What is my occupation?

by

Darshankumar Zalavadiya
202011061

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of

MASTER OF TECHNOLOGY
in
INFORMATION AND COMMUNICATION TECHNOLOGY
to

DHIRUBHAI AMBANI INSTITUTE OF INFORMATION AND COMMUNICATION TECHNOLOGY

June, 2022

Declaration

I hereby declare that

i) the thesis comprises of my original work towards the degree of Master of Technology in Information and Communication Technology at Dhirubhai Ambani Institute of Information and Communication Technology and has not been submitted elsewhere for a degree,

ii) due acknowledgment has been made in the text to all the reference material used.

[illegible]

Darshankumar Zalavadiya

Certificate

This is to certify that the thesis work entitled "WHAT IS MY OCCUPATION?" has been carried out by **DARSHANKUMAR ZALAVADIYA (202011061)** for the degree of Master of Technology in Information and Communication Technology at *Dhirubhai Ambani Institute of Information and Communication Technology* under my/our supervision.

[illegible]

Prof. Manish K. Gupta
Thesis Supervisor

Acknowledgments

I want to express my gratitude to my thesis supervisor Dr. Manish K. Gupta for for providing me with the oppurtunity to work under his supervision for this M.Tech thesis. He gives me continuous support and guidance throughout the journey. I am greatly honoured to work with him. His cheerful attitude, eagerness to help, and domain expertise have all aided me in bringing out the best in myself. It would have been so difficult to carry out this work without his supervision.

I would like to thank my friend Kashyap who helped me in various phases of the thesis work. I also would like to thank my batchmates Kripal, Mahir, Aakash and all other friends, who helped me to gather the data for the research work.

I want to thank all the responders for filling the data through the Google Form. The sample photographs shown in this thesis are used with the consent of the responder. Thus I would like to thank these responders for their valuable input.

Lastly I would like to thank my parents, who always believed in me and supported me. Thank you for giving your blessings and moral support.

Darshankumar Zalavadiya

Contents

Abstract

The facial image of a person contains much rich information like gender, race, age, hairstyle, and other essential pieces of information that are related to a person's occupation. Predicting or recognizing occupation is a classification problem. Because of its wide reach in intelligent systems and services, this is a feasible computer vision challenge.

In this thesis, we have stated a related work that includes the different studies on this topic. The previous study was done based on recognizing occupation from human clothing, scene context, social context, and a facial image of a person. My thesis objective is to predict occupation from a facial image of a human.

We have collected data on Indian people and made a new dataset. In this work, we have used two different datasets. The first dataset, DB1, is used in previous studies as well, which is based on eastern Asian people. The second dataset is based on Indian people, which is made by us. Both the dataset contains five different classes. We have used multiple image classification algorithms and compared them in terms of accuracy and performance. Out of all of these algorithms that we have used, The vision transformer performs best as compared to other algorithms. Also, the vision transformer achieves better accuracy as compared to all other previous works, which is based on occupation prediction from the facial image.

Keywords : Occupation prediction, image classification

List of Tables

List of Figures

CHAPTER 1
Introduction

In this chapter, we have briefly discussed the introduction of the entire research work. This chapter describes the objective of the thesis and the motivation behind the thesis. We have also mentioned contribution and thesis organization.

1.1 Thesis Objective

The main goal of my work is to predict occupation from the facial image of a person. It is a classification problem in which we classify the facial image based on occupation. In this work, we are using the supervised learning method to train a model, in which we train a model with a facial image that is associated with a label. After the training, a model takes a new test image, and based on previous data; our model predicts the occupation.

In this thesis work, we have implemented various image classification algorithms and did a comparative analysis. We have also gathered the data of Indian people and made an Indian dataset that contains facial images and occupations associated with it. Thus we have drawn our conclusions based on various experiments on different algorithms.

1.2 Thesis Motivation

The research based on retrieving face attributes is widely popular. From the face of the person, we can derive very important characteristics like age, gender, ethnicity, personality, etc. This face information is beneficial in intelligent systems and services. These face attributes are very useful in recommendation and security. For example, face recognition is useful in biometrics and security. These attributes are also used in surveillance. Age, ethnicity, gender, personality, etc. Data is essential for a variety of real-world applications, such as, identity verification, video surveillance, human-computer interaction, biometrics, electronic customers, online advertising, crowd analysis, and item suggestion, among others.

Our research involves determining a person's occupation based on a single face image. Because of its immense potential in intelligent systems, occupation prediction from pictures is a significant topic in the computer vision field.

The previous studies on occupation prediction is mainly based on human clothing, social contexts, and scene contexts. In those studies, they predict occupation based on which dress people wear, background image of a person, social interaction, or where people work. In this work, our approach is to predict the occupation from the single facial image of a person. Although predicting employment only based on facial features may appear counterintuitive at first, but it is a realistic goal that may complement existing clothing and context-based approaches.

Examples of facial images are shown in figure 3.1 of professors, athletes & anchorpersons. From those images, we can easily observe that athletes be younger persons, anchorpersons primarily female and young, and professors tend to be older.

There are many other facial features like glasses-wearing, skin color, hairstyle, gender, age, etc. can be a facial features for predicting an occupation from facial images. We come to the conclusion that a profession may be thought of as a combined distribution of a collection of facial characteristics, and that a computational model to predict occupation from facial pictures can be developed.

1.3 Contributions

The thesis contributions include the following:

- In this work, we have gathered the data and made a new dataset that is based on the Indian people. Because in previous studies, they have used a dataset that contains eastern Asian people's facial images.
- We have implemented multiple image classification algorithms on two different datasets and did comparative studies based on the result that we got from the experiments.
- We have also made a demo website based on the flask to showcase the occupation prediction from the facial image.

1.4 Thesis Organization

The rest of this thesis work includes the following :

The second chapter is about related work, which includes all previous studies that are related to occupation prediction. The third chapter is regarding the dataset which was used in this work. The fourth chapter is about experiments that we have conducted for this work. The fifth chapter contains the results of our experiments and analysis. The sixth contains possible applications and challenges of this defination. At last seventh chapter describes the Conclusion and Future Work.

CHAPTER 2
Related Work

In this chapter, we have described various previous studies which are related to occupation prediction. Here we have mentioned an introduction regarding studies, an overview of previous studies, and at the end summary of related work.

2.1 Introduction

There are many studies published that are based on retrieving different face attributes from facial images. These types of studies are widely popular. Our study is also based on retrieving face attribute. We are predicting occupation from the facial appearance of a person.

The occupation prediction is a classification problem. In this classification problem, an image of a person is classified into different classes. The various features are used for predicting the occupation, i.e., Appearance.

2.2 Human clothing and context

The part-based modeling was proposed by Song et al. [13]. This was first work on occupation prediction. They use part-based modeling to predict occupation by looking at the head, upper body and left as well as right shoulders. The head of a person has many features like hat style, hairstyle, gender, age, etc. are related to the profession. In the central upper body, clothing style and uniform is an important feature for occupation prediction. At last, left and right shoulders express many essential details about human dressing, like the design of sleeves, after detecting all four human body parts. In order to generate semantic level representation, low-level characteristics with sparse coding are used. It delivers good accuracy by integrating clothing and context information. The framework for predicting occupation from clothing and context information is shows good performance. The framework for predicting occupation from clothing features and context features from the scene is shown in Fig. 2.1

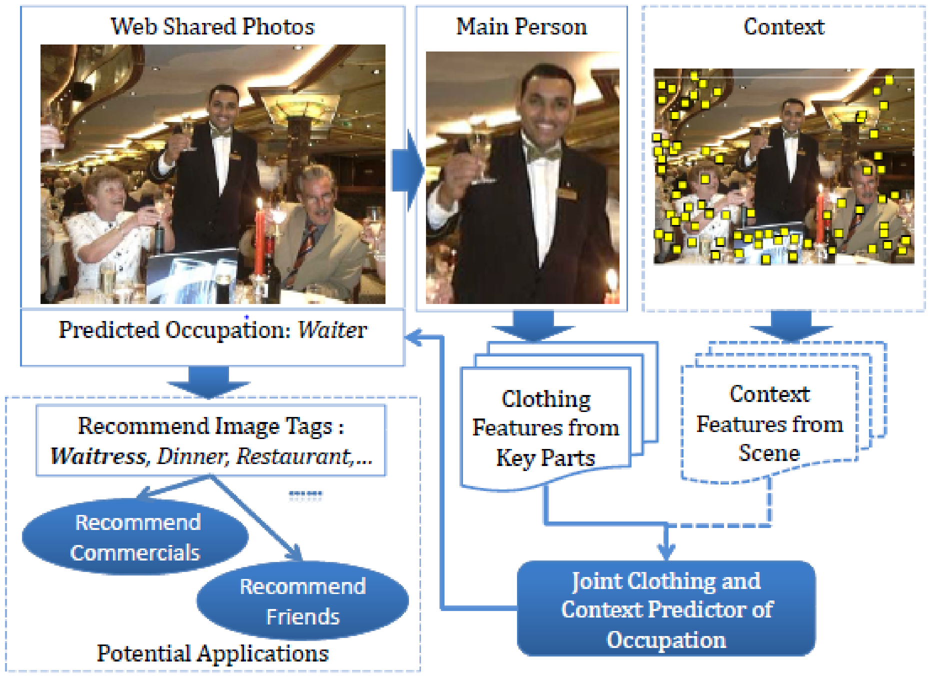

Web Shared Photos
Main Person
Context
Predicted Occupation: *Waiter*
Clothing Features from Key Parts
Context Features from Scene
Recommend Image Tags : *Waitress, Dinner, Restaurant,...*
Recommend Commercials
Recommend Friends
Potential Applications
Joint Clothing and Context Predictor of Occupation

Figure 2.1: Prediction from Human clothing and scene context. [13]. ©[2011] IEEE. with Permission for reprint from, from Z. Song, Meng Wang, Xian-sheng Hua and S. Yan, "Predicting occupation via human clothing and contexts," 2011 International Conference on Computer Vision, 2011, pp. 1084-1091, doi: 10.1109/ICCV.2011.6126355.

2.3 Occupation via social context

M. Shao [12] proposed a method for predicting numerous people's occupations in arbitrary postures while taking into account their social context. In this work, they detect human body parts in random poses and also consider the visual appearance and social context to extract features and train a model by using a structure SVM.

2.4 Occupation prediction by using face & body

Wei-Ta et al. in [4] By incorporating face and body context information, a method for predicting occupation from photos was presented. The framework for this work that contains four different components: first, Data preprocessing, feature extractions, DMC-SVM, and prediction fusion. In preprocessing, the image is divided into two parts, facial part and body part. The top part of the head, the middle upper body, and the left and right arm are the four components divided from the body.

Features will be retrieved from three viewpoints after preprocessing: lower level features, higher level features, and deep features. These extracted attributes are fed into the multi-channel SVM framework is used for classification. To further consider inter-class and intra-class variation, they proposed discriminant multi-channel SVM. The fusion component will combine the outcomes from DMC-SVM based lower level and higher level attributes. And it gives the output by using output of fusion and output of DMC-SVM from viewpoint of deep feature.

2.5 Occupation from a single facial image

Wei-Ta chu in [3] proposed an approach of predicting occupation from a single photograph of a person's face. In this work, they first extracted features from an image and then fed it to the model. For feature extraction, they first used 64 x 64 images, and that parallel with recognized eye which are fixed at a certain position. After that, all photos were subjected to intensity histogram equalisation to eliminate the effects of lighting variations. Then dense SIFT descriptors [11] are extracted, which is 128 dimensional.

These descriptors are transformed into LLC code[14] which is of 1024 dimensional code. For extracting features from multiple scales, the spatial pyramid scheme [10] was used. Total 21 pyramid grids are constructed. From all these pyramid grids, LLC codes were extracted. In the end, a feature vector of 21 x 1024 = 21504 dimensions is extracted for all facial images.

These features are used to predict occupation. To integrate those features, they have used a multi-feature support vector machine framework [2] which is focused on the boosting method was used to build the support vector machine model. Further, they proposed a model of discriminant multi-feature SVM. Those features are given to discriminant multi-feature support vector machine to predict occupation from facial image. This model also considers inter-class and intra-class variation. This proposed solution in this study gives a good performance.

2.6 Summary

From all these previous works, we can say that most of the studies in the past are mostly based on appearance. This means they predict the occupation from human clothing, scene context, and social context. And two paper was published which is based on the predicting occupation from a facial image by only one set of authors. So this is a significantly less explored field in the computer vision domain.

CHAPTER 3
Dataset

In this chapter, we have briefly discussed the dataset. In this work, we have used two datasets. The first one is the DB1 dataset which is based on eastern Asian people, and the second dataset, which is based on the Indian dataset.

3.1 DB1 dataset

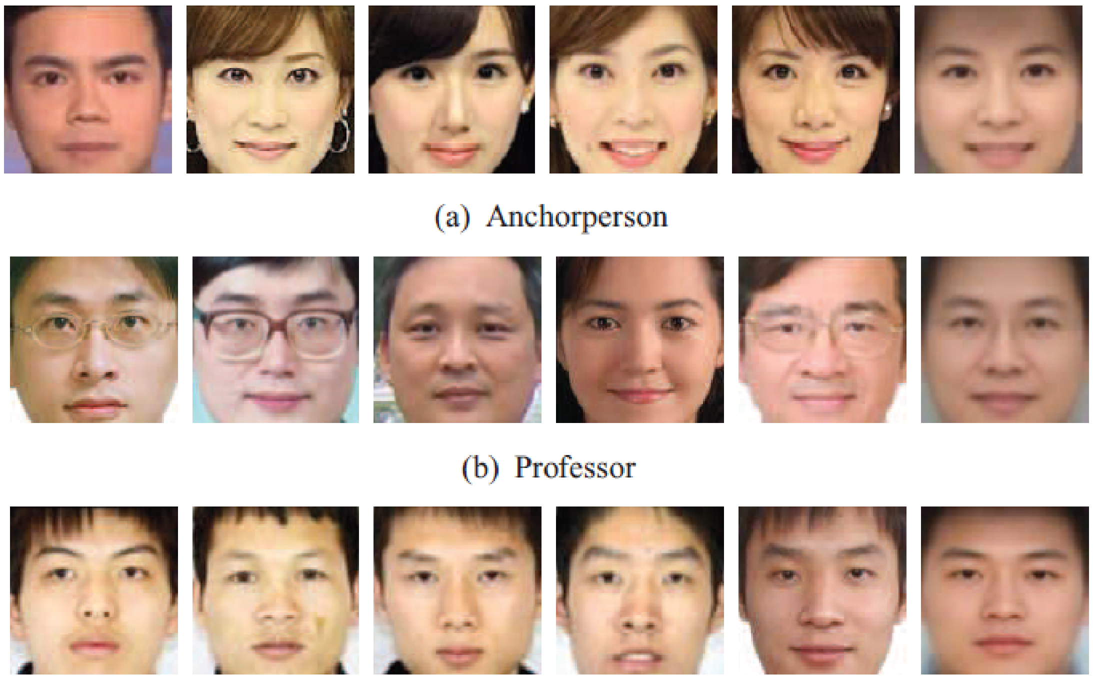


Figure 3.1: Sample image of DB1 dataset and their average faces. [3]. ©[2014] IEEE with permission for reprint from, W. -T. Chu and C. -H. Chiu, "Predicting Occupation from Single Facial Images," 2014 IEEE International Symposium on Multimedia, 2014, pp. 9-12, doi: 10.1109/ISM.2014.13.

For occupation prediction, the dataset was taken from [4]. This dataset was created and used by Wei-ta chu et al. in [3] and [4]. This dataset contains a total of 2062 facial images. These images are distributed among five different classes anchorperson, doctor, professor, athlete, and policeman. All categories have 300 to 500 images. Among those images, some of the images from all classes are chosen randomly for evaluation, where the all other photos for training purpose. In this dataset, all images are of size 64 x 64 pixels, and all pictures are parallel with eyes at certain location that have been recognised. After collecting all images, they extracted images that belonged to only eastern Asian people only.

3.2 Indian dataset

We have also built a dataset for this thesis. Only photos of Indian individuals are included in this dataset. There are 1000 facial photos in this dataset. These photographs are divided into five categories: anchorperson, doctor, farmer, professor, and athlete. There are 200 photographs in each of these classes. We preprocessed the images after gathering them and created an Indian facial image dataset. All of these images are of size 64 x 64 pixels after preprocessing. These photographs were gathered from the internet.

We also have circulated a Google Form to gather the data from our networks, friends, families, and social media. But we have a lesser number of responses than we expected. The sample images shown below are collected from a google form. And also, we have taken the consent of the owner of the image to use them.

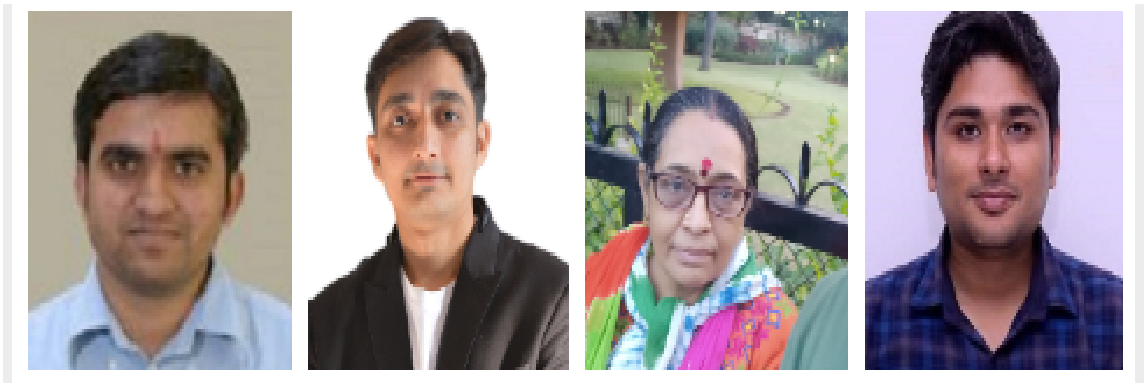


Figure 3.2: Sample image of professors from indian dataset.

CHAPTER 4

Experiments

4.1 Introduction

This chapter is all about the experiments that we have conducted for this work. In this experiment, we implemented various image classification algorithms on two datasets: DB1 and the Indian dataset. We have implemented a support vector machine, logistic regression, AdaBoost, bagging, and vision transformer. Also, we have discussed the feature extraction method that we have used in the first four algorithms to extract the features from facial images.

4.2 Feature Extraction

For extracting the features from the facial image, we have used the Spatial pyramid pooling method[9], which is shown in figure 4.1. The Spatial pyramid pooling method extracts the deep learning features from the multiple scales of the image. In this work, we have extracted features from 3 levels.

For the first level, from the entire image, we extract the features. The image is divided into four different non-overlapping subregions on the second level. And from all four subregions, we extract the features. And at the third level, the entire image is divided into sixteen different non-overlapping sub-regions. And from all these regions, we extract the features. In the end, all features from 21 parts are concatenated. From all parts of the images at multiple levels, we extract a 256-dimensional feature vector. So from all images we extract 21 x 256 = 5376 dimensional feature vector. The features are used as input to the various algorithms to train a model. So this spatial pyramid pooling method extracts robust features from the image.

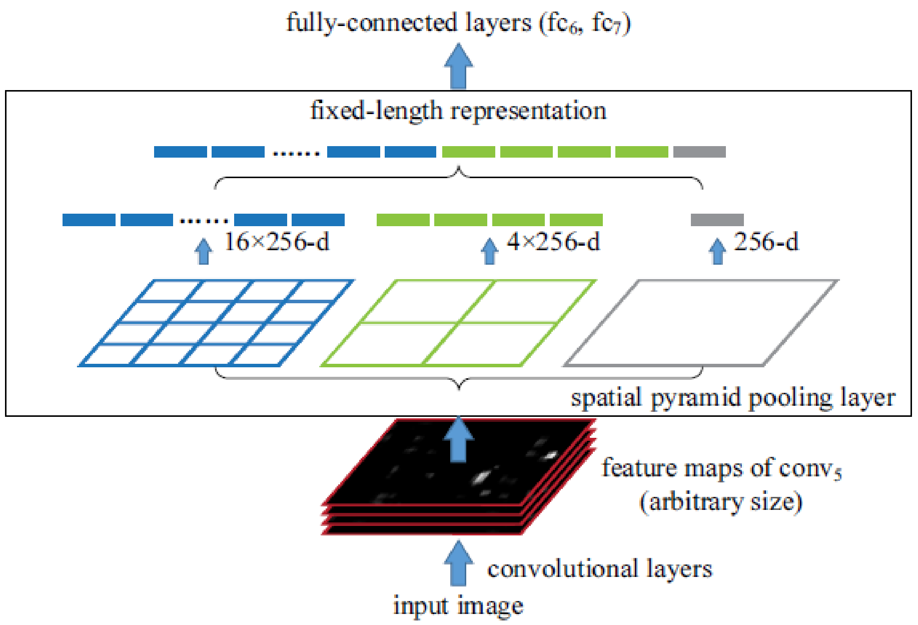

fully-connected layers (fc_6, fc_7)
fixed-length representation
16×256-d
4×256-d
256-d
spatial pyramid pooling layer
feature maps of $conv_5$
(arbitrary size)
convolutional layers
input image

Figure 4.1: Architecture of Spatial pyramid pooling. ©[2015] IEEE. Reprinted, with permission, from K. He, X. Zhang, S. Ren and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 37, no. 9, pp. 1904-1916, 1 Sept. 2015, doi: 10.1109/TPAMI.2015.2389824.

4.3 Support vector machine

The SVM [5] is a discriminative model which maximizes the discrimination among the classes. This algorithm is used for both the regression as well as classification applications. It is a one of the supervised machine learning algorithm. The goal of the SVM technique is to evaluate the best line or decision boundaries for grouping n-dimensional space into categories such that t he following data points can easily be assigned to the appropriate category.

The best decision boundary is referred to as a hyperplane. The extreme points that help construct the hyperplane are chosen using a support vector machine. The technique is known as a Support vector machine, and these support vectors are extreme instances. For multiclass classification, we have used the one vs. rest method in SVM. In this, we split the multiclass dataset into multiple binary classification problems. In SVM, we used features that were extracted from the spatial pyramid and trained the model. This model gives better performance and predicts more accurately as compared to random guesses.

4.4 Logistic regression

The logistic regression [6] is a supervised algorithm. Here we used multiclass regression for classification. Generally, logistic regression is used for binary classification. But for multiclass classification here, we used the one Vs. rest algorithm. In this technique, we select one class and group the remaining classes into a second virtual class and then run binary classification. In this method, we divide two classes by linear line. We have used extracted features from the spatial pyramid as input to the model.

4.5 Ensemble Learning

Multiple weak learners are used and trained in ensemble learning methods to make strong classifiers. In this method, we use base learners or weak learners, which are any classifier, such as a decision tree, a support vector machine, a logistic regression, a naive Bayes, and so on. In this work, we have used a support vector machine as a base learner or weak learner. In this work, we have used two ensemble learning methods: Adaboost and Bagging.

4.5.1 Adaboost

The Adaboost[8] is an ensemble learning technique, which is also known as the meta-learning technique. It is a boosting technique. We use multiple weak learners. Although a single classifier may not reliably forecast an image's class, we may develop a robust model by combining multiple weak classifiers, each learning from the others' incorrectly categorized items. Based on the weighted samples, a weak classifier is built on top of the training data. The weights of every sample show how critical it is to be categorized accurately.

For the very first base learner, we assign all of the samples the same weight. The current base learner is trained on the data. After training, it calculates the error while training. Based on the error, it updates the weight of the classifier and the weight of the images. If any image is classified incorrectly, then the weights of that image will increase. And if any image is classified correctly, the weight of that image will decrease. Based on these classifier weights and image weights next base learner will train and try to minimize the error. This process is kept on until maximum iteration is reached. In Adaboost, we first extract deep learning features from the spatial pyramid, which is discussed above. These features are used to train a classifier.

4.5.2 Bagging

Bootstrap aggregation is another name for bagging[1]. It's an ensemble learning approach that helps machine learning algorithms increase their performance and accuracy. This method can be used to solve both regression and classification challenges. In this method, multiple base learner is used for training on a subset of the dataset in parallel and independently.

In this method, the first stage was bootstrapping. In this stage, the specific number of equal sizes of datasets was extracted from the original dataset with replacements. This specific number is equal to the number of base or weak learners. In this, we use replacements, some records used in multiple base learners. After training the classifier, the test data is given to all base learners, and these base learners predict output independently. We collect all results from all base learners and based on voting, and it provides the final outcome. which is called aggregation. In this work, we first extract features from the spatial pyramid, which is used to train the classifier.

4.6 Vision transformer

The Vision transformer [7] is one of the most popular algorithms in the area of Deep learning. This algorithm is based on the transformer, which is used in the natural language processing task. In NLP, the internally transformer learns by measuring the relationship between the token pairs. Whereas in Vision transformer, it is a measured relationship between the pair of patches. Here all patches of the image consider a token. The relationship will be learned by applying the attention mechanism in the network. The architecture shown in figure 4.2.

Here we gave raw images as input to the model. Then this model divides the entire image into fixed size patches. In this work size of the patch is 16 x 16, which are nonoverlapping patches. These patches are then flattened and given as input to the linear projection layer, which creates lower dimensional linear embedding from these flattened patches. After that, we include positional embedding. This positional embedding is used to retain positional information of the patches in the image. After adding positional embedding, provide input as a vector to the transformer encoder layer.

A multi-head self-attention layer, multi-layer perceptrons layer, and nomalizing layer make up the transformer encoder. These multi-head attention layers are

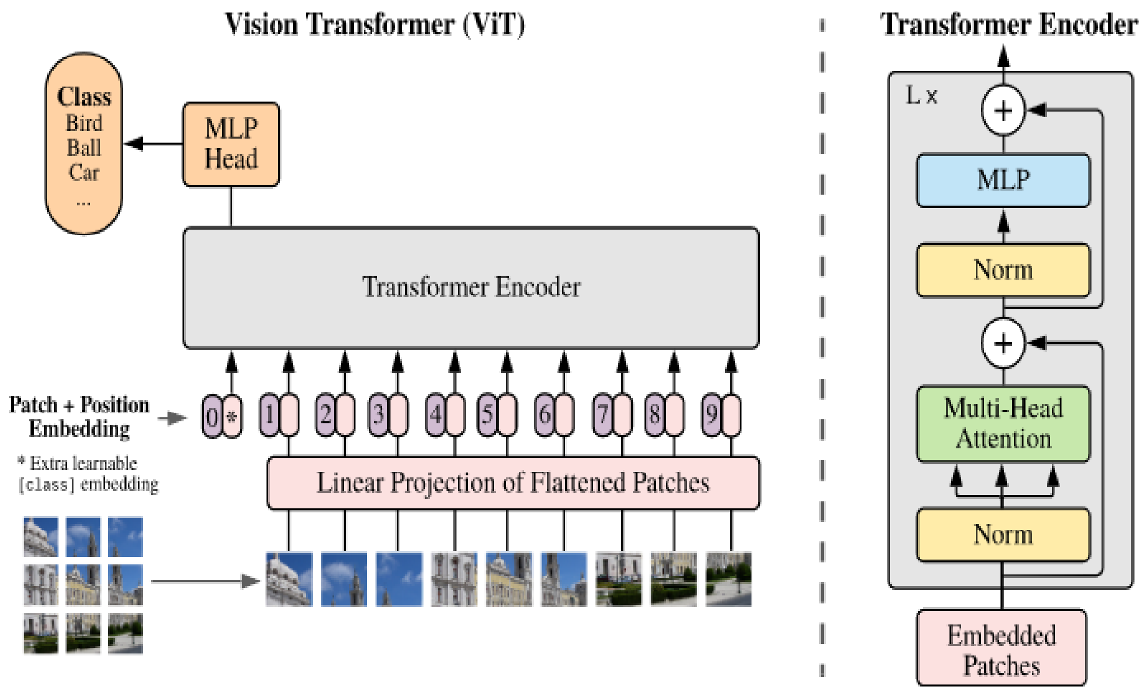


Figure 4.2: Architecture of Vision transformer. [7]. Reprinted, with permission, from A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. CoRR, abs/2010.11929, 2020

used to find the relationship between pairs of patches. The MLP layer is the standard feedforward layer. The normalization layer is added to prior to each block. It helps in improving training time and overall performance. After the multiple encoder block, the MLP head will attach at position 0. We added class embedding, an extra learnable token at position 0 used for classification. At the MLP head, which is attached at position 0, we get the final output of the input image.

The Vision transformer will not work well if you directly train the model on your custom dataset. So, we have to train the vision transformer model on a very large dataset with a high-resolution dataset. Here we have used the ViT B16 model of the vision transformer, which is pretrained on the imagenet 21K dataset, which contains more than 10 million images and 21 thousand classes. This model we have used to train on our custom dataset to improve the model's performance.

CHAPTER 5

Results and Analysis

In this chapter, we have described the results. These results are from the experiments that we have conducted. We show the result table of the model and its accuracy. Also, show and analyze the confusion matrix and outputs of the results.

5.1 Results

We have used two datasets. The DB1 dataset was used in [3] and [4] Previous studies and the Indian dataset which is made by us. The table shows the accuracy of different models. The highest accuracy of occupation prediction from the facial image in the previous study was 77.50% in Discriminant multi-channel support vector machine(DMC-SVM).

Table 5.1: Results of Our Experiments

Results table of classification accuracy (%). The models with higher accuracies than the baseline are represented in bold.

Sr. No.	Model	DB1	Indian
1	SVM	73.6%	74.4%
2	Logistic Regression	72.8%	74.8%
3	Adaboost	72.4%	77%
4	Bagging	75.8%	73.8%
5	**ViT**	**83.6%**	**83.8%**

If we randomly guess the occupation from the five classes, then the probability of getting the right answer is 1/5 means 20% only. Whereas from the table, we can see that all models in both datasets achieved very high accuracy as compared to random guesses. In previous studies, the highest accuracy on occupation prediction from the facial image was 77.50% on the DB1 dataset, and if we compare with our results, then the vision transformer(ViT) gives the best results on both the DB1 and Indian datasets. So, the vision transformer shows an improvement from the previous work on the DB1 dataset.

From the result table, we can see that only Bagging shows a better result on the DB1 dataset as compared to the Indian dataset. Whereas SVM, logistic regression, AdaBoost, and vision transformer perform well on the Indian dataset as compared to the DB1 dataset. Here Vision Transformer achieved 83.6% accuracy on the DB1 dataset, and it achieved 83.8% accuracy on the Indian dataset, Which is the best accuracy among all other image classification algorithms.

5.2 Confusion Matrix

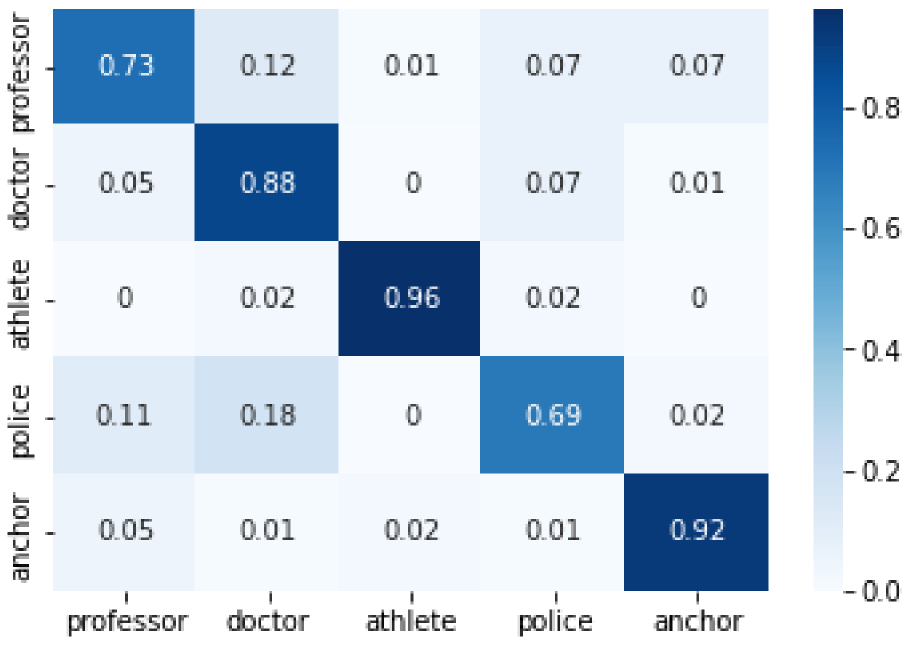


Figure 5.1: Confusion matrix for DB1 dataset on vision transformer model.

The confusion matrix of the vision transformer for the DB1 dataset is shown in Figure 5.1. In the DB1 dataset, the name of classes is Professor, Doctor, Athlete, Police, and Anchor. From the confusion matrix, we can see that the highest accuracy is achieved by athlete class, in which the least misclassification has happened. This is only because of their uniqueness in age and gender. Because most probably, The athletes are younger and male.

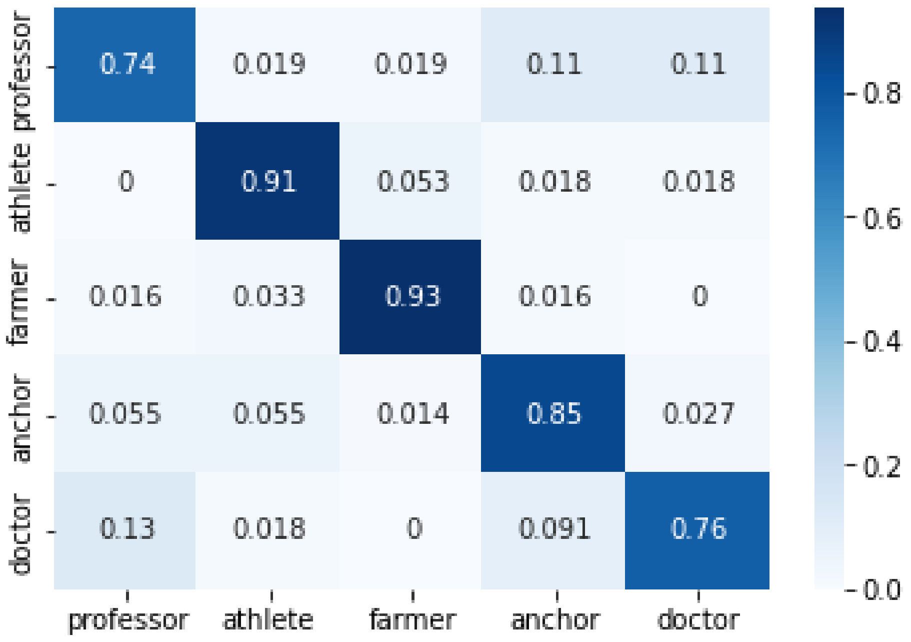


Figure 5.2: Confusion matrix for Indian dataset on vision transformer model.

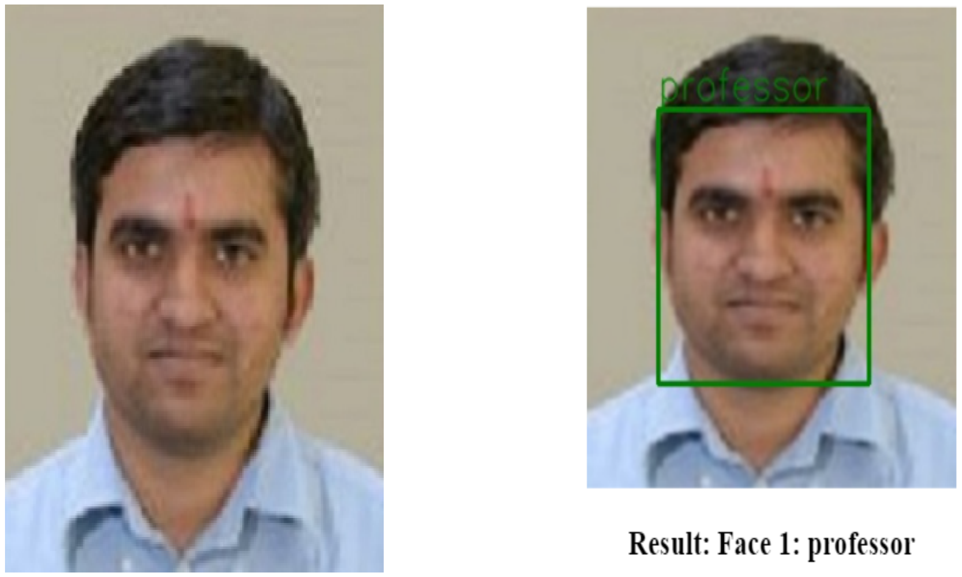
The anchor also has higher accuracy and lesser misclassification rate as compared to the professor, doctor, and police. Because most probably, the anchor is primarily a female and young. So anchor has some uniqueness in gender and age.

From the figure 5.1, we can see that the professor and doctor have the least accuracy and higher misclassification rate. The police class is more confused with the professor and doctor class. Same way professor's class is also confused with the doctor and police class. Because the police, doctor, and professor classes have some similar characteristics. For example, similarity in age. So from this confusion matrix on the DB1 dataset, the anchor and athlete classes have the least, and the police class has a higher misclassification rate.

Figure 5.2 shows the confusion matrix of the vision transformer for the Indian dataset. In the Indian dataset, the name of classes is professor, athlete, farmer, anchor, and doctor. From the above confusion matrix, we can see that the highest accuracy was achieved by the farmer, and it has the least misclassification has happened because of uniqueness in the age. Most probably, the farmer is old people.

Here Athlete and Anchor also achieved more accuracy and lesser misclassification rate as compared to the doctor and professor. Because of uniqueness in age and gender, as discussed above.

The professor and doctor classes have the least accuracy and higher misclassification rate. We can see that the professor class has higher confusion with the



(a) Raw Input Image (b) Final Output

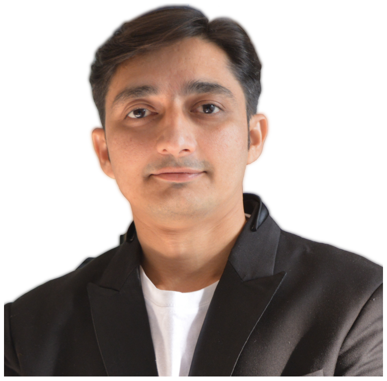
Figure 5.3: Professor correctly classified as Professor

doctor and the doctor class have higher confusion with the professor class. This is because of some similarities in characteristics, for example, age. So in the Indian dataset, the farmer has the highest accuracy and lowest misclassification. At the same time, the professor and doctor classes have higher misclassification.

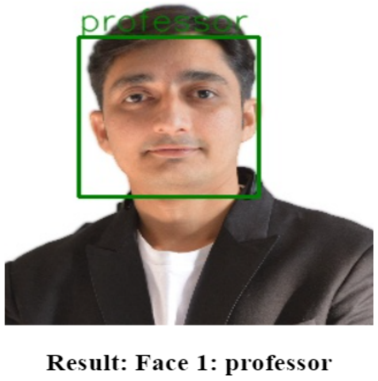
5.3 Outputs

In this section, we have shown the input image and output of that image. Here, the first three pairs of images, Figures 5.3, 5.4, and 5.5, belong to the professor, and the fourth pair of images, figure 5.5, belong to the farmer. All outputs are predicted by the vision transformer model.

From the output images, we can see that in the first two images, figures 5.3 and 5.4, The input images belong to the professor class, and the model is correctly classified as a professor. Whereas in figure 5.5, the input image belongs to the professor's class, but the image was misclassified as an anchor. And at last, figure 5.6, which belongs to the farmer class and is correctly classified as a farmer.

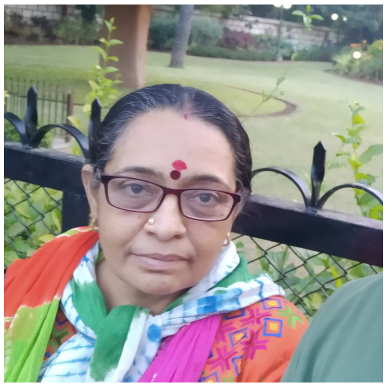


(a) Raw Input Image

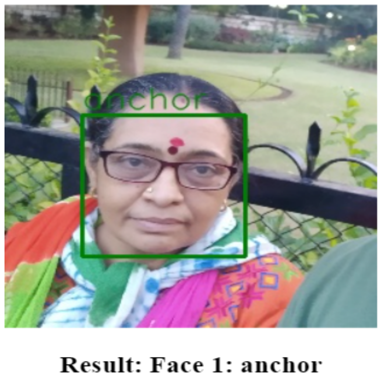


(b) Final Output

Figure 5.4: Professor correctly classified as Professor

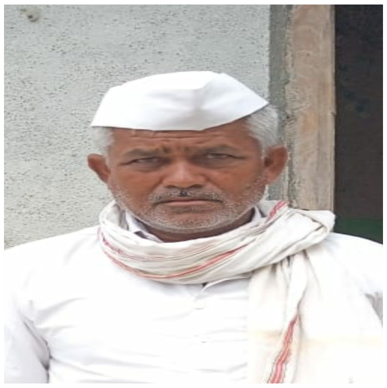


(a) Raw Input Image

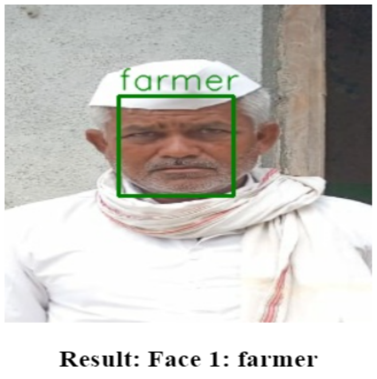


(b) Final Output

Figure 5.5: Professor wrongly classified as Anchor



(a) Raw Input Image



(b) Final Output

Figure 5.6: Farmer correctly classified as Farmer

CHAPTER 6

Applications and Challenges

In this section, we briefly discussed the application and challenges we faced during the study. In the application section, we have discussed all possible applications of the occupation prediction from the person's facial image. In the second section, we have discussed the challenges that we have faced during the research work.

6.1 Applications

In this section, we are going to discuss possible applications of occupation prediction from the facial image of a person. The occupation prediction can be used in various fields like e-commerce, recommendation system, etc.

If we can predict occupation from the facial image, then it is very beneficial in e-commerce to recommend specific products. This recommendation will be based on the occupation. Because in some occupations, the person needs some products that are used frequently. By using this occupation prediction, we can recommend an appropriate product. which is very beneficial for the business. Also, we can use occupation prediction in advertisements. This application is very useful for targeting a specific audience by showing ads for a certain product. So this occupation prediction can be used for marketing as well. And it is advantageous for businesses.

Another application of this work can be used in social media. There are many social media and community websites. In this type of web application, we can suggest friends, groups, and pages by using occupation prediction. Because on this type of platform, there are many groups and pages which are directly related to the profession. Also, we can suggest friends based on the profession. So, the occupation can be used on social media platforms as well.

6.2 Challenges

This section has discussed various challenges that we have faced while conducting this work. One of the challenges is about the related work. In previous studies, a very less amount of work was done on occupation prediction from facial images of the person. Most of the studies are based on appearance like human clothing, scene context, social context, etc. So one of the challenges is previous research related to this definition is very limited.

In occupation prediction from the facial image, the first task is to select the classes. This means selecting the occupation and number of occupations that we have to predict. There are many occupations, but we have selected some common occupations: doctor, actor, athlete, professor, and farmer.

After selecting the occupation, the next big challenge is to gather data. In this data, we need a facial image of a person and his occupation. Also, we need the consent of the person to use their facial image and data for this work. Many people are not willing to give their data, and some people do not give their facial images because of the privacy issue. We have tried to gather the data from various websites like Linkedin, emails, WhatsApp, and from our networks and request to give the data. We have circulated the google form on the above channels so people can give the data.

The other challenge is that predicting occupation from the facial image is one of the difficult tasks. In this definition, we do not get very high accuracy. And if we increase the number of classes, then the accuracy will also decrease. So these are the challenges of this work.

CHAPTER 7
Conclusion and Future work

In this chapter, we have included a conclusion that is derived from our work. And also included potential future work.

7.1 Conclusion

The occupation prediction from the facial image of a person is a very inspiring problem to solve. This work doesn't receive much attention in this field. Many rich facial features like gender, age, hairstyle, etc., are related to occupation. Those facial features are advantageous to predicting occupation from a facial image of a person. In this work, we have implemented various image classification algorithms. We also created our own dataset that is based on Indian people. From all experiments, we can say that the vision transformer gives the best performance on both datasets. Vision transformer also shows improvement as compared to previous studies.

7.2 Future Work

As part of future work, we can expand our custom Indian dataset. We can add more images per class. So there can be a chance of an increase in the accuracy and performance of the model. Also, we can add more classes. So that model can predict more number of occupations. We can make multiple datasets that belong to different nationalities and can find patterns. By using these types of different datasets, we can also compare them.

References

[1] L. Breiman. Bagging predictors. In *Machine learning*, volume 24, pages 123–140. Springer, 1996.

[2] H. Chen, A. C. Gallagher, and B. Girod. What's in a name? first names as facial attributes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2013.*, pages 3366–3373, 2013.

[3] W.-T. Chu and C.-H. Chiu. Predicting occupation from single facial images. In *2014 IEEE International Symposium on Multimedia*, pages 9–12, 2014.

[4] W.-T. Chu and C.-H. Chiu. Predicting occupation from images by combining face and body context information. In *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, volume 13, pages 1–21. ACM New York, NY, USA, 2016.

[5] C. Cortes and V. Vapnik. Support-vector networks. In *Machine learning*, volume 20, pages 273–297. Springer, 1995.

[6] J. S. Cramer. The origins of logistic regression. In *Tinbergen Institute Discussion Paper*, number 2002-119/4, pages 167–178. Tinbergen Institute, 2002.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[8] T. Hastie, S. Rosset, J. Zhu, and H. Zou. Multi-class adaboost. In *Statistics and its Interface*, volume 2, pages 349–360. International Press of Boston, 2009.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 37, pages 1904–1916, 2015.

[10] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2169–2178, 2006.

[11] D. G. Lowe. Distinctive image features from scale-invariant keypoints. In *International journal of computer vision*, volume 60, pages 91–110. Springer, 2004.

[12] M. Shao, L. Li, and Y. Fu. What do you do? occupation recognition in a photo via social context. In *2013 IEEE International Conference on Computer Vision*, pages 3631–3638, 2013.

[13] Z. Song, M. Wang, X. sheng Hua, and S. Yan. Predicting occupation via human clothing and contexts. In *2011 International Conference on Computer Vision*, pages 1084–1091, 2011.

[14] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3360–3367, 2010.

CHAPTER A
Source code

A.1 Preprocessing of collected dataset

```
####################################################################
'''
I am Darshankumar Zalavadiya, M.Tech (ICT) student at Dhirubhai
Ambani Institute of Information and Communication Technology.
My M.Tech Thesis supervisor is Prof. Manish Gupta, DA-IICT.
This source code is the part of my M.Tech thesis.

In this code, we have used the following libraries:

- OpenCV: Available at: https://opencv.org/
- OS: Available at: https://docs.python.org/3/library/os.html

  The source code also available at :
  https://github.com/darshan154/occupation-prediction
'''
####################################################################

# This code preprocesses the collected dataset and generates
# an output dataset with facial images.
# All input images must be arranged classwise in the folder.

import cv2
import os

root_path = "Path of dataset"
save_path = "Path of output folder"
imagePath = root_path


for img_class in os.listdir(root_path):
    cnt=0
    save_path_file=os.path.join(save_path,img_class)
    try:
      os.mkdir(save_path_file)
    except OSError as error:
      print(error)

    for img in os.listdir(os.path.join(root_path,img_class)):

      path_=os.path.join(root_path,img_class,img)
      image = cv2.imread(path_)
      gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

      faceCascade = cv2.CascadeClassifier(cv2.data.haarcascades + "
                                           haarcascade_frontalface_default
                                           .xml")
      faces = faceCascade.detectMultiScale(
          gray,
          scaleFactor=1.3,
          minNeighbors=3,
          minSize=(30, 30)
      )


      for (x, y, w, h) in faces:
          roi_color = image[y:y + h, x:x + w]
          roi_color = cv2.resize(roi_color, (64, 64))
          filename=img_class + '_' + str(cnt) + '.jpg'
          cv2.imwrite(os.path.join(save_path_file,filename),
                                                  roi_color)

      status = cv2.imwrite('faces_detected.jpg', image)
      cnt+=1
```

A.2 Vision Transformer

```
###################################################################
'''
I am Darshankumar Zalavadiya, M.Tech (ICT) student at Dhirubhai
Ambani Institute of Information and Communication Technology.
My M.Tech Thesis supervisor is Prof. Manish Gupta, DA-IICT.
This source code is the part of my M.Tech thesis.

```

```
In this code, we have used the following libraries:

- OpenCV: Available at: https://opencv.org/
- OS: Available at: https://docs.python.org/3/library/os.html
- matplotlib: Available at: https://matplotlib.org/
- numpy: Available at: https://numpy.org/
- pathlib: Available at: https://docs.python.org/3/library/pathlib.
  html
- torch: Available at: http://torch.ch/
- glob: Available at: https://docs.python.org/3/library/glob.html
- pytorch lightning: Available at: https://www.pytorchlightning.ai/
- torchvision: Available at: https://pytorch.org/vision/stable/
  index.html
- transformer: Available at: https://huggingface.co/docs/
  transformers/index
- tensorflow: Available at: https://www.tensorflow.org/
- sklearn: Available at: https://scikit-learn.org/stable/
- seaborn: Available at: https://seaborn.pydata.org/
- datasets: Available at: https://huggingface.co/docs/datasets/index
- keras: Available at: https://keras.io/
- scikitplot: Available at: https://scikit-plot.readthedocs.io/en/
  stable/index.html

  The source code also available at :
  https://github.com/darshan154/occupation-prediction
'''

####################################################################

!pip install transformers pytorch-lightning
!sudo apt -qq install git-lfs
!pip install transformers "datasets>=1.17.0" tensorboard --upgrade
import math
import matplotlib.pyplot as plt
import numpy as np
from PIL import Image, UnidentifiedImageError
from pathlib import Path
import torch
import glob
import pytorch_lightning as pl
from torch.utils.data import DataLoader
from torchvision.datasets import ImageFolder
from torchmetrics import Accuracy
from transformers import ViTFeatureExtractor,
                                    ViTForImageClassification,
                                  TFViTForImageClassification,
                                  create_optimizer,
                                  DefaultDataCollator
from pytorch_lightning.callbacks import ModelCheckpoint
import matplotlib.pyplot as plt
import random
import cv2
import os
import tensorflow as tf
from sklearn.metrics import confusion_matrix, classification_report
import seaborn as sns

!pip install datasets

# download dataset
!git clone https://github.com/darshan154/db2.git

import datasets

def create_image_folder_dataset(root_path):
  """creates ‘Dataset‘ from image folder structure"""

  # get class names by folders names
  _CLASS_NAMES = os.listdir(root_path)
  # defines ‘datasets‘ features‘
  features=datasets.Features({
                    "img": datasets.Image(),
                    "label": datasets.features.ClassLabel(names=
                                                          _CLASS_NAMES
                                                          ),
                })
  # temp list holding datapoints for creation
  img_data_files=[]
  label_data_files=[]
  # load images into list for creation
  for img_class in os.listdir(root_path):
    for img in os.listdir(os.path.join(root_path,img_class)):
      path_=os.path.join(root_path,img_class,img)
      img_data_files.append(path_)
      label_data_files.append(img_class)
  # create dataset
  ds = datasets.Dataset.from_dict({"img":img_data_files,"label":
                                  label_data_files},features=
                                  features)
  return ds
```

```
# path to dataset
occupation_ds = create_image_folder_dataset("./db2/data")
occupation_ds

class_labels = occupation_ds.features["label"].names
id_model = "google/vit-base-patch16-224"

from tensorflow.keras import layers

feature_extractor = ViTFeatureExtractor.from_pretrained(model_id)

def process(examples):
    examples.update(feature_extractor(examples['img'], ))
    return examples

occupation_ds = occupation_ds.rename_column("label", "labels")
processed_dataset = occupation_ds.map(process, batched=True)
% processed_dataset

test_size=.30
processed_dataset = processed_dataset.shuffle().train_test_split(
                                  test_size=test_size)

id_2_label = {str(i): label for i, label in enumerate(class_labels)}
label_2_id = {v: k for k, v in id_2_label.items()}

train_epochs = 7
training_batch_size = 64
evaluation_batch_size = 64
learning_rate = 1e-4
decay=0.4
warmup_steps=0
output_dir=id_model.split("/")[1]

collator = DefaultDataCollator(return_tensors="tf")

train_dataset = processed_dataset["train"].to_tf_dataset(
   columns=['pixel_values'],
   label_cols=["labels"],
   shuffle=True,
   batch_size=training_batch_size,
   collate_fn=collator)
```

```
eval_dataset = processed_dataset["test"].to_tf_dataset(
   columns=['pixel_values'],
   label_cols=["labels"],
   shuffle=False,
   batch_size=evaluation_batch_size,
   collate_fn=collator)

train_steps = len(train_dataset) * train_epochs
optimizer, lr_schedule = create_optimizer(
    init_lr=learning_rate,
    num_train_steps=train_steps,
    weight_decay_rate=decay,
    num_warmup_steps=warmup_steps,
)

model = TFViTForImageClassification.from_pretrained(
    id_model,
    num_labels=len(class_labels),
    ignore_mismatched_sizes=True,
    id2label=id_2_label,
    label2id=label_2_id,
)

loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True)

metrics=[
    tf.keras.metrics.SparseCategoricalAccuracy(name="accuracy"),
    tf.keras.metrics.SparseTopKCategoricalAccuracy(3, name="top-3-
                                       accuracy"),
]

model.compile(optimizer=optimizer,
              loss=loss,
              metrics=metrics
              )

from tensorflow.keras.callbacks import TensorBoard as
                                  TensorboardCallback, EarlyStopping

callbacks=[]
callbacks.append(TensorboardCallback(log_dir=os.path.join(output_dir
                                  ,"logs")))
callbacks.append(EarlyStopping(monitor="val_accuracy",patience=3))

res = model.fit(
    train_dataset,
    validation_data=eval_dataset,
    callbacks=callbacks,
    epochs=train_epochs,
)

lbls = []
for i in range(len(id_2_label)):
    lbls.append(id_2_label[str(i)])

ypred = model.predict(eval_dataset)
ypred = ypred.logits.argmax(-1)
true_categories = tf.concat([y for x, y in eval_dataset], axis=0)
true_categories = np.array(true_categories)

ypred = ypred.reshape((ypred.shape[0], 1))
true_categories = true_categories.reshape((true_categories.shape[0],
                                            1))
print("Accuracy:",metrics.accuracy_score(true_categories, ypred))

ax = plt.subplot()
confusionmatrix = confusion_matrix(true_categories, ypred, normalize
                                   ='true')
sns.heatmap(confusionmatrix, cmap = 'Blues', annot = True, cbar =
                                   True, ax = ax)
ax.xaxis.set_ticklabels(lbls)
ax.yaxis.set_ticklabels(lbls)
```

A.3 Support vector machine

```
##################################################################
'''
I am Darshankumar Zalavadiya, M.Tech (ICT) student at Dhirubhai
Ambani Institute of Information and Communication Technology.
My M.Tech Thesis supervisor is Prof. Manish Gupta, DA-IICT.
This source code is the part of my M.Tech thesis.

In this code, we have used the following libraries:

- OpenCV: Available at: https://opencv.org/
- OS: Available at: https://docs.python.org/3/library/os.html
- math: Available at: https://docs.python.org/3/library/math.html
- matplotlib: Available at: https://matplotlib.org/
- numpy: Available at: https://numpy.org/
- glob: Available at: https://docs.python.org/3/library/glob.html
- tensorflow: Available at: https://www.tensorflow.org/
- sklearn: Available at: https://scikit-learn.org/stable/
- seaborn: Available at: https://seaborn.pydata.org/
- keras: Available at: https://keras.io/
- random: Available at: https://docs.python.org/3/library/random.
  html
- scikitplot: Available at: https://scikit-plot.readthedocs.io/en/
  stable/index.html

  The source code also available at :
  https://github.com/darshan154/occupation-prediction
'''
####################################################################

# download dataset
!git clone https://github.com/darshan154/data.git

from tensorflow.keras.preprocessing.image import img_to_array
from keras.applications.vgg16 import VGG16
from keras.models import Model
from keras.layers import Dense, MaxPooling2D, Flatten
import math
from PIL import Image
import matplotlib.pyplot as plt
import numpy as np
import random
import cv2
import os
import glob

train_file = []
for filename in glob.glob(r"./data/occ240/occupation/train/" + "
                                  /**/*", recursive=True):
    train_file.append(filename)

test_file = []
for filename in glob.glob(r"./data/occ240/occupation/test/" + "/**/*
                                  ", recursive=True):
    test_file.append(filename)

random.shuffle(train_file)
random.shuffle(test_file)
print(len(train_file))
```

```
label = {"doctor":0, "news":1, "player":2, "police":3, "teachers": 4
                                  }

trainx = []
testx = []
trainy = []
testy = []

for img in train_file:
  try:
    image = cv2.imread(img)
    image = cv2.resize(image, (64, 64))
    image = img_to_array(image)
    trainx.append(image)

    lbl = img.split(os.path.sep)[-2]
    l = label[lbl]

    trainy.append([l]) # [[1], [0], [0], ...]

  except:
    print("Corrupt")


for img in test_file:
  try:
    image = cv2.imread(img)
    image = cv2.resize(image, (64, 64))
    image = img_to_array(image)
    testx.append(image)

    lbl = img.split(os.path.sep)[-2]
    l = label[lbl]

    # trainy.append([l])

    testy.append([l]) # [[1], [0], [0], ...]
  except:
    print("Corrupt")

trainx = np.array(trainx, dtype="float")/255.0
testx = np.array(testx, dtype="float")/255.0

# load model without classifier layers
```

```
base_model = VGG16(include_top=False, weights='imagenet',
                                    input_shape=(64, 64, 3))

win1 = math.ceil(16/1)
str1 = math.floor(16/1)

win2 = math.ceil(16/2)
str2 = math.floor(16/2)

win3 = math.ceil(16/4)
str3 = math.floor(16/4)

l1 = MaxPooling2D(pool_size=(win1), strides=str1, padding="valid")(
                                    base_model.layers[-10].output)
l2 = MaxPooling2D(pool_size=(win2), strides=str2, padding="valid")(
                                    base_model.layers[-10].output)
l3 = MaxPooling2D(pool_size=(win3), strides=str3, padding="valid")(
                                    base_model.layers[-10].output)

flat1 = Flatten()(l1)
flat2 = Flatten()(l2)
flat3 = Flatten()(l3)

for layer in base_model.layers:
    layer.trainable = False

model = Model(inputs=base_model.inputs, outputs=[flat1, flat2, flat3
                                    ])
trainy = np.array(trainy)
testy = np.array(testy)

train = model.predict(trainx)
test = model.predict(testx)
train = np.hstack([train[0], train[1], train[2]])
test = np.hstack([test[0], test[1], test[2]])

from sklearn.svm import SVC
clf=SVC(probability=True, kernel="linear")
clfmodel = clf.fit(train, trainy)
y_pred = clfmodel.predict(test)

!pip install scikit-plot
from sklearn.metrics import confusion_matrix
import scikitplot as skplt
skplt.metrics.plot_confusion_matrix(testy, y_pred, normalize=True)
```

```
plt.show()
```

A.4 Logistic Regression

```
######################################################################
'''
I am Darshankumar Zalavadiya, M.Tech (ICT) student at Dhirubhai
Ambani Institute of Information and Communication Technology.
My M.Tech Thesis supervisor is Prof. Manish Gupta, DA-IICT.
This source code is the part of my M.Tech thesis.

In this code, we have used the following libraries:

- OpenCV: Available at: https://opencv.org/
- OS: Available at: https://docs.python.org/3/library/os.html
- math: Available at: https://docs.python.org/3/library/math.html
- matplotlib: Available at: https://matplotlib.org/
- numpy: Available at: https://numpy.org/
- glob: Available at: https://docs.python.org/3/library/glob.html
- tensorflow: Available at: https://www.tensorflow.org/
- sklearn: Available at: https://scikit-learn.org/stable/
- seaborn: Available at: https://seaborn.pydata.org/
- keras: Available at: https://keras.io/
- random: Available at: https://docs.python.org/3/library/random.
  html
- scikitplot: Available at: https://scikit-plot.readthedocs.io/en/
  stable/index.html

  The source code also available at :
  https://github.com/darshan154/occupation-prediction
'''
######################################################################

#download dataset
!git clone https://github.com/darshan154/data.git

from tensorflow.keras.preprocessing.image import img_to_array
from keras.applications.vgg16 import VGG16
from keras.models import Model
from keras.layers import Dense, MaxPooling2D, Flatten
import math
from PIL import Image
import matplotlib.pyplot as plt
import numpy as np
import random
import cv2
import os
import glob

train_file = []
for filename in glob.glob(r"./data/occ240/occupation/train/" + "
                                  /**/*", recursive=True):
    train_file.append(filename)

test_file = []
for filename in glob.glob(r"./data/occ240/occupation/test/" + "/**/*
                                  ", recursive=True):
    test_file.append(filename)

random.shuffle(train_file)
random.shuffle(test_file)

label = {"doctor":0, "news":1, "player":2, "police":3, "teachers": 4
                                  }

trainx = []
testx = []
trainy = []
testy = []

for img in train_file:
  try:
    image = cv2.imread(img)
    image = cv2.resize(image, (64, 64))
    image = img_to_array(image)
    trainx.append(image)

    lbl = img.split(os.path.sep)[-2]
    l = label[lbl]

    trainy.append([l]) # [[1], [0], [0], ...]

  except:
    print("Corrupt")


for img in test_file:
  try:
    image = cv2.imread(img)
    image = cv2.resize(image, (64, 64))
    image = img_to_array(image)
    testx.append(image)

    lbl = img.split(os.path.sep)[-2]
    l = label[lbl]

    # trainy.append([l])

    testy.append([l]) # [[1], [0], [0], ...]
  except:
    print("Corrupt")

trainx = np.array(trainx, dtype="float")/255.0
testx = np.array(testx, dtype="float")/255.0
trainy = np.array(trainy)
testy = np.array(testy)

base_model = VGG16(include_top=False, weights='imagenet',
                                  input_shape=(64, 64, 3))

win1 = math.ceil(16/1)
str1 = math.floor(16/1)

win2 = math.ceil(16/2)
str2 = math.floor(16/2)

win3 = math.ceil(16/4)
str3 = math.floor(16/4)

l1 = MaxPooling2D(pool_size=(win1), strides=str1, padding="valid")(
                                  base_model.layers[-10].output)
l2 = MaxPooling2D(pool_size=(win2), strides=str2, padding="valid")(
                                  base_model.layers[-10].output)
l3 = MaxPooling2D(pool_size=(win3), strides=str3, padding="valid")(
                                  base_model.layers[-10].output)

flat1 = Flatten()(l1)
flat2 = Flatten()(l2)
flat3 = Flatten()(l3)

for layer in base_model.layers:
    layer.trainable = False
```

```
model = Model(inputs=base_model.inputs, outputs=[flat1, flat2, flat3
                                   ])

train = model.predict(trainx)
test = model.predict(testx)
train = np.hstack([train[0], train[1], train[2]])
test = np.hstack([test[0], test[1], test[2]])

from sklearn.linear_model import LogisticRegression
clf=LogisticRegression()
clfmodel = clf.fit(train, trainy)
y_pred = clfmodel.predict(test)


!pip install scikit-plot
from sklearn.metrics import confusion_matrix
import scikitplot as skplt
skplt.metrics.plot_confusion_matrix(testy, y_pred, normalize=True)
plt.show()
```

A.5 Adaboost

```
####################################################################
'''
I am Darshankumar Zalavadiya, M.Tech (ICT) student at Dhirubhai
Ambani Institute of Information and Communication Technology.
My M.Tech Thesis supervisor is Prof. Manish Gupta, DA-IICT.
This source code is the part of my M.Tech thesis.

In this code, we have used the following libraries:

- OpenCV: Available at: https://opencv.org/
- OS: Available at: https://docs.python.org/3/library/os.html
- math: Available at: https://docs.python.org/3/library/math.html
- matplotlib: Available at: https://matplotlib.org/
- numpy: Available at: https://numpy.org/
- glob: Available at: https://docs.python.org/3/library/glob.html
- tensorflow: Available at: https://www.tensorflow.org/
- sklearn: Available at: https://scikit-learn.org/stable/
- seaborn: Available at: https://seaborn.pydata.org/
- keras: Available at: https://keras.io/
- random: Available at: https://docs.python.org/3/library/random.
  html
```

```
- scikitplot: Available at: https://scikit-plot.readthedocs.io/en/
  stable/index.html

  The source code also available at :
  https://github.com/darshan154/occupation-prediction
'''
######################################################################

# download dataset
!git clone https://github.com/darshan154/data.git

from tensorflow.keras.preprocessing.image import img_to_array
from keras.applications.vgg16 import VGG16
from keras.models import Model
from keras.layers import Dense, MaxPooling2D, Flatten
import math
from PIL import Image
import matplotlib.pyplot as plt
import numpy as np
import random
import cv2
import os
import glob

train_file = []
for filename in glob.glob(r"./data/occ240/occupation/train/" + "
                                    /**/*", recursive=True):
    train_file.append(filename)

test_file = []
for filename in glob.glob(r"./data/occ240/occupation/test/" + "/**/*
                                    ", recursive=True):
    test_file.append(filename)

random.shuffle(train_file)
random.shuffle(test_file)
label = {"doctor":0, "news":1, "player":2, "police":3, "teachers": 4
                                    }

trainx = []
testx = []
trainy = []
testy = []

for img in train_file:
  try:
    image = cv2.imread(img)
    image = cv2.resize(image, (64, 64))
    image = img_to_array(image)
    trainx.append(image)

    lbl = img.split(os.path.sep)[-2]
    l = label[lbl]

    trainy.append([l]) # [[1], [0], [0], ...]

  except:
    print("Corrupt")


for img in test_file:
  try:
    image = cv2.imread(img)
    image = cv2.resize(image, (64, 64))
    image = img_to_array(image)
    testx.append(image)

    lbl = img.split(os.path.sep)[-2]
    l = label[lbl]

    # trainy.append([l])

    testy.append([l]) # [[1], [0], [0], ...]
  except:
    print("Corrupt")

trainx = np.array(trainx, dtype="float")/255.0
testx = np.array(testx, dtype="float")/255.0
trainy = np.array(trainy)
testy = np.array(testy)

base_model = VGG16(include_top=False, weights='imagenet',
                                  input_shape=(64, 64, 3))

win1 = math.ceil(16/1)
str1 = math.floor(16/1)

win2 = math.ceil(16/2)
str2 = math.floor(16/2)
```

```
win3 = math.ceil(16/4)
str3 = math.floor(16/4)

l1 = MaxPooling2D(pool_size=(win1), strides=str1, padding="valid")(
                                    base_model.layers[-10].output)
l2 = MaxPooling2D(pool_size=(win2), strides=str2, padding="valid")(
                                    base_model.layers[-10].output)
l3 = MaxPooling2D(pool_size=(win3), strides=str3, padding="valid")(
                                    base_model.layers[-10].output)

flat1 = Flatten()(l1)
flat2 = Flatten()(l2)
flat3 = Flatten()(l3)

for layer in base_model.layers:
    layer.trainable = False

model = Model(inputs=base_model.inputs, outputs=[flat1, flat2, flat3
                                    ])

from sklearn.ensemble import AdaBoostClassifier
from sklearn.svm import SVC
svc=SVC(probability=True, kernel="linear")
clf = AdaBoostClassifier(n_estimators=21, base_estimator=svc,
                                    learning_rate=0.5, algorithm='
                                    SAMME.R')

train = model.predict(trainx)
test = model.predict(testx)
train = np.hstack([train[0], train[1], train[2]])
test = np.hstack([test[0], test[1], test[2]])

clfmodel = clf.fit(train, trainy)
y_pred = clfmodel.predict(test)

!pip install scikit-plot
import scikitplot as skplt
skplt.metrics.plot_confusion_matrix(testy, y_pred, normalize=True)
plt.show()
```

A.6 Bagging

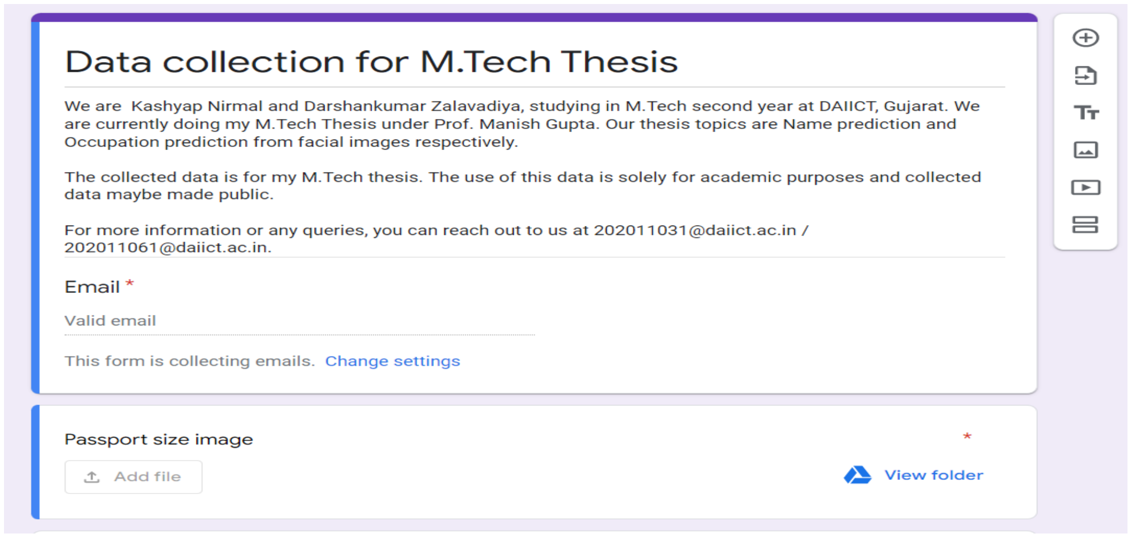
```
########################################################################
```

```
'''
I am Darshankumar Zalavadiya, M.Tech (ICT) student at Dhirubhai
Ambani Institute of Information and Communication Technology.
My M.Tech Thesis supervisor is Prof. Manish Gupta, DA-IICT.
This source code is the part of my M.Tech thesis.

In this code, we have used the following libraries:

- OpenCV: Available at: https://opencv.org/
- OS: Available at: https://docs.python.org/3/library/os.html
- math: Available at: https://docs.python.org/3/library/math.html
- matplotlib: Available at: https://matplotlib.org/
- numpy: Available at: https://numpy.org/
- glob: Available at: https://docs.python.org/3/library/glob.html
- tensorflow: Available at: https://www.tensorflow.org/
- sklearn: Available at: https://scikit-learn.org/stable/
- seaborn: Available at: https://seaborn.pydata.org/
- keras: Available at: https://keras.io/
- random: Available at: https://docs.python.org/3/library/random.
  html
- scikitplot: Available at: https://scikit-plot.readthedocs.io/en/
  stable/index.html

  The source code also available at :
  https://github.com/darshan154/occupation-prediction
'''
##################################################################

!git clone https://github.com/darshan154/data.git
from tensorflow.keras.preprocessing.image import img_to_array
from keras.applications.vgg16 import VGG16
from keras.models import Model
from keras.layers import Dense, MaxPooling2D, Flatten
import math
from PIL import Image
import matplotlib.pyplot as plt
import numpy as np
import random
import cv2
import os
import glob

train_file = []
for filename in glob.glob(r"./data/occ240/occupation/train/" + "
                                  /**/*", recursive=True):
```

```
    train_file.append(filename)

test_file = []
for filename in glob.glob(r"./data/occ240/occupation/test/" + "/**/*
                                    ", recursive=True):
    test_file.append(filename)

random.shuffle(train_file)
random.shuffle(test_file)
label = {"doctor":0, "news":1, "player":2, "police":3, "teachers": 4
                                    }

trainx = []
testx = []
trainy = []
testy = []

for img in train_file:
  try:
    image = cv2.imread(img)
    image = cv2.resize(image, (64, 64))
    image = img_to_array(image)
    trainx.append(image)

    lbl = img.split(os.path.sep)[-2]
    l = label[lbl]

    trainy.append([l]) # [[1], [0], [0], ...]

  except:
    print("a")


for img in test_file:
  try:
    image = cv2.imread(img)
    image = cv2.resize(image, (64, 64))
    image = img_to_array(image)
    testx.append(image)

    lbl = img.split(os.path.sep)[-2]
    l = label[lbl]

    # trainy.append([l])
    testy.append([1]) # [[1], [0], [0], ...]
  except:
    print("b")

trainx = np.array(trainx, dtype="float")/255.0
testx = np.array(testx, dtype="float")/255.0
trainy = np.array(trainy)
testy = np.array(testy)

base_model = VGG16(include_top=False, weights='imagenet',
                                    input_shape=(64, 64, 3))

win1 = math.ceil(16/1)
str1 = math.floor(16/1)

win2 = math.ceil(16/2)
str2 = math.floor(16/2)

win3 = math.ceil(16/4)
str3 = math.floor(16/4)

l1 = MaxPooling2D(pool_size=(win1), strides=str1, padding="valid")(
                                    base_model.layers[-12].output)
l2 = MaxPooling2D(pool_size=(win2), strides=str2, padding="valid")(
                                    base_model.layers[-12].output)
l3 = MaxPooling2D(pool_size=(win3), strides=str3, padding="valid")(
                                    base_model.layers[-12].output)

flat1 = Flatten()(l1)
flat2 = Flatten()(l2)
flat3 = Flatten()(l3)

for layer in base_model.layers:
    layer.trainable = False

model = Model(inputs=base_model.inputs, outputs=[flat1, flat2, flat3
                                    ])

from sklearn.ensemble import BaggingClassifier
from sklearn.svm import SVC
svc=SVC(probability=True, kernel="linear")
clf = BaggingClassifier(base_estimator=svc, n_estimators=21)

a = model.predict(trainx)
b = model.predict(testx)
```

```
a = np.hstack([a[0], a[1], a[2]])
b = np.hstack([b[0], b[1], b[2]])
clfmodel = clf.fit(a, trainy)
y_pred = clfmodel.predict(b)
print("Accuracy:",metrics.accuracy_score(testy, y_pred))

!pip install scikit-plot
import scikitplot as skplt
skplt.metrics.plot_confusion_matrix(testy, y_pred, normalize=True)
plt.show()
```

CHAPTER B
Google Form



Data collection for M.Tech Thesis

We are Kashyap Nirmal and Darshankumar Zalavadiya, studying in M.Tech second year at DAIICT, Gujarat. We are currently doing my M.Tech Thesis under Prof. Manish Gupta. Our thesis topics are Name prediction and Occupation prediction from facial images respectively.

The collected data is for my M.Tech thesis. The use of this data is solely for academic purposes and collected data maybe made public.

For more information or any queries, you can reach out to us at 202011031@daiict.ac.in / 202011061@daiict.ac.in.

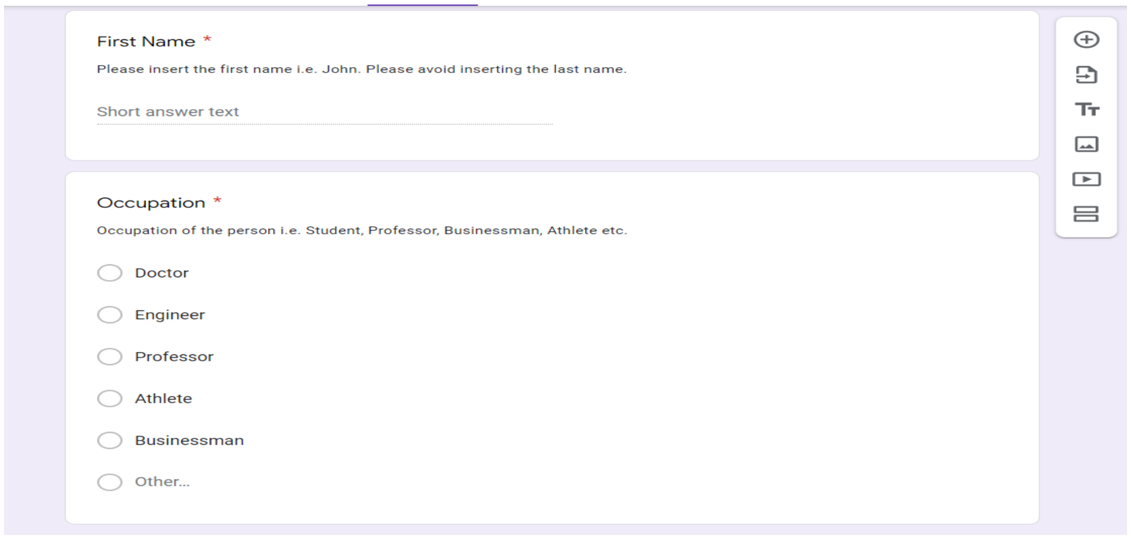
Email *

Valid email

This form is collecting emails. Change settings

Passport size image *

Figure B.1: Snap shot for Title and Introduction of Google form.



First Name *

Please insert the first name i.e. John. Please avoid inserting the last name.

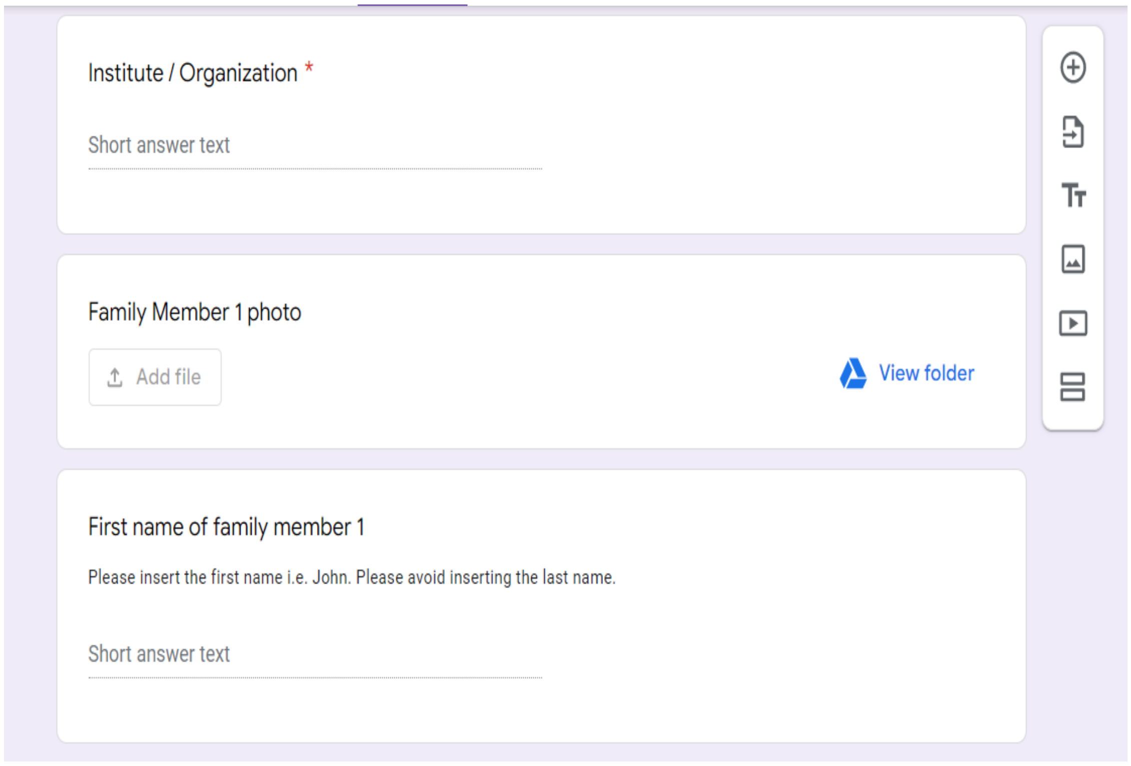
Short answer text

Occupation *

Occupation of the person i.e. Student, Professor, Businessman, Athlete etc.

- ○ Doctor
- ○ Engineer
- ○ Professor
- ○ Athlete
- ○ Businessman
- ○ Other...

Figure B.2: Snap shot1 for information field of Google form.


Institute / Organization *

Short answer text

Family Member 1 photo

Add file

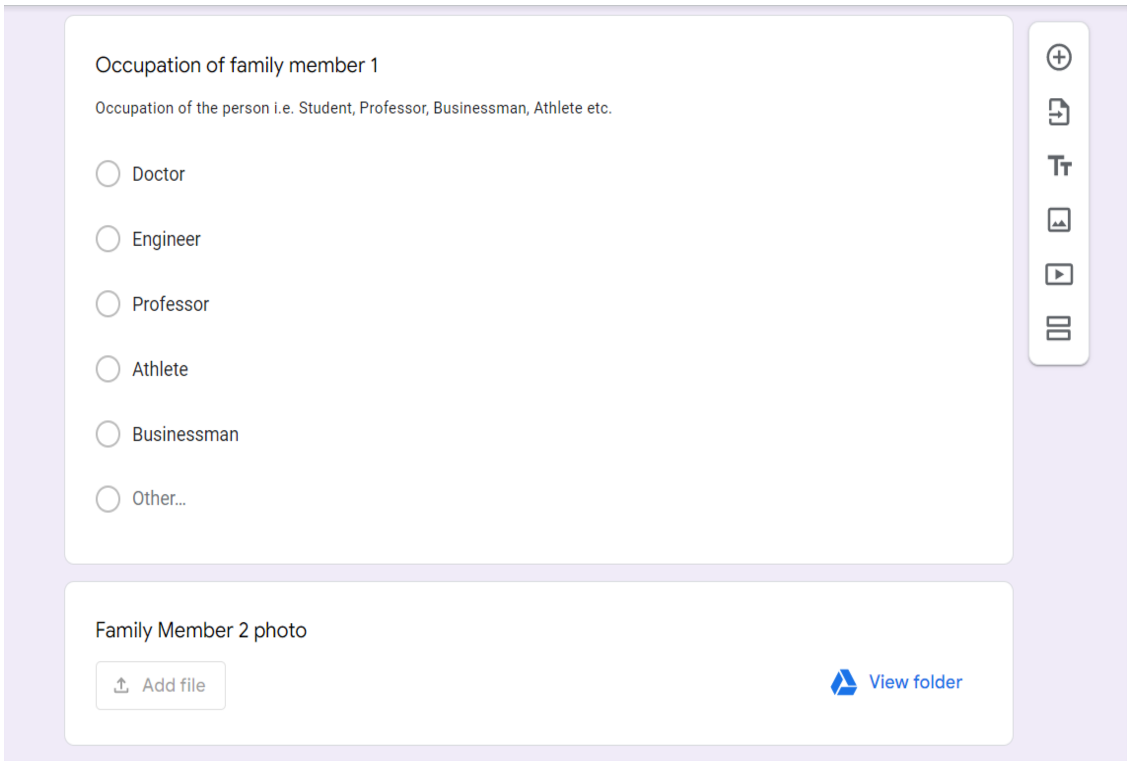
View folder

First name of family member 1

Please insert the first name i.e. John. Please avoid inserting the last name.

Short answer text

Figure B.3: Snap shot2 for information field of Google form.


Occupation of family member 1

Occupation of the person i.e. Student, Professor, Businessman, Athlete etc.

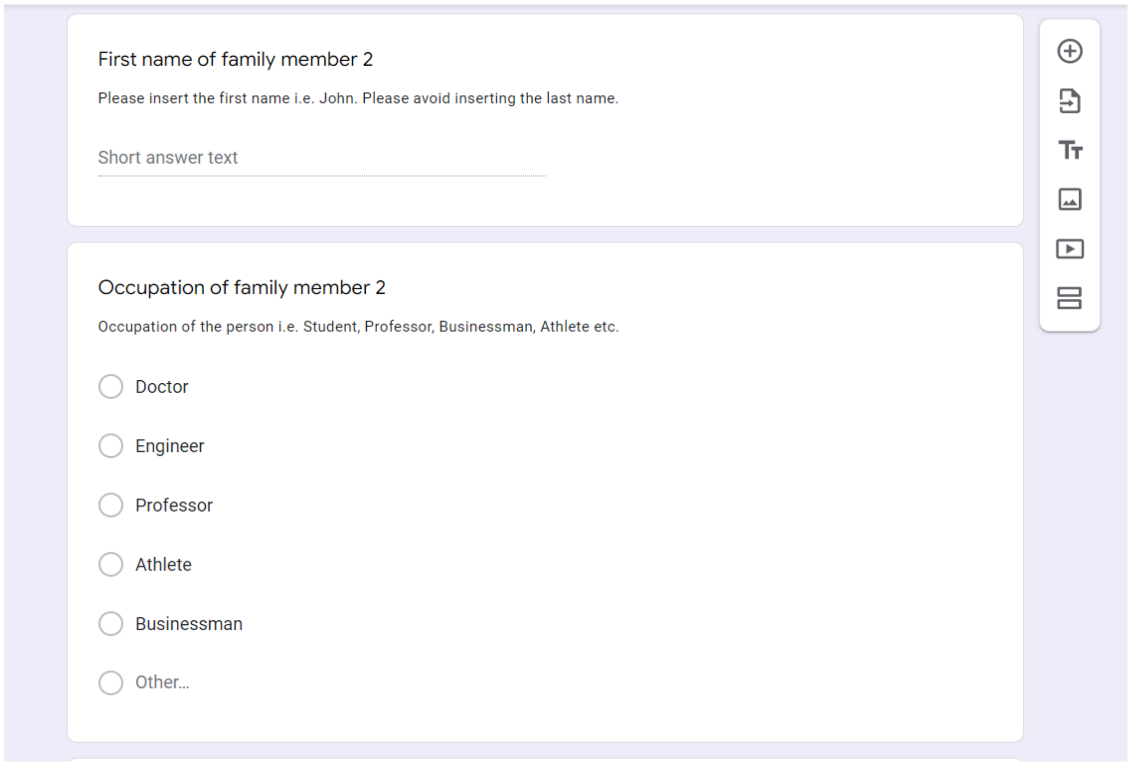
- Doctor
- Engineer
- Professor
- Athlete
- Businessman
- Other...

Family Member 2 photo

Add file

View folder

Figure B.4: Snap shot3 for information field of Google form.


First name of family member 2

Please insert the first name i.e. John. Please avoid inserting the last name.

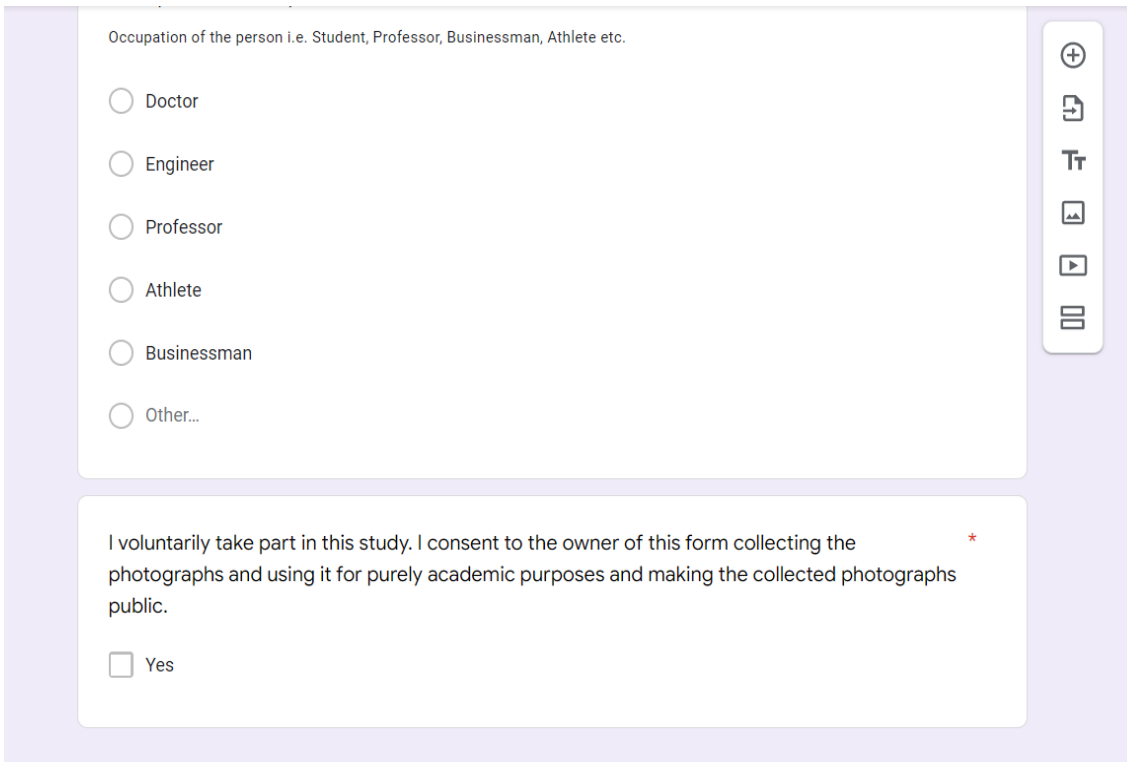
Short answer text

Occupation of family member 2

Occupation of the person i.e. Student, Professor, Businessman, Athlete etc.

- ◯ Doctor
- ◯ Engineer
- ◯ Professor
- ◯ Athlete
- ◯ Businessman
- ◯ Other...

Figure B.5: Snap shot4 for information field of Google form.

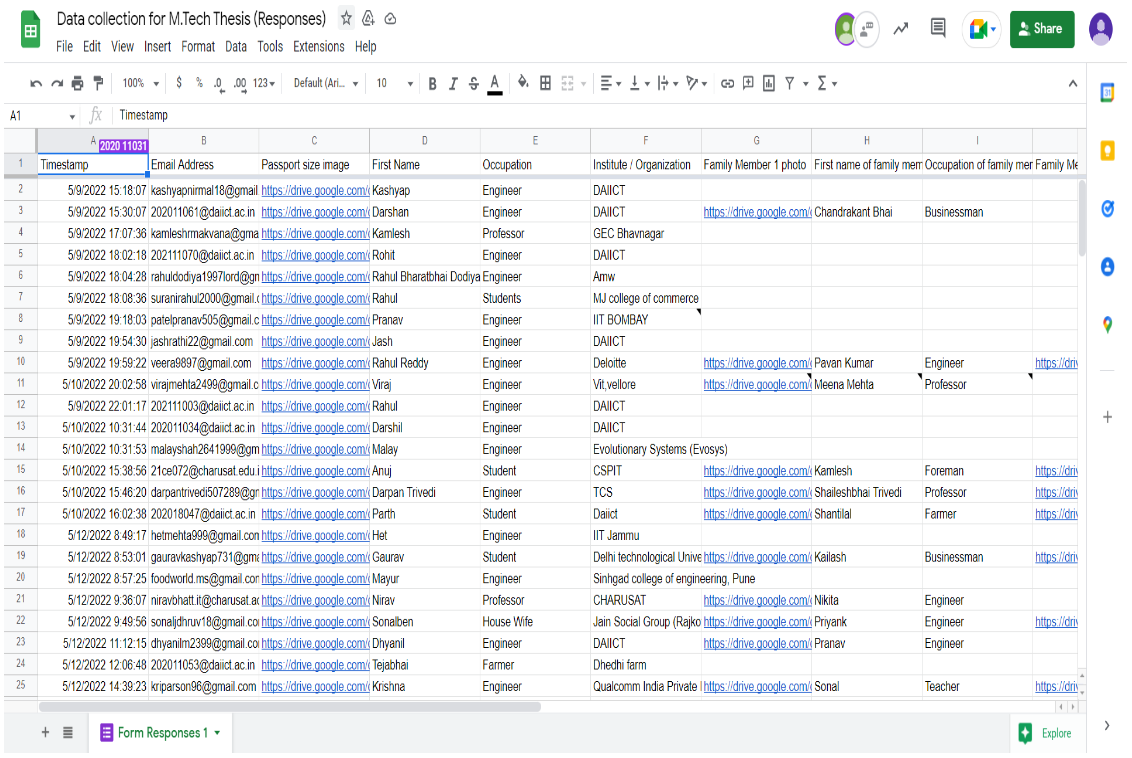

Occupation of the person i.e. Student, Professor, Businessman, Athlete etc.

- ◯ Doctor
- ◯ Engineer
- ◯ Professor
- ◯ Athlete
- ◯ Businessman
- ◯ Other...

I voluntarily take part in this study. I consent to the owner of this form collecting the photographs and using it for purely academic purposes and making the collected photographs public. *

☐ Yes

Figure B.6: Google form Consent snap shot.



Data collection for M.Tech Thesis (Responses)

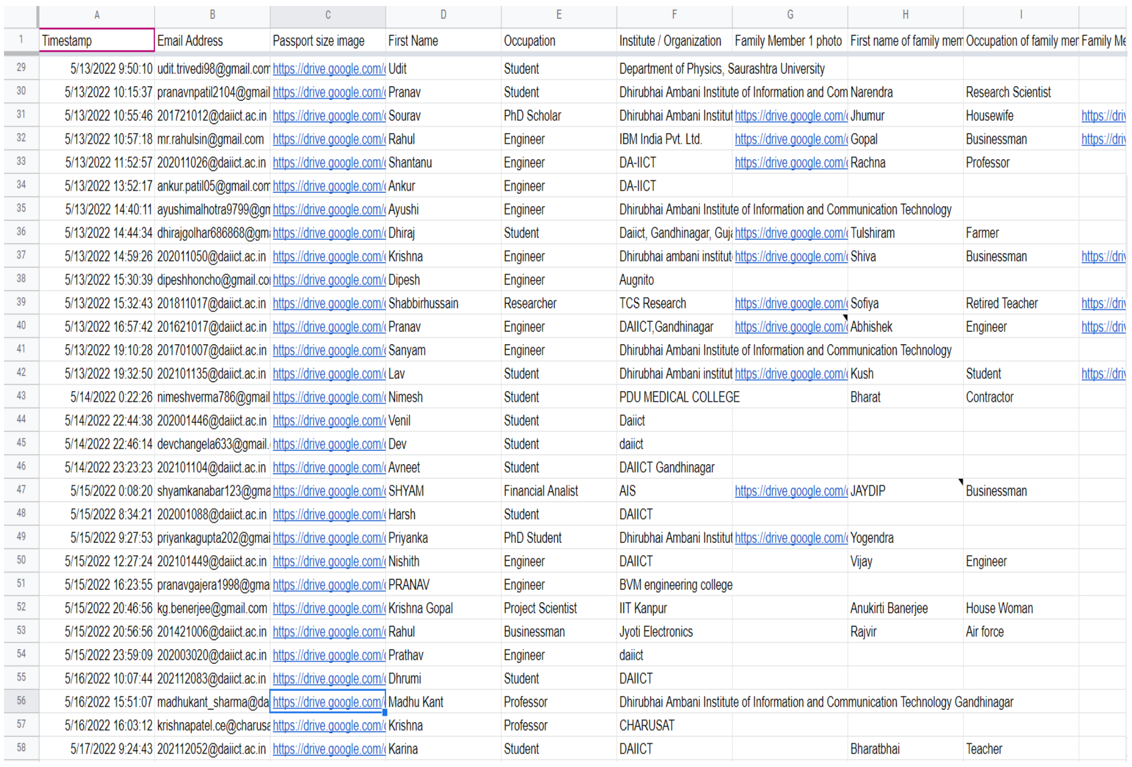
File Edit View Insert Format Data Tools Extensions Help

A1 Timestamp

	A	B	C	D	E	F	G	H	I	J
1	Timestamp	Email Address	Passport size image	First Name	Occupation	Institute / Organization	Family Member 1 photo	First name of family mem	Occupation of family mer	Family Me
2	5/9/2022 15:18:07	kashyapnirmal18@gmail.	https://drive.google.com/	Kashyap	Engineer	DAIICT				
3	5/9/2022 15:30:07	202011061@daiict.ac.in	https://drive.google.com/	Darshan	Engineer	DAIICT	https://drive.google.com/	Chandrakant Bhai	Businessman	
4	5/9/2022 17:07:36	kamleshrmakvana@gma	https://drive.google.com/	Kamlesh	Professor	GEC Bhavnagar				
5	5/9/2022 18:02:18	202111070@daiict.ac.in	https://drive.google.com/	Rohit	Engineer	DAIICT				
6	5/9/2022 18:04:28	rahuldodiya1997lord@gn	https://drive.google.com/	Rahul Bharatbhai Dodiya	Engineer	Amw				
7	5/9/2022 18:08:36	suranirahul2000@gmail.(	https://drive.google.com/	Rahul	Students	MJ college of commerce				
8	5/9/2022 19:18:03	patelpranav505@gmail.c	https://drive.google.com/	Pranav	Engineer	IIT BOMBAY				
9	5/9/2022 19:54:30	jashrathi22@gmail.com	https://drive.google.com/	Jash	Engineer	DAIICT				
10	5/9/2022 19:59:22	veera9897@gmail.com	https://drive.google.com/	Rahul Reddy	Engineer	Deloitte	https://drive.google.com/	Pavan Kumar	Engineer	https://driv
11	5/10/2022 20:02:58	virajmehta2499@gmail.c	https://drive.google.com/	Viraj	Engineer	Vit,vellore	https://drive.google.com/	Meena Mehta	Professor	
12	5/9/2022 22:01:17	202111003@daiict.ac.in	https://drive.google.com/	Rahul	Engineer	DAIICT				
13	5/10/2022 10:31:44	202011034@daiict.ac.in	https://drive.google.com/	Darshil	Engineer	DAIICT				
14	5/10/2022 10:31:53	malayshah2641999@gm	https://drive.google.com/	Malay	Engineer	Evolutionary Systems (Evosys)				
15	5/10/2022 15:38:56	21ce072@charusat.edu.i	https://drive.google.com/	Anuj	Student	CSPIT	https://drive.google.com/	Kamlesh	Foreman	https://driv
16	5/10/2022 15:46:20	darpantrivedi507289@gn	https://drive.google.com/	Darpan Trivedi	Engineer	TCS	https://drive.google.com/	Shaileshbhai Trivedi	Professor	https://driv
17	5/10/2022 16:02:38	202018047@daiict.ac.in	https://drive.google.com/	Parth	Student	Daiict	https://drive.google.com/	Shantilal	Farmer	https://driv
18	5/12/2022 8:49:17	hetmehta999@gmail.con	https://drive.google.com/	Het	Engineer	IIT Jammu				
19	5/12/2022 8:53:01	gauravkashyap731@gma	https://drive.google.com/	Gaurav	Student	Delhi technological Unive	https://drive.google.com/	Kailash	Businessman	https://driv
20	5/12/2022 8:57:25	foodworld.ms@gmail.con	https://drive.google.com/	Mayur	Engineer	Sinhgad college of engineering, Pune				
21	5/12/2022 9:36:07	niravbhatt.it@charusat.ac	https://drive.google.com/	Nirav	Professor	CHARUSAT	https://drive.google.com/	Nikita	Engineer	
22	5/12/2022 9:49:56	sonaljdhruv18@gmail.co	https://drive.google.com/	Sonalben	House Wife	Jain Social Group (Rajko	https://drive.google.com/	Priyank	Engineer	https://driv
23	5/12/2022 11:12:15	dhyanilm2399@gmail.co	https://drive.google.com/	Dhyanil	Engineer	DAIICT	https://drive.google.com/	Pranav	Engineer	
24	5/12/2022 12:06:48	202011053@daiict.ac.in	https://drive.google.com/	Tejabhai	Farmer	Dhedhi farm				
25	5/12/2022 14:39:23	kriparson96@gmail.com	https://drive.google.com/	Krishna	Engineer	Qualcomm India Private I	https://drive.google.com/	Sonal	Teacher	https://driv

Form Responses 1

Figure B.7: Snap shot 1 for responses of Google form.



	A	B	C	D	E	F	G	H	I	J
1	Timestamp	Email Address	Passport size image	First Name	Occupation	Institute / Organization	Family Member 1 photo	First name of family mem	Occupation of family mer	Family Me
29	5/13/2022 9:50:10	udit.trivedi98@gmail.com	https://drive.google.com/	Udit	Student	Department of Physics, Saurashtra University				
30	5/13/2022 10:15:37	pranavnpatil2104@gmail	https://drive.google.com/	Pranav	Student	Dhirubhai Ambani Institute of Information and Com		Narendra	Research Scientist	
31	5/13/2022 10:55:46	201721012@daiict.ac.in	https://drive.google.com/	Sourav	PhD Scholar	Dhirubhai Ambani Institut	https://drive.google.com/	Jhumur	Housewife	https://driv
32	5/13/2022 10:57:18	mr.rahulsin@gmail.com	https://drive.google.com/	Rahul	Engineer	IBM India Pvt. Ltd.	https://drive.google.com/	Gopal	Businessman	https://driv
33	5/13/2022 11:52:57	202011026@daiict.ac.in	https://drive.google.com/	Shantanu	Engineer	DA-IICT	https://drive.google.com/	Rachna	Professor	
34	5/13/2022 13:52:17	ankur.patil05@gmail.com	https://drive.google.com/	Ankur	Engineer	DA-IICT				
35	5/13/2022 14:40:11	ayushimalhotra9799@gn	https://drive.google.com/	Ayushi	Engineer	Dhirubhai Ambani Institute of Information and Communication Technology				
36	5/13/2022 14:44:34	dhirajgolhar686868@gma	https://drive.google.com/	Dhiraj	Student	Daiict, Gandhinagar, Guja	https://drive.google.com/	Tulshiram	Farmer	
37	5/13/2022 14:59:26	202011050@daiict.ac.in	https://drive.google.com/	Krishna	Engineer	Dhirubhai ambani institut	https://drive.google.com/	Shiva	Businessman	https://driv
38	5/13/2022 15:30:39	dipeshhoncho@gmail.co	https://drive.google.com/	Dipesh	Engineer	Augnito				
39	5/13/2022 15:32:43	201811017@daiict.ac.in	https://drive.google.com/	Shabbirhussain	Researcher	TCS Research	https://drive.google.com/	Sofiya	Retired Teacher	https://driv
40	5/13/2022 16:57:42	201621017@daiict.ac.in	https://drive.google.com/	Pranav	Engineer	DAIICT,Gandhinagar	https://drive.google.com/	Abhishek	Engineer	https://driv
41	5/13/2022 19:10:28	201701007@daiict.ac.in	https://drive.google.com/	Sanyam	Engineer	Dhirubhai Ambani Institute of Information and Communication Technology				
42	5/13/2022 19:32:50	202101135@daiict.ac.in	https://drive.google.com/	Lav	Student	Dhirubhai Ambani institut	https://drive.google.com/	Kush	Student	https://driv
43	5/14/2022 0:22:26	nimeshverma786@gmail	https://drive.google.com/	Nimesh	Student	PDU MEDICAL COLLEGE		Bharat	Contractor	
44	5/14/2022 22:44:38	202001446@daiict.ac.in	https://drive.google.com/	Venil	Student	Daiict				
45	5/14/2022 22:46:14	devchangela633@gmail.	https://drive.google.com/	Dev	Student	daiict				
46	5/14/2022 23:23:23	202101104@daiict.ac.in	https://drive.google.com/	Avneet	Student	DAIICT Gandhinagar				
47	5/15/2022 0:08:20	shyamkanabar123@gma	https://drive.google.com/	SHYAM	Financial Analist	AIS	https://drive.google.com/	JAYDIP	Businessman	
48	5/15/2022 8:34:21	202001088@daiict.ac.in	https://drive.google.com/	Harsh	Student	DAIICT				
49	5/15/2022 9:27:53	priyankagupta202@gmai	https://drive.google.com/	Priyanka	PhD Student	Dhirubhai Ambani Institut	https://drive.google.com/	Yogendra		
50	5/15/2022 12:27:24	202101449@daiict.ac.in	https://drive.google.com/	Nishith	Engineer	DAIICT		Vijay	Engineer	
51	5/15/2022 16:23:55	pranavgajera1998@gma	https://drive.google.com/	PRANAV	Engineer	BVM engineering college				
52	5/15/2022 20:46:56	kg.benerjee@gmail.com	https://drive.google.com/	Krishna Gopal	Project Scientist	IIT Kanpur		Anukirti Banerjee	House Woman	
53	5/15/2022 20:56:56	201421006@daiict.ac.in	https://drive.google.com/	Rahul	Businessman	Jyoti Electronics		Rajvir	Air force	
54	5/15/2022 23:59:09	202003020@daiict.ac.in	https://drive.google.com/	Prathav	Engineer	daiict				
55	5/16/2022 10:07:44	202112083@daiict.ac.in	https://drive.google.com/	Dhrumi	Student	DAIICT				
56	5/16/2022 15:51:07	madhukant_sharma@da	https://drive.google.com/	Madhu Kant	Professor	Dhirubhai Ambani Institute of Information and Communication Technology Gandhinagar				
57	5/16/2022 16:03:12	krishnapatel.ce@charusa	https://drive.google.com/	Krishna	Professor	CHARUSAT				
58	5/17/2022 9:24:43	202112052@daiict.ac.in	https://drive.google.com/	Karina	Student	DAIICT		Bharatbhai	Teacher	

Figure B.8: Snap shot 2 for responses of Google form.

CHAPTER C
System outputs

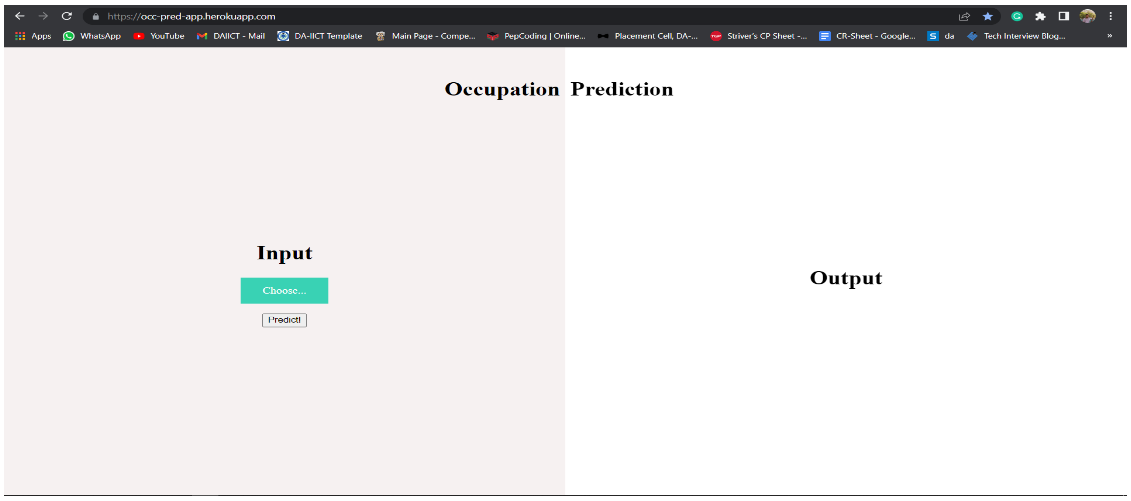
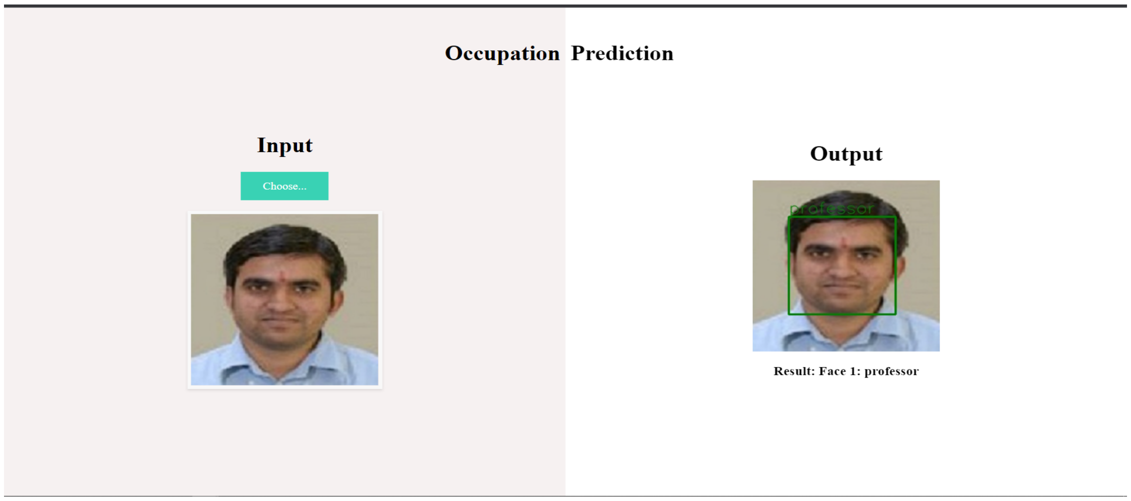


Figure C.1: Occupation Prediction demo using flask



Result: Face 1: professor

Figure C.2: Demonstration of the system.
Occupation Prediction from the face. Here we give professor's image as input. The right side's picture shows the predicted occupation of the input image. Here professor correctly classified as professor.

Figure C.3: QR Code for Github repository.
URL: https://github.com/darshan154/occupation-prediction