1; Код напр./спец.: 122
Перевірив		Бодянський Є.В.				
Н.контр.		Малєєва І.А.				ХНУРЕ кафедра ШІ
Затв.		Філатов В.О.				