Semester Project

Write your name
1. Prathyusha Reddy Thumma- (pthumma)
2. Kavya Sri Kasarla -(kkasarla)
3. Bharath chowdary Anumolu-(banumolu)[1]*

Abstract
BACKGROUND: Face recognition, handwritten digit recognition has become an appreciable area of research for the practical approaches but this task of making new approaches has become difficult due to the existing data problem and defect in the the data.So, this classification of faces and digits make automatic classification a difficult tasks. Classify task with higher accuracy and proposing models for dynamism data sets has become a challenging task.
OBJECTIVE: Deep Convolutional Neural Networks (DCNNs) have emerged as a popular architecture for facial emotion classification due to their ability to learn complex features directly from raw image data. DCNNs have demonstrated state-of-the-art performance on various facial emotion datasets, including the FER2013 dataset, which contains over 35,000 facial expression images.
METHODS: In this project, we aim to develop a DCNN model for the classification of facial emotions using the dataset. We employ various techniques like ELU activation, he-normal kernel initializer, dropouts, and batch normalization to improve the model's generalization performance.
We also explore data augmentation techniques to further enhance the model's performance. The developed model can potentially aid in emotion recognition and effective computing applications, contributing towards a better understanding of human-computer interaction.

Keywords
Convolution layer, Deep Convolution Neural Network

[1] *Luddy School of Informatics, Computing, and Engineering, Indiana University, Bloomington, IN, USA*

Contents

1. Problem and Data Description

Face Digit recognition and classification play sin important role in today's life. It has become the medium of communication to connect with people and to present someone's idea in the form of text. So, identification has become a necessary in all kinds of areas. The diversity often makes it difficult to recognize and read/identify. The necessity of recognition of images and digits is necessary and thus several deep learning algorithms were implemented. They provide better results than conventional methods and these algorithms require less time to recognize especially when we have large data-sets. For this project we have taken MNIST data-set , the reason that it is chosen is that, it has all the data pre-processed and when we apply the models then we could trigger the exact output for classification.

DATA DESCRIPTION:

• The dataset contains grayscale images of faces, each with a size of 48x48 pixels.
• The faces in the images are centered and occupy similar amounts of space.
• The goal is to classify each image into one of seven emotion categories: Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral.
• The dataset has two files: train.csv and test.csv.
• The train.csv file contains two columns: "emotion" and "pixels".
• The "emotion" column has a number between 0 and 6 indicating the emotion in the image.
• The "pixels" column has a string with space-separated pixel values in row-major order.
• The test.csv file has only the "pixels" column, and the task is to predict the emotion for each image.
• The training set has 28,709 examples, while the public test

set has 3,589 examples.
• The final test set used for determining the winner of the competition also has 3,589 examples.

2. Data Preprocessing & Exploratory Data Analysis

The following steps are followed on face image data-set for Data Pre-processing:
1. Load the data: In this code, we use **pd.read-csv** function for faceimages.csv data set.

```
df = pd.read_csv('faceimages.csv')
print(df.shape)
df.head()
```

2.Data cleaning: We ahve taken th edata from kaggle and used some cleaning and pre-processing, data cleaning steps that can be taken to improve the quality of the data. Some of these steps include:
1. Checking missing values: Check for any missing or corrupted data in the data set and either remove the corresponding examples or impute the missing values with appropriate replacements.

```
df.emotion.value_counts()
```

2. Removing duplicates: Check for any duplicate entries in the data-set and remove them to ensure that each image and label is unique.

```
#Data Cleaning:

df.emotion.unique()

df.emotion.value_counts()
```

3. Algorithm and Methodology

Creating a Model: Creating a model on the given data set involves building a machine learning or deep learning model that can accurately classify the images in the data set.
This is a description of a Deep Convolutional Neural Network (DCNN) used in a particular case. The network includes dropout layers at regular intervals to aid generalization. The activation function used is ELU, which avoids the dying ReLU problem and has been found to perform well in this case compared to LeakyReLU. The henormal kernel initializer is used, as it is suitable for use with ELU. Batch normalization is also used to improve the results.

```
#developing DCNN:

def build_net(optim):
    net = Sequential(name='DCNN')

    net.add(
        Conv2D(
            filters=64,
            kernel_size=(5,5),
            input_shape=(img_width, img_height, img_depth),
            activation='elu',
            padding='same',
            kernel_initializer='he_normal',
            name='conv2d_1'
        )
    )
    net.add(BatchNormalization(name='batchnorm_1'))
    net.add(
        Conv2D(
            filters=64,
            kernel_size=(5,5),
            activation='elu',
            padding='same',
            kernel_initializer='he_normal',
            name='conv2d_2'
        )
    )
```

4. Experiments and Results

Loading the data:

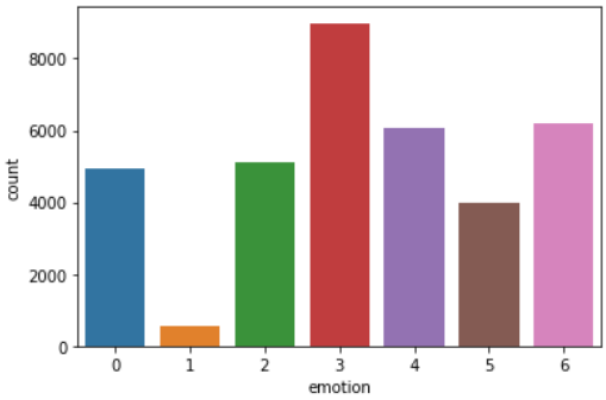
(35887, 3)

	emotion	pixels	Usage
0	0	70 80 82 72 58 58 60 63 54 58 60 48 89 115 121...	Training
1	0	151 150 147 155 148 133 111 140 170 174 182 15...	Training
2	2	231 212 156 164 174 138 161 173 182 200 106 38...	Training
3	4	24 32 36 30 32 23 19 20 30 41 21 22 32 34 21 1...	Training
4	6	4 0 0 0 0 0 0 0 0 0 0 0 0 3 15 23 28 48 50 58 84...	Training

Creating labels:
The sns.countplot() function plots the count of each unique

value in the specified column, in this case, the emotion labels. The resulting plot shows the number of images in the dataset that correspond to each emotion.

The pyplot.show() function displays the resulting plot. This code is useful to get an idea of the distribution of the different emotions in the dataset and can be used to check if the dataset is balanced or imbalanced with respect to the different classes.



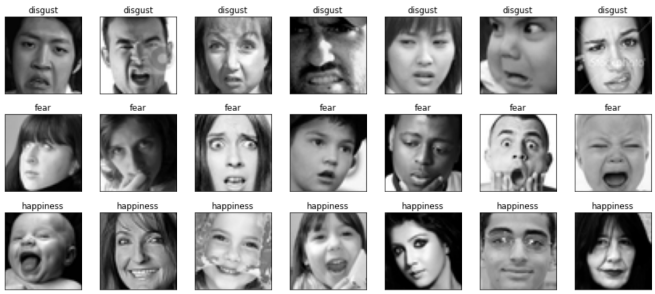
Grid creation-emotion labels:

The fig = pyplot.figure(1, (14, 14)) line creates a figure object with a size of 14x14 inches, which will be used to display the grid of images.

The nested for-loops iterate over each unique emotion label in the dataset and display an image for each label in its corresponding row in the grid.

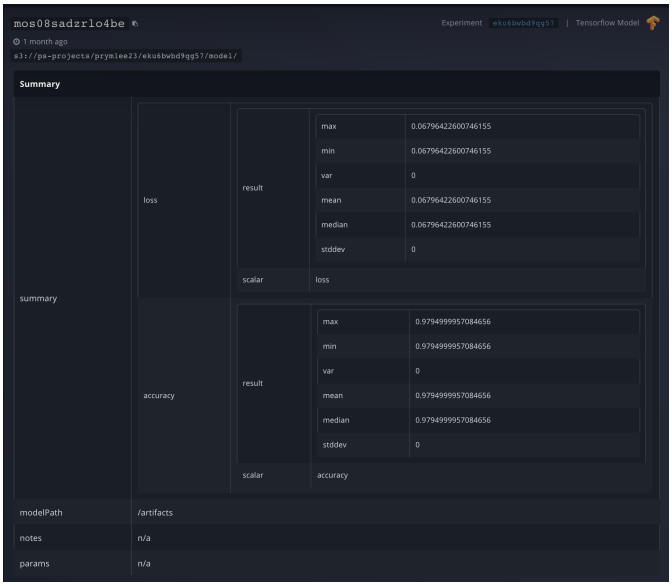
The ax.imshow() function displays the image in grayscale using the cmap='gray' parameter. The ax.set-xticks([]) and ax.set-yticks([]) lines remove the tick marks on the x and y axes. The ax.set-title() function sets the title of each cell in the grid to the corresponding emotion label.

The pyplot.tight-layout() function adjusts the spacing between subplots to improve the overall layout of the grid of images. This code is useful to visualize a sample of the images in the dataset and get an idea of what the images look like for each emotion class.



5. Deployment and Maintenance

We used **https://docs.paperspace.com/gradient/models/** and we are working on the deployment on free platform



6. Summary and Conclusions

From above project description we have 2 domains of classification as of : **Digit classification Face classification.**. In this **Check-in phase-II** we have implemented the model (DCNN) for face classification.The major tasks that we accomplished in this phase are:

1. Data Preprocessing
2. Data cleaning
3. Check Data Structure
4. Check sample instance
5. Create a model
6. Compile the model
7. Train the model
8. Predict the model

Tasks that should be accomplished in further requirements:

Deploy the model and check the same for different data sets to ensure the accuracy for the model that we have developed and we need to work from the initial model creation if it fails to work on data-sets.

Acknowledgments

We would like to thank Professor and all the Teaching Assistants who gave us idea on how to proceed with the outline of the project and detailing on working sources.

References

1.https://www.researchgate.net/publication/
2.https://arxiv.org/ftp/arxiv/papers/1505/1505.04058.pdf