Demographics Face Recognition using CNN

Applications of Artificial Intelligence, Machine Learning and Data Science
(CSC-40070)

1 Introduction

Over the last decade, image uploads to the Internet have increased at a near-exponential rate. These technologies can be used for everything from advising someone to "tag" in Facebook images to detecting pedestrians in self-driving cars. However, building on this work, the next significant step is to ask not only how many faces are in a picture and where they are, but also what qualities those faces have. The goal of this project is to classify facial demographic features such as age, gender, and ethnicity in an image.

The possibilities for this project are endless. We can assist law enforcement agencies by determining age and gender from an image. Automatic age and gender detection can quickly identify potential suspects based on their age and gender. Under the underage acts, it can also prevent children from acquiring forbidden items. It helps in Human-Computer Interaction because the system can only present content that is appropriate for the person's age. Age and gender detection systems have a variety of other uses, including criminal and terrorist investigations, social security, border control, passport control, medical records, and more [1].

Detecting age and gender from an image is a more difficult challenge than many other computers vision tasks. We must train our model with sample data to predict anything. The main problem arises from the nature of the sample data required to train these systems. We do have access to millions of photos for object classification tasks, but the data required for supervised learning should be labelled data. In this case, the images should be labelled with the people's ages, genders, and ethnicity. Because there are so few of them compared to labelled data, finding this type of data is tough. The fundamental issue with data labelling is that we do not have access to some personal information about people, such as their date of birth for example. As a result, we must accept the limitations of the challenge we are tackling and customize network architectures and algorithmic approaches to work around them. These are the key motivations for employing convolutional neural networks to create a basic architecture for age, gender, and race categorization.

2 Related Works

The classification of age and gender has been explored for decades. Traditional Machine Learning methods like as Support Vector Machine and Random Forest Algorithm could be used to detect age, gender, and race from an image, and have had varying degrees of success.

1. **Support Vector Classifier**

   Support Vector Machines (SVM) are commonly thought of as a classification strategy, however they may be used to solve both classification and regression problems. It can handle both continuous and categorical variables with ease. To differentiate various classes, SVM creates a hyperplane in multidimensional space. SVM iteratively generates the best hyperplane, which is then utilized to minimize an error. The goal of SVM is to find a maximum marginal hyperplane (MMH) that splits a dataset into classes as evenly as possible. SVM can be utilized in image classification and object detection too. [2].

2. **Random Forest Algorithm**

   Random Forest is a well-known machine learning algorithm that uses the supervised learning method. It can be utilized to solve both classification and regression issues. It is based on ensemble learning, which is a method of integrating several classifiers to solve a complex problem and thereby improves the model's performance. Rather than depending on a single decision tree, the random forest takes each tree's prediction and predicts the final output based on the majority votes of predictions. So greater the number of trees in the forest, the higher the accuracy and the lower the risk of overfitting issues [3].

To feed the filtered images to traditional machine Learning classifiers, we must convert them to scalars format. Based on the results, we can conclude that these models did not produce models with extremely high accuracy scores. Moreover, key features from images needed to extract for using it in classification models and this approach necessitates a significant amount of domain knowledge and image data processing expertise.

The deep learning and neural networks approach, on the other hand, does not necessitate extensive domain knowledge or image processing expertise because no image features must be extracted manually. This method also produced significantly better performing models with higher accuracy scores. CNNs can perform faster and produce the best detection results even if the image is slightly warped, stretched, or altered with certain lighting conditions.

3 Demographics Face Recognition using CNN

In this proposed system, we will develop a deep learning model using Convolutional neural networks. Deep learning techniques have recently proven to be a tremendous success in the Computer Vision discipline. Deep learning enables multi-layered computing models to determine and interpret data at multiple abstraction levels, mimicking how the brain perceives and interprets information. A convolutional neural network (CNN) is a type of deep neural network that is commonly used to analyze visual imagery in deep learning. CNN consists of the following layers (Figure 1).

- **Convolutional Layer:** Since CNN employs a variety of kernels, the convolutional operation of the layer increased the learning time of the developed model [4].
- **Pooling Layers:** It reduces the spatial dimensions of the next convolutional layer's input volume. It only influences the length and height, not the depth [5].
- **Fully connected Layers:** To perform any high-level reasoning, several connected neural network layers should be used [4].

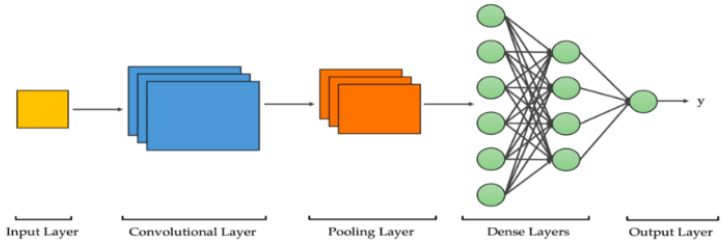

Figure 1: Architecture of CNN.

4 Dataset

We use UTKFace Kaggle dataset which contains over 20 thousand images of human beings which include different ethnicity, color, and many more factors. It is a large-scale face dataset with a long age span (ranging from 0 to 116 years old). The images cover large variation in pose, facial expression, illumination, occlusion, resolution, etc. This dataset serves as a baseline for face pictures and includes noise, lighting, posing, and looks, among other real-world imaging scenarios. The labels of each face image are embedded in the file name, formatted like [age]_[gender]_[race]_[date&time].jpg. For gender, 0 denotes male and 1 denotes female. For race, 0 denotes white, 1 denotes black, 2 denotes Asian, 3 denotes Indian, and 4 denotes other races respectively [8].

5 Preprocessing Data

Our Dataset consists of raw images of various faces labelled with demographic details. Initially we need to transform this data to generate a dataframe which aids exploratory data analysis. So, we wrote a function to iterate over each file of the UTKFace dataset and returns required demographic attributes. We found 3 files in the dataset have invalid labels and these results in null values in the dataframe. So, we eliminated null values using pandas dropna(). The final output of the dataframe is depicted in Figure 2.

```
In [146]: df.head()
```

Out[146]:

	age	gender	race	file
0	9.0	female	asian	UTKFace/9_1_2_20161219204347420.jpg.chip.jpg
1	36.0	male	black	UTKFace/36_0_1_20170117163203851.jpg.chip.jpg
2	86.0	female	white	UTKFace/86_1_0_20170120225751953.jpg.chip.jpg
3	26.0	female	white	UTKFace/26_1_0_20170116171048641.jpg.chip.jpg
4	1.0	female	asian	UTKFace/1_1_2_20161219154612988.jpg.chip.jpg

Figure 2: Dataframe details.

We did some exploratory data analysis and could see that most of the population is between the age group 20 to 30 Also from the dataframe distribution, we could see that, we got less data for age above 60, so we can sample dataframe under the age 60 for better model creation (Figure 3).

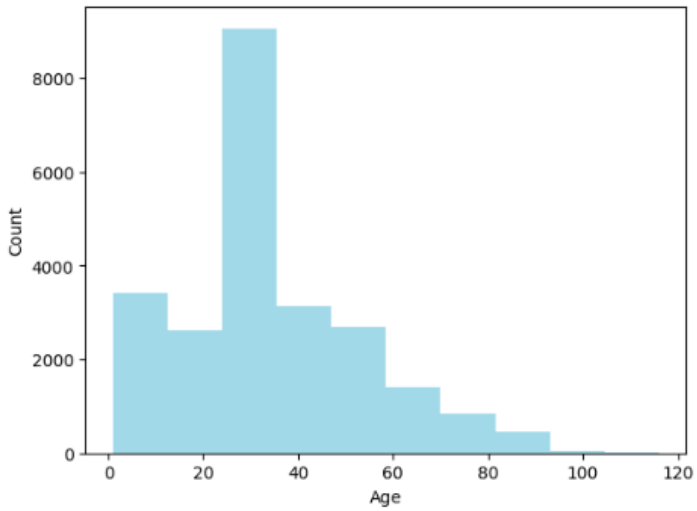

Figure 3: Age distribution on the Dataset.

Now we will divide the dataframe for training and testing CNN model. We are using 80% of the data for training and the remaining 20% for testing purposes. We split a portion of training data to get hold of validation set (Figure 4).Since we are using categorical data for gender and race columns in the dataframe, we need to encode it to numerical format before feeding it into the model. We use One hot encoding to convert categorical data variables so that they can be fed into machine and deep learning algorithms, which improves prediction and classification accuracy of a model. We created batches of data, which will be utilized to feed both the photos and their labels into the Kera's multi-output model rather than loading the entire dataset into memory at once, which could result in an out of memory error.

```
Training Data: 13450
Validation Data: 3363
Testing Data: 4204
```

Figure 4: Training Test Data.

Before we feed the image data to the CNN model, we need to re-scale the size and should undergo normalization process. In an image each pixel values can range from 0 to 256. Each pixel in an image can have a value ranging from 0 to 256. Each number represents a distinct color code. The computation of high numeric values may become more complex when using the image in a Deep Neural Network. We can reduce this by normalizing the values to a range of 0 to 1. To convert the image into an array format we use NumPy's array module 'np.array'.

6 Feature extraction

The main advantage of CNN over its alternative approaches is that it automatically detects prominent features without the need for human intervention. A CNN model is made up of two parts: the feature extraction part and the classification part. The feature extraction is done by the convolution and pooling layers. Given an image, the convolution layer detects features such as a moustache, beard, long or short hair, and so on. On top of these features, the fully connected layers act as a classifier, assigning a probability to whether the input image is male or female.

A CNN model's main powerhouse is its convolution layers. It is difficult to detect meaningful features automatically given only an image and a label. Convolution layers learn such complex features by stacking them on top of one another. The first layers detect edges, the subsequent layers combine them to detect shapes, and the final layers combine this information to identify the image. The fully connected layers learn how to use the convolutional features to correctly classify the images.

7 Implementation

7.1 Network architecture

Our proposed network architecture is used throughout our experiments to classify people based on Demographics features. Figure 5 shows an architecture model for our CNN. The network comprises of six convolutional layers and 3 fully connected layers representing the age, gender, and race attributes.

As an initial step, we will pass the preprocessed images to the network. The six subsequent convolutional layers configured with multiples of 32, followed by an activation function, rectified linear operator (ReLU) and a max pooling layer. The activation function in a neural network is responsible of converting the node's summed weighted input into the node's activation or output for that input. [6].We are doing batch normalization in each convolution layer. Batch Norm is a normalization technique that is performed between the layers of a Neural Network rather than on the raw data. It is done in mini batches rather than the entire data set. It facilitates learning by speeding up training and utilizing higher learning rates. [7]. Following that, a fully connected layer with 128 neurons receives the output of the six convolutional layers, followed by a ReLU and a bottleneck layer. A bottleneck layer is one with fewer nodes than the preceding layers. It can

be used to obtain a reduced-dimensionality representation of the input. The output of our fully connected layer is then fed into a SoftMax and sigmoid layer, which assigns a probability to each class. The prediction is made by selecting the class with the greatest probability for the given test image. Figure 6 illustrates the model summary from the code.

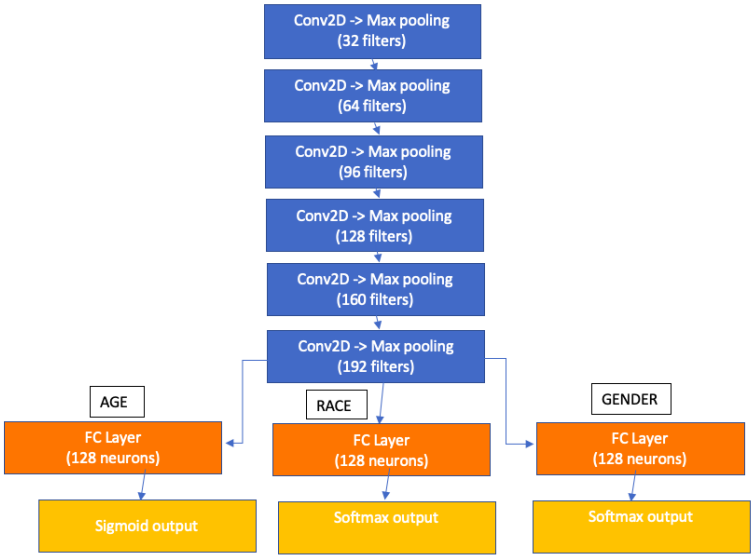

Figure 5: CNN Network Architecture

7.2 Training and testing

We have already split our filtered dataset to training, validation, and test subsets with ratios 80%, 10% and 10% respectively. We trained the model from scratch by using our own tuned network architecture and configured other training parameters like the number of epochs to train as 10 and batch size as 64. We trained the model using various optimizers and starting learning rates. We use standard gradient descent and RMSprop as optimizers. The most significant disadvantage of standard gradient descent (SGD) is that it is not adaptive. The selection of the appropriate learning rate is critical, and the same learning rate applies to all parameters. RMSprop, or Root Mean Square Propagation, attempts to address gradient descent issues by dividing the learning rate by an exponentially weighted average of squared gradient. Since RMSprop has overcome the limitation of SGD, we decided to use RMSprop optimizer. Figure 6 illustrates the training history. In each epoch, we could see the model training progress.

7.3 Results and Discussion

From Figure 7 on training history, we can see that the accuracy of training and validation data of gender and race parameters improves over time. For age attribute we could see Mean Absolute Error is decreasing over time. Mean Absolute Error is the average of the absolute values of each prediction error across all instances of the test dataset. The difference between the actual and predicted values for that instance is referred to as prediction error. Also, we could notice that the validation loss decreases but then begins to rise again. This typically indicates that the model is starting to overfit and is unable to generalize to new data.

Figure 8 displays the classification report of our model. It is a machine learning performance evaluation metric that displays the precision, recall, F1 Score, and support score of our trained classification model. Here, we can see that the race attribute had an accuracy of 75%, while the gender attribute had an accuracy of 86%.

Figure 9 provides confusion matrix for our gender and race classification results. A confusion matrix is a summary of classification problem prediction results. The number of correct and incorrect predictions is summarized with count values and divided by class. The diagonal elements represent the number of points for which the predicted label equals the true label,

```
Model: "model_1"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_2 (InputLayer)           [(None, 198, 198, 3  0           []
                                )]

 conv2d_6 (Conv2D)              (None, 196, 196, 32  896         ['input_2[0][0]']
                                )

 conv2d_7 (Conv2D)              (None, 194, 194, 64  18496       ['conv2d_6[0][0]']
                                )

 batch_normalization_5 (BatchNo  (None, 194, 194, 64  256        ['conv2d_7[0][0]']
 rmalization)                   )

 max_pooling2d_5 (MaxPooling2D)  (None, 97, 97, 64)  0           ['batch_normalization_5[0][0]']

 conv2d_8 (Conv2D)              (None, 95, 95, 96)   55392       ['max_pooling2d_5[0][0]']

 batch_normalization_6 (BatchNo  (None, 95, 95, 96)  384         ['conv2d_8[0][0]']
 rmalization)

 max_pooling2d_6 (MaxPooling2D)  (None, 47, 47, 96)  0           ['batch_normalization_6[0][0]']

 conv2d_9 (Conv2D)              (None, 45, 45, 128)  110720      ['max_pooling2d_6[0][0]']

 batch_normalization_7 (BatchNo  (None, 45, 45, 128)  512        ['conv2d_9[0][0]']
 rmalization)

 max_pooling2d_7 (MaxPooling2D)  (None, 22, 22, 128)  0         ['batch_normalization_7[0][0]']

 conv2d_10 (Conv2D)             (None, 20, 20, 160)  184480      ['max_pooling2d_7[0][0]']

 batch_normalization_8 (BatchNo  (None, 20, 20, 160)  640        ['conv2d_10[0][0]']
 rmalization)

 max_pooling2d_8 (MaxPooling2D)  (None, 10, 10, 160)  0         ['batch_normalization_8[0][0]']

 conv2d_11 (Conv2D)             (None, 8, 8, 192)    276672      ['max_pooling2d_8[0][0]']

 batch_normalization_9 (BatchNo  (None, 8, 8, 192)   768         ['conv2d_11[0][0]']
 rmalization)

 max_pooling2d_9 (MaxPooling2D)  (None, 4, 4, 192)   0           ['batch_normalization_9[0][0]']

 global_max_pooling2d_1 (Global  (None, 192)         0           ['max_pooling2d_9[0][0]']
 MaxPooling2D)

 dense_3 (Dense)                (None, 128)          24704       ['global_max_pooling2d_1[0][0]']

 dense_4 (Dense)                (None, 128)          24704       ['global_max_pooling2d_1[0][0]']

 dense_5 (Dense)                (None, 128)          24704       ['global_max_pooling2d_1[0][0]']

 age_output (Dense)             (None, 1)            129         ['dense_3[0][0]']

 race_output (Dense)            (None, 5)            645         ['dense_4[0][0]']

 gender_output (Dense)          (None, 2)            258         ['dense_5[0][0]']

==================================================================================================
Total params: 724,360
Trainable params: 723,080
Non-trainable params: 1,280
```

Figure 6: Model Summary.

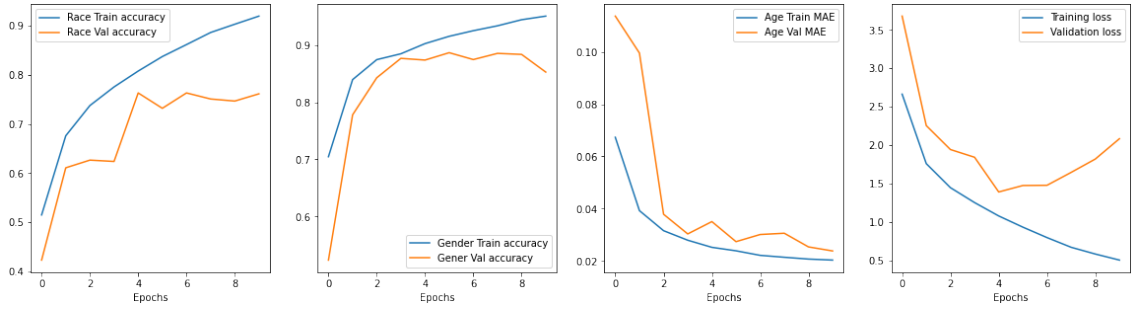

Figure 7: Plot of Model Accuracy and Loss on Train and Validation Datasets.

Classification report for race

	precision	recall	f1-score	support
0	0.82	0.79	0.80	57
1	0.71	0.75	0.73	20
2	0.67	0.94	0.78	17
3	0.73	0.73	0.73	22
4	0.67	0.33	0.44	12
accuracy			0.75	128
macro avg	0.72	0.71	0.70	128
weighted avg	0.75	0.75	0.74	128

Classification report for gender

	precision	recall	f1-score	support
0	0.96	0.73	0.83	59
1	0.81	0.97	0.88	69
accuracy			0.86	128
macro avg	0.88	0.85	0.85	128
weighted avg	0.88	0.86	0.86	128

Figure 8: Classification report.

whereas off-diagonal elements are those for which the classifier mislabeled. The higher the diagonal values of the confusion matrix, the more correct predictions there are. Here we could see we got higher counts in diagonal elements suggesting our model is a better prediction model.

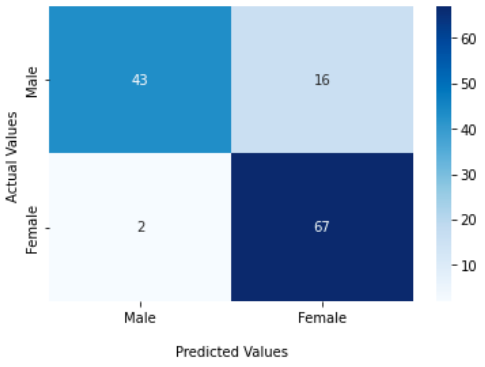

Figure 9: Confusion Matrix on Gender Attribute.

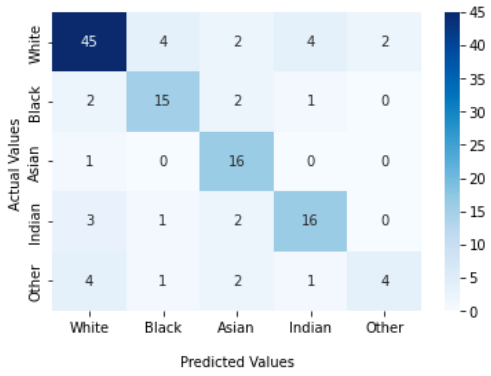

Figure 10: Confusion Matrix on Race Attribute.

Using the test data, we predicted demographic features successfully. The final output of the model is depicted in Figure 11. Predicted details are placed on the top of the images whereas actual details are placed on the bottom of the images. From the output we could see, gender prediction has more accuracy than the other 2 attributes.

The flowchart of the entire implementation of the system is represented in Figure 10.

8 Methods for Improving Performance of CNN Model

We could improve the performance of CNN Model by the following techniques

- **Tune Parameters:** We can tune parameters such as epochs, learning rate, and so on to improve CNN model performance. The number of epochs has a significant impact on performance. There has been an improvement in performance over many epochs. However, some experimentation is required to determine epochs and learning rates. In the

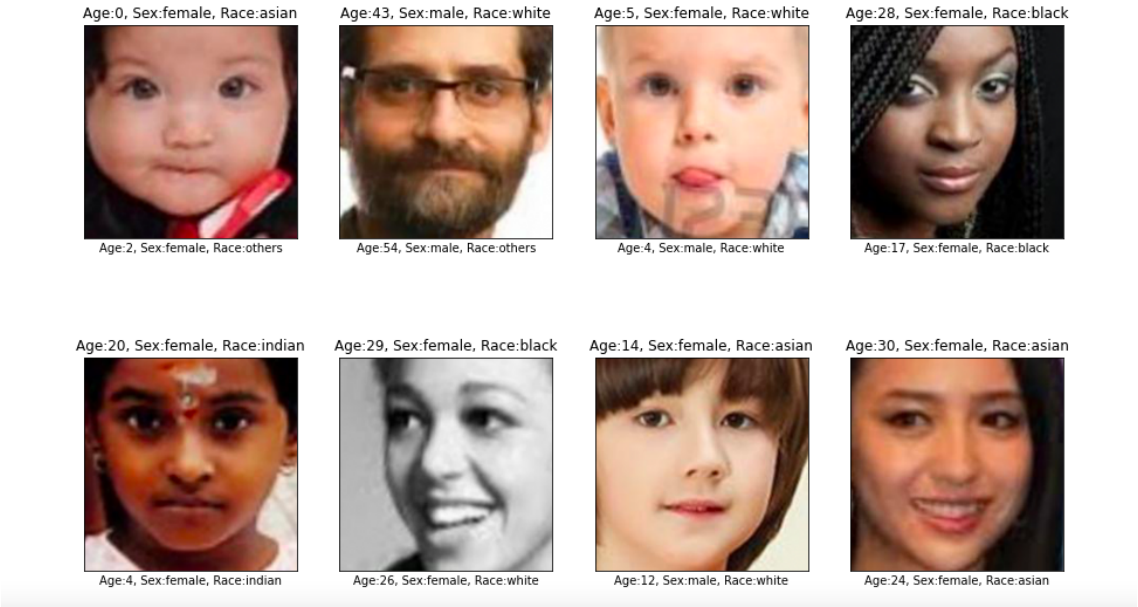

Figure 11: Final Output of the model.

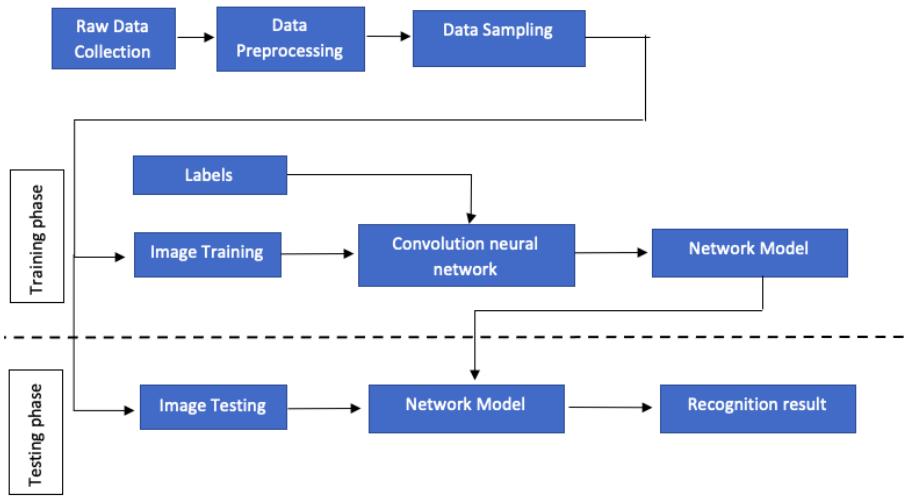

Figure 12: Flowchart of the system.

CNN model, we can also use a dropout layer. Depending on the application, the proper optimizer must be selected during model compilation. We can use a variety of optimizers, such as SGD, RMSprop, Adam, and others. There is a need to fine-tune the model using various optimizers. All these factors have an impact on CNN's performance.

- **Image Data Augmentation:** Deep learning is only useful when there is a large amount of data. CNN requires the ability to learn features automatically from data, which is typically only possible when there is a large amount of training data available. If we have a limited amount of training data, we can use Image Augmentation. Zoom, shear, rotation, preprocessing function, and other image augmentation parameters are commonly used to increase the data sample count.
- **Deeper Network Topology:** It is possible to train a large neural network with any input value. As a result, these networks excel at memorization but struggle with generalization. Deeper networks capture the natural "hierarchy" that exists in all of nature. The benefit of using multiple layers is that they can learn features at various levels of abstraction. Therefor a deep network is preferable to a very wide but shallow network.
- **Properly Handling Overfitting and Underfitting problems**

9 Conclusion and Future work

We could see many methods have solved the problems that were raised when detecting age, gender, and race, most of these methods have focused on images that had constraints and limitations. Even though we need an enormous collection of labelled data for supervised learning, its lack of availability is a complicated issue. In this project, we have used UKTFace Kaggle dataset, which consist of 20 thousand labelled images for training our deep CNN network. Even though there are fewer labelled images available for demographic details, CNN can be used to provide better age and gender detection results. Furthermore, the performance of this system can be marginally improved by using more training data and running it on a more computational system.

Running this complex model was bit challenging, since it consumes more time on training the data. I sampled dataset to fasten the training process and could observe the results were not accurate and later spend hours of training with whole data to get more accurate results. If I had more time, I would like to spend more time fine-tuning the parameters and the modified architectures I experimented with. I would have preferred to have the Adam learning algorithm in place with performance comparable to or better than SGD or RMSprop. I anticipate that future scope work in this area will include the use of face demographics, human expression classification to aid in face recognition, facial disease detection, study of human psychology of emotions and improve experiences with photos on social media, and much more.

References

[1] A. Mustafa and K. Meehan, "Gender Classification and Age Prediction using CNN and ResNet in Real-Time," Oct. 2020. Accessed: Apr. 02, 2022. [Online]. Available: http://dx.doi.org/10.1109/icdabi51230.2020.9325696

[2] "What is SVM," Analytics Vidhya, Jun. 18, 2021. https://www.analyticsvidhya.com/blog/2021/06/build-an-image-classifier-with-svm/ (accessed Apr. 04, 2022).

[3] "Machine Learning Random Forest Algorithm - Javatpoint," www.javatpoint.com. https://www.javatpoint.com/machine-learning-random-forest-algorithm (accessed Apr. 04, 2022).

[4] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep Learning for Computer Vision: A Brief Review," Computational Intelligence and Neuroscience, vol. 2018, pp. 1–13, 2018, doi: 10.1155/2018/7068349.

[5] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A Theoretical Analysis of Feature Pooling in Visual Recognition," p. 8, 2010.

[6] "A Gentle Introduction to the Rectified Linear Unit (ReLU)," Machine Learning Mastery, Jan. 08, 2019. https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/ (accessed Apr. 14, 2022).

[7] https:/www.baeldung.comcsauthormartin7557#author, "Batch Normalization in Convolutional Neural Networks," Baeldung on Computer Science, Oct. 29, 2020. https://www.baeldung.com/cs/batch-normalization-cnn (accessed Apr. 14, 2022).

[8] S. Subedi, "UTKFace," Kaggle. https://www.kaggle.com/datasets/jangedoo/utkface-new (accessed Apr. 01, 2022)