AGE VERIFICATION FOR HEALTHY ONLINE ENVIRONMENT PROTECTING CHILDREN AND ADOLESCENTS

Kexin (Alissa) Xiang
Student# 1008140094
kxalissa.xiang@mail.utoronto.ca

Rachel Min Yu
Student# 1008761751
rachelmyl.yu@mail.utoronto.ca

Xiongrui (Jerry) Shi
Student# 1008968253
jerryxr.shi@mail.utoronto.ca

Yubin (Olivia) Zhang
Student# 1008864837
oliviayubin.zhang@mail.utoronto.ca

Colab Link:
https://colab.research.google.com/drive/1xsqwtRxQh1zdiF4SpeXIqP2zWJG8u1F0?usp=sharin

ABSTRACT

This report aims to identify project goals, data processing, results of data processing, baseline model, primary model, and future steps. The purpose of our project is to develop an age verification system capable of recognizing and distinguishing human faces and classifying them into two categories: individuals aged 16 and below, and those above 16. The motivation behind our project stems from the increasing number of minors accessing unrestricted websites and online platforms. To address this issue, our project focuses on verifying the age of individuals attempting to access unrestricted internet information using deep learning models. The dataset, obtained from Kaggle.com, undergoes a data preprocessing phase involving relabeling, resizing, and division into train, validation, and test sets. For image classification and baseline model, we have employed a convolutional neural network (CNN), known for its robust picture classification capabilities. Based on the performance of the baseline model, we have harnessed a pre-trained VGG19 model feature extraction and applied transfer learning on two fully-connected layers. We assess the model's accuracy across our datasets. Additionally, the model is evaluated on self-collected data to experiment its performance in real life scenario.

—-Total Pages: 9

1 INTRODUCTION

In today's digital age, with the abundance of online content, ensuring appropriate age verification has become increasingly crucial. Unrestricted access to the internet poses risks, especially for children who may accidentally come across restricted or inappropriate material. To address this concern, our project focuses on developing an efficient and reliable age verification system using facial recognition technology.

The need of our project is to develop a facial recognition system capable of effectively classifying individuals into two distinct categories based on age —— those above the specified age threshold and those below it —— based on a full face photo of the user (the person attempting to access the website). Figure 1 shows an example of successful classification of age verification given the threshold of 16 years old.

Deep learning presents itself as a reasonable approach for age verification due to its ability to mimic the decision-making process of the human brain, but with enhanced capabilities. The multi-layered architecture of deep learning models enables them to capture intricate facial features and learn intricate representations. This scalability and adaptability make deep learning suitable for age verification.

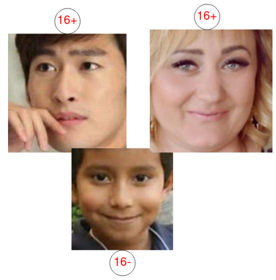

Figure 1: The Project Illustration

2 ILLUSTRATION

Figure 2 presented illustrates the fundamental concept of our age verification model. Initially, we obtained human face data as input. The input image is normalized and resized to a standardized dimension of 200x200 pixels. The processed input then undergoes feature extraction using pre-trained model VGG19 to facilitate accurate classification. Ultimately, the model generates a Boolean output of prediction, where a value of ”false” indicates that the individual is below the age of 16, while a value of ”true” signifies that the person is above the age of 16.

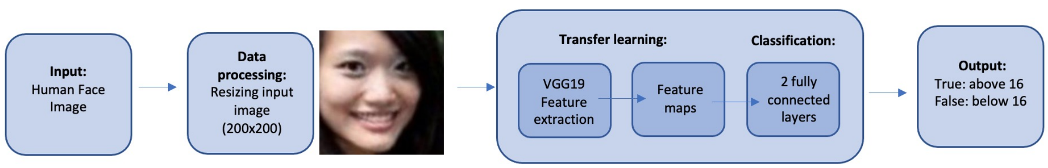

Figure 2: Age Verification Model. Image: Kaggle Facial Age dataset (Rabbi, 2019)

3 BACKGROUND AND RELATED WORKS

Scientists sought to categorize facial images by age, building upon prior research. Early work explored age through facial features like anthropometric models and wrinkles (Kwon & Niels da Vitoria, 1998). While Kwon pioneered feature-based age estimation, Geng et al. (2007) introduced CNNs and long short-term memory networks for improved precision by capturing aging trends. Their method, AGES, enhanced predictions through label distribution learning and modeled aging patterns with a subspace. Moreover, reports like Sai et al. (2014) (2014) outlined age recognition phases involving feature extraction and age estimator training.

Building on the researches, age estimation transitioned from science to public entertainment. One of the examples is selfie cams, which is a popular application featured by functions like age estimation based on a full-face picture, claiming an error range of 2 years. At the same time, age estimation technology has also been used in social media. Age estimation by AI serves as one of the options to verify whether they are over 18 or not on Instagram. This verification will prompt the user to take a video selfie with their full face to further analyze their age based on a model designed by Yoti (yot, 2022). This new testing method run out smoothly for many reporters, but has faced criticism due to concerns about the impact of skin tones on accuracy and issues related to privacy.

Studies have also focused on various flaws of current age detection technologies. In one of such studies, Ganel et al. (2022) suggested that AI tends to overestimate the ages of smiling individuals. By examining these diverse works and critiques, we can gain valuable insights into the development of age estimation methods and understand the challenges and limitations that exist in this field.

4 Data Processing

The dataset for our project was obtained from Kaggle.com's Facial Age dataset (Rabbi, 2019), comprising 99 folders of human faces categorized by age from 1 to 99. Our project aims to predict whether an individual's age is below or above 16 based solely on an image of their face. To achieve this, we will select a subset of the dataset and divide it into two groups: below 16 and above 16.

4.1 Data Cleaning and Organization

Initially, data was sourced from Kaggle.com, focusing on ages 8 to 60 to meet the purpose of our project: protecting adolescents in online environment. This dataset was stored in the "uncleaned_data" folder on Google Drive, which was then mounted onto the Colab file for later usage (Rabbi, 2019). Data preprocessing involved employing `torchvision.datasets.ImageFolder()` to load data from the "uncleaned_data" folder, through which images were resized to 200x200 RGB and normalized to standardize dimensions. The resultant data set, named "uncleaned_data_set", was loaded into "uncleaned_data_loader" via `torch.utils.data.DataLoader()`.

Subsequent processing employed a for loop iterating through "uncleaned_data_loader" for each image and its label. As shown in Figure 3, This loop categorized images by age into "below_16" and "over_16" folders. Age tags and "below/over_16" indications were assigned as labels. To balance data, the loop included counters for each class, ensuring roughly 1500 images in each. These organized folders were stored in the "cleaned_dataset" directory. "Below_16" contained 1271 image tensors, while "over_16" held 1500 image tensors.

```
n = 1
m= 1

for img, label in uncleaned_data_loader:
  if(m>1500 and n>1500):
    break

  if (int(classes[label]) <= 16 and n <= 1500):
    torch.save(img.squeeze(), folder + 'below_16/'+ str(n) +
               '_' + str(classes[label]) +
               '_' + 'below_16' + '.tensor')
    n += 1

  elif(int(classes[label]) > 16 and m <= 1500):
    torch.save(img.squeeze(), folder + 'over_16/'+ str(m) +
               "_" + str(classes[label]) + '_' +
               'over_16'+ '.tensor')
    m += 1
```

Figure 3: Sorting Process

4.2 Data Splitting and Our Own Test Set

To further prepare the data, we loaded the newly sorted data set, `cleaned_dataset`, using `torchvision.datasets.DatasetFolder()` from the `cleaned_dataset` folder's path. Next, we randomly split the cleaned data set into training, validation, and testing sets using `torch.utils.data.random_split()`. The training set encompassed 75% of the cleaned data set, while the validation and test sets each accounted for 15%. These three data sets were then loaded into `train_loader`, `val_loader` and `test_loader` which will be utilized in training and testing. We displayed some sample data from the `train_loader` to ensure that our data set is randomized and labeled correctly.

To test the model's performance in a real life scenario, we gathered our own testing data. We manually collected our own and our friends' photos, including childhood pictures, given that our project aims to predict whether a person is above or below 16. The new testing data is resized to 200x200 pixels and normalized with the human face centered for the model to process and predict. This data set contains 102 images, 51 for each class.

```python
for images, labels in train_loader:
    # since batch_size = 1, there is only 1 image in `images`
    image = images[0]
    image = image.clip(0,1)
    label = labels[0]
    # place the colour channel at the end, instead of at the beginning
    img = np.transpose(image, [1, 2, 0])
    # normalize pixel intensity values to [0, 1]
    img = img

    # Display the image
    ax = fig.add_subplot(3, 5, k+1)
    ax.axis('off')
    ax.imshow(img)
    ax.set_title(classes[label])  # Use label index to get the corresponding class

    k += 1
    if k >= 15:
        break
```

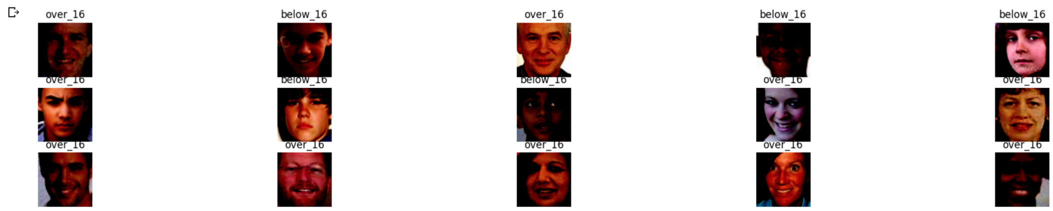

Figure 4: Visualization of Cleaned Training Data. Image: Normalized and Resized Kaggle Facial Age Image (Rabbi, 2019)

4.3 Transfer Learning Feature Maps

Later in the model construction process, we applied transfer learning onto an ANN classifier, using a pre-trained VGG19 model to extract features from the human face images and then pass through the classifier to make predictions. Thus, feature maps are obtained and saved for images in the train, validation, and test sets using `vgg19.features(image)`. These maps are later used in training the classifier.

```python
def get_vgg_feature(dataset, category):
  loader = torch.utils.data.DataLoader(dataset, batch_size=batch_size,
                                        num_workers=num_workers, shuffle=True)
  n = 0
  folder = vgg19_path + '/' + category + '/'

  # save features to folder as tensors
  # the print are for checking how much images are completed in the transformation
  for img, label in loader:
      features = vgg19.features(img)
      features_tensor = torch.from_numpy(features.detach().numpy())
      torch.save(features_tensor.squeeze(0),
                 folder + '/' + str(classes[label]) + '/' + str(classes[label]) + '_' + str(n) + '.tensor')
      n += 1
```

Figure 5: VGG19 Feature Extraction

5 Architecture

The final model is built upon CNN structure. CNNs are particularly effective in image processing tasks, making them suitable for facial recognition. They can extract relevant features from images by using convolutional layers, pooling layers, and non-linear activation functions. Designed to learn and recognize patterns and features in images automatically, CNNs can be used to learn representations that are discriminative for age classification.

The team chooses to use VGG19 to do the transfer learning in our final model. VGG19 is a complex CNN model. In CNN, built-in convolutional layer reduces the high-dimensional image to a low-dimensional representation without losing important information (Lang, 2021). This is particularly helpful in image processing and face recognition. Moreover, the fully-connected layer provides a more accurate connection when dividing a large image into some sub-images (Lang, 2021), which is the most common technique for image processing, and the sub-images are linked again through the CNN to carry out the classification. It consists of 16 convolutional layers, 5 pooling layers, which is 21 layers in total. Due to more layers added and pre-trained features. Our primary model could have better performance in image processing than the baseline model.

By connecting the pre-trained VGG19 features to the team's own classifier. The image will first pass through the VGG to reduce its dimensions from 200*200*3 to 6*6*512 while keeping its important features. Then, it will pass to our classifier's first layers, and classify into 100 classes according to their features. Finally, the second fully connected layer will perform the classification task by reducing 100 to 2 classes that are over or under 16.

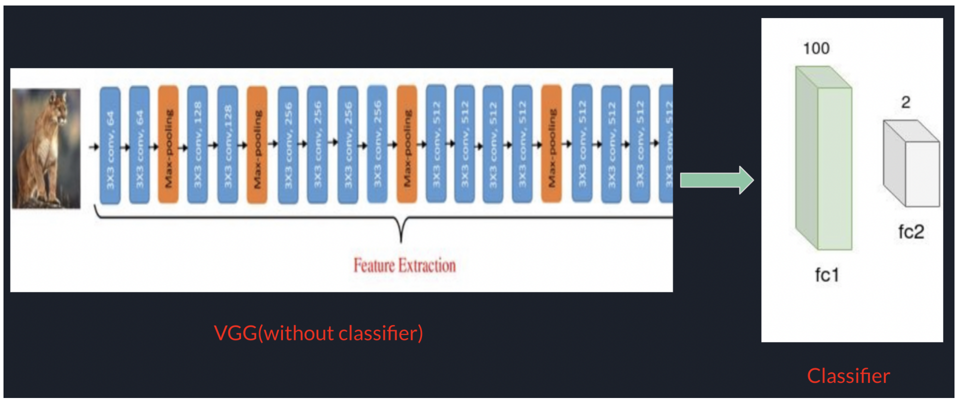

Figure 6: Primary Model

6 Baseline Model

The architecture of our baseline model, as depicted in Figure 7, consists of 2 convolutional layers, 2 max-pooling layers, and 2 fully-connected layers. This CNN model is derived from the one used in Lab 3, as both the lab and our project involve extracting features from images and classifying them, making the CNN architecture well-suited for this problem. For training and testing purposes, we will utilize the same dataset for both the baseline and primary models.

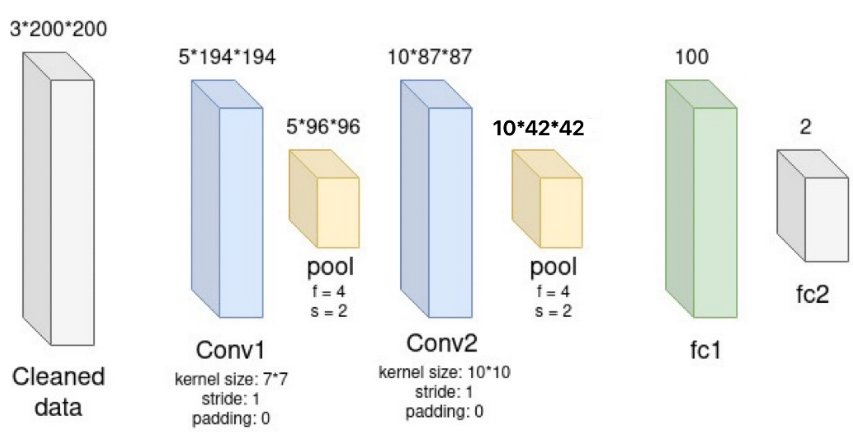

Figure 7: Baseline Model

7 RESULTS

7.1 QUANTITATIVE RESULTS

To measure and compare the performance of our models, we use the metric of accuracy as shown in Figure 8, which measures the proportion of correctly classified samples out of the total number of samples in the dataset.

During the training progress, we tracked the training accuracy and validation accuracy after each epoch (Figure 9), representing how well the model performs on the training and validation datasets. As the model was trained for multiple epochs, we observed an improvement in both accuracy, indicating that the model was effectively learning from the data.

```python
def get_accuracy(model,train_loader,val_loader,train=False):
    if train:
        data = train_loader
    else:
        data = val_loader

    correct = 0
    total = 0
    for imgs, labels in data:


        #############################################
        #To Enable GPU Usage
        if use_cuda and torch.cuda.is_available():
          imgs = imgs.cuda()
          labels = labels.cuda()
        #############################################


        output = model(imgs)

        #select index with maximum prediction score
        pred = output.max(1, keepdim=True)[1]
        correct += pred.eq(labels.view_as(pred)).sum().item()
        total += imgs.shape[0]
    return correct / total
```

Figure 8: Get Accuracy Function

```
CUDA is available!  Training on GPU ...
Epoch: 1 | Training Accuracy: 0.6867179980750722 | Validation Accuracy: 0.6069364161849711
Epoch: 2 | Training Accuracy: 0.690567853705486 | Validation Accuracy: 0.6213872832369942
Epoch: 3 | Training Accuracy: 0.8214629451395573 | Validation Accuracy: 0.7630057803468208
Epoch: 4 | Training Accuracy: 0.8402309913378249 | Validation Accuracy: 0.7774566473988439
Epoch: 5 | Training Accuracy: 0.8537054860442733 | Validation Accuracy: 0.8236994219653179
Epoch: 6 | Training Accuracy: 0.8695861405197305 | Validation Accuracy: 0.8208092485549133
Epoch: 7 | Training Accuracy: 0.8474494706448508 | Validation Accuracy: 0.7832369942196532
Epoch: 8 | Training Accuracy: 0.8960538979788258 | Validation Accuracy: 0.8265895953757225
Epoch: 9 | Training Accuracy: 0.8926852743022137 | Validation Accuracy: 0.815028901734104
Epoch: 10 | Training Accuracy: 0.8633301251203079 | Validation Accuracy: 0.7976878612716763
Epoch: 11 | Training Accuracy: 0.8445620789220404 | Validation Accuracy: 0.7890173410404624
Epoch: 12 | Training Accuracy: 0.9143407122232916 | Validation Accuracy: 0.846820809248555
Epoch: 13 | Training Accuracy: 0.8893166506256015 | Validation Accuracy: 0.8121387283236994
Epoch: 14 | Training Accuracy: 0.940808469682387 | Validation Accuracy: 0.8265895953757225
Epoch: 15 | Training Accuracy: 0.9504331087584216 | Validation Accuracy: 0.8526011560693642
Epoch: 16 | Training Accuracy: 0.9706448508180944 | Validation Accuracy: 0.8497109826589595
Epoch: 17 | Training Accuracy: 0.971126082771896 | Validation Accuracy: 0.8236994219653179
Epoch: 18 | Training Accuracy: 0.9826756496631376 | Validation Accuracy: 0.8554913294797688
Epoch: 19 | Training Accuracy: 0.9817131857555341 | Validation Accuracy: 0.8352601156069365
Epoch: 20 | Training Accuracy: 0.9682386910490857 | Validation Accuracy: 0.8121387283236994
Epoch: 21 | Training Accuracy: 0.9942252165543792 | Validation Accuracy: 0.8439306358381503
Epoch: 22 | Training Accuracy: 0.9903753609239654 | Validation Accuracy: 0.846820809248555
Epoch: 23 | Training Accuracy: 1.0 | Validation Accuracy: 0.8439306358381503
Epoch: 24 | Training Accuracy: 0.9995187680461982 | Validation Accuracy: 0.861271676300578
Epoch: 25 | Training Accuracy: 1.0 | Validation Accuracy: 0.8497109826589595
Epoch: 26 | Training Accuracy: 1.0 | Validation Accuracy: 0.838150289017341
Epoch: 27 | Training Accuracy: 1.0 | Validation Accuracy: 0.846820809248555
Epoch: 28 | Training Accuracy: 1.0 | Validation Accuracy: 0.8497109826589595
Epoch: 29 | Training Accuracy: 1.0 | Validation Accuracy: 0.8497109826589595
Epoch: 30 | Training Accuracy: 1.0 | Validation Accuracy: 0.8554913294797688
```

Figure 9: Sample Tracking

Figure 10 and Figure 11 below visually illustrates the relationship between the training accuracy and loss curves with the number of iterations for the baseline model. Throughout the training process, the training loss gradually decreases, demonstrating that the model is minimizing its errors and getting closer to accurate predictions. We achieved a final impressive training accuracy of 100%, indicating its ability to perfectly classify the training data. However, this high training accuracy raises concerns about possible overfitting, as it might be too tailored to the training data and may not generalize well to new, unseen examples. In Figure 11, the curve is noisy and the validation accuracy plateaued in the later epochs. To assess its generalization capability, the baseline model obtained a validation accuracy of 85.55% and a testing accuracy of 85.3%. Although these values are lower than the training accuracy, they still demonstrate reasonable performance on unseen data.

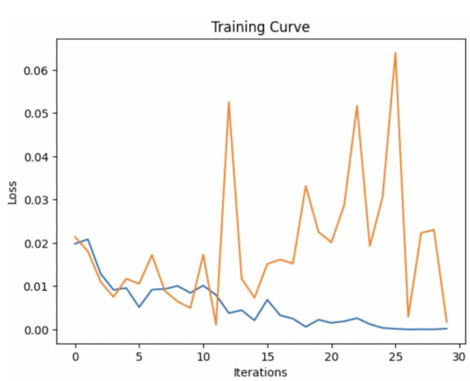

Figure 10: Loss of Baseline Model

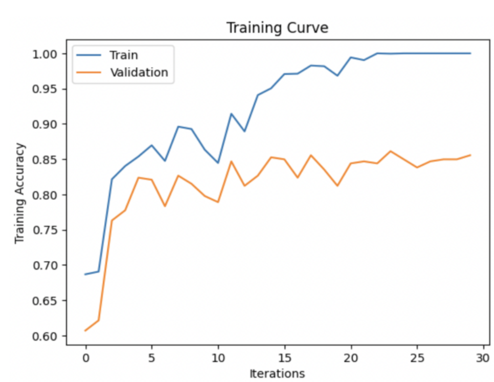

Figure 11: Accuracy of Baseline Model

On the other hand, the primary model, which uses transfer learning with a pre-trained VGG feature extractor, achieved a training accuracy of 96.49%. This is shown in Figure 13 which shows the learning curve of first 10 epochs with an upward trend. This result is slightly lower than the baseline model's training accuracy but remains quite high, indicating that the primary model effectively captures important patterns and features from the pre-trained VGG model. The primary model's validation accuracy of 87.28% and testing accuracy of 86.45% demonstrate strong generalization performance, which is promising for a real-world age verification system. By using transfer learning, the primary model benefits from the pre-trained VGG feature extractor's ability to recognize complex patterns in images, leading to improved generalization to unseen data.

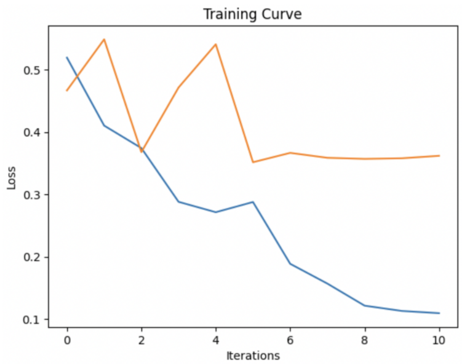

Figure 12: Loss of Primary Model

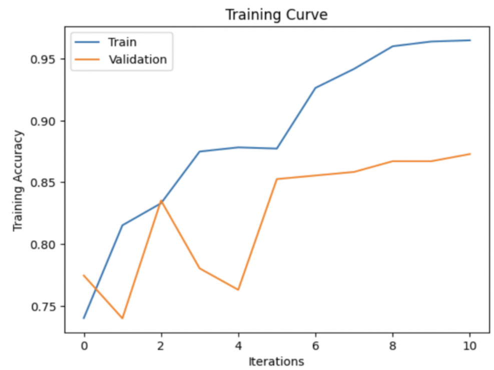

Figure 13: Accuracy of Primary Model

7.2 Qualitative Results

As part of our evaluation, we took a selection of several images from our test set to visually analyze how well our model performs. This allowed us to gain a better understanding of its capabilities in real-world scenarios. What we found is that our model demonstrates a high degree of robustness, meaning it's able to make accurate predictions across a variety of situations, irrespective of changes in scale or orientation of the input images. In Figure 14, you can observe the outcome of this analysis. The model effectively classified all the faces presented in the images, correctly determining the age category of each individual. This success is indicative of the model's ability to discern and recognize facial features and patterns that are essential for accurate age prediction. The model's consistent performance across various instances underscores its reliability and highlights its potential for real-world applications.

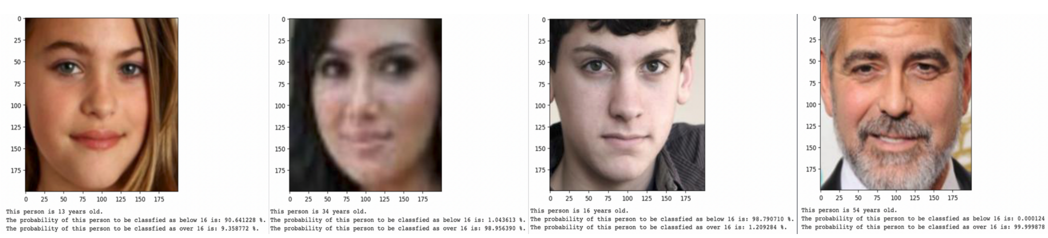

Figure 14: Sample Predictions of Primary Model Using Data from Kaggle

However, our qualitative analysis also sheds light on certain limitations of our model. Specifically, we realized the model's accuracy encounters a challenge when dealing with the age range of teenagers. We achieved an accuracy of 82.46% for "below_16" class and 88.07% for "over_16" class for data sourced from the Kaggle's test data set. This phenomenon can be attributed to the inherent similarity in facial features among individuals within the age group 13 to 19, where teenagers often share comparable facial characteristics. The model could potentially misclassify a 19-year-old who possesses youthful facial attributes, like a lighter skin tone or a smiling expression, as someone younger than 16. Conversely, a 14-year-old with more mature facial characteristics, such as a darker skin tone and a non-smiling face, might be inaccurately categorized as older than 16. As a result, the model's performance might perform less accurately this particular age group compared to others, where age-related facial changes are more distinctive.

7.3 Model Evaluation on New Data

To double ensure the accuracy of our results, we've taken care to evaluate our models on our own new, unseen data. We collected a separate set of real-life photos, distinct from our training and validation data, to form an independent test dataset. This dataset, not used during model training or tuning, serves as an unbiased measure of our models' real-world performance. Using our primary model, we achieved an overall accuracy of 73.47%, with an 83.67% accuracy for "below_16" and a 63.27% accuracy for "over_16". We observed a reduce in accuracy in the "over_16" class, which can also be explained by the model's inaccuracy in the age range from 13 to 19. Since we collected the data ourself, the photos we used for the "over_16" class are photos taken when we and our friends were around 17 and 18 so that we may display a more youthful facial features in the photos, causing inaccuracy in the model's performance in the "over_16" class. Figure 15 illustrates the sample predictions using our four pictures. Based on the prediction, we are all classified as below 16.

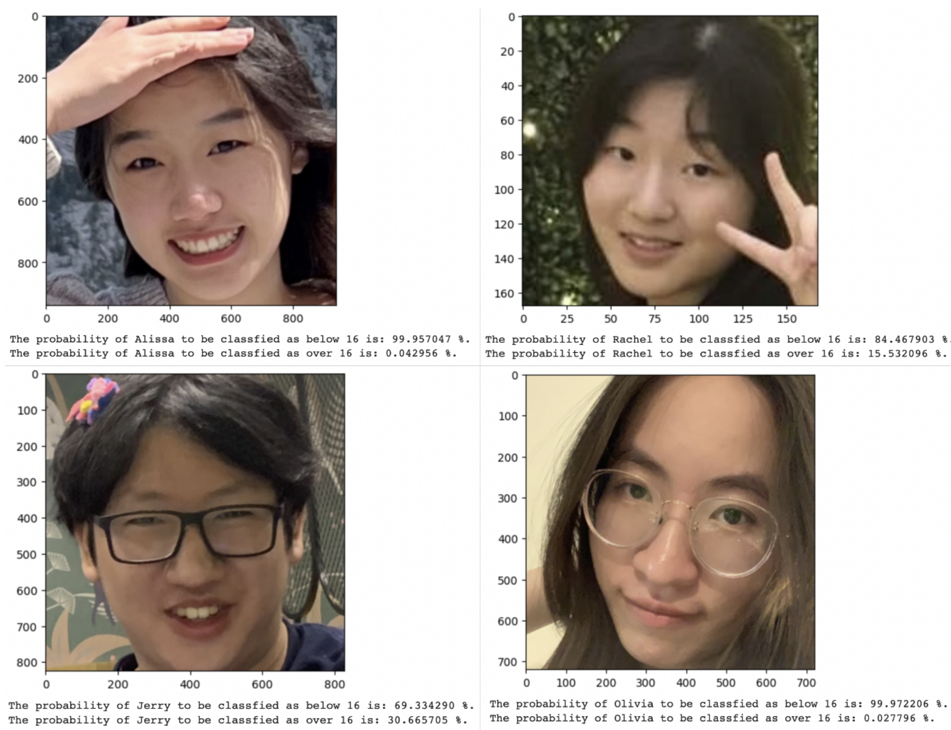

Figure 15: Sample Predictions of Primary Model Using Self-Collected Data

8 Discussion

We believed that both baseline and primary model perform well on self-collected data, because they both achieved an accuracy surpassing 70%. This signifies that these models have acquired crucial features enabling accurate predictions, rather than relying on random guesses. While both baseline and primary models show potential for age detection based on facial recognition, the primary model performed well compared to the baseline model. The baseline model achieved remarkably high training accuracy, a possible indication of overfitting, as it struggled to generalize to new, unseen data. The primary model, leveraging transfer learning, exhibited a slight drop in training accuracy but showcased impressive validation and testing accuracies, highlighting its ability to effectively generalize. The primary model undergoes training for considerably shorter epochs compared to the baseline model. It demonstrates that transfer learning is effective in reducing training time and extracting deeper features in the data set so that it is easier to train the classifier.

Notably, the primary model's success in recognizing faces and predicting ages across various individuals is a testament to the power of transfer learning. Its performance underscores the advantage of leveraging pre-trained neural network architectures, enabling the model to capture intricate facial patterns that contribute to accurate age prediction. This outcome aligns with the idea that features learned from a large, diverse data set can be effectively transferred to specialized tasks. However, the primary model's challenges in accurately classifying teenagers due to subtle facial feature variations underscore the inherent complexities of age estimation from facial images. Teenagers often exhibit similar facial attributes, leading to confusion in the model's predictions. This finding emphasizes the importance of refining the model's training data set to include a more comprehensive representation of diverse age groups, particularly those with subtle age-related variations.

9 Ethical Consideration

The primary ethical challenge will be encountered during the model utilization is the preservation of user portrait rights and privacy. Given the nature of our face recognition technology, it necessitates the incorporation of human facial data as input. In this process, it is difficult to predict whether the user will upload unauthorized photos, resulting in infringement of other people's portrait rights and breach of privacy, among other concerns.

To address this issue, we implemented stringent measures in data collection and dataset selection. All the data and photos are employed are from open-source repositories on the Internet, specially the Kaggle database (Rabbi, 2019). The facial and age data are used with the permission of the user. Therefore, we ensure that no privacy violations occur throughout the data training and testing processes.

10 Project Difficulty

One of the difficulties in our project existed within the limited availability of high-quality labeled age-specific facial image datasets. Also, as we look into our predictions, we find that age-related changes in facial features are subtle and can be influenced by various factors such as lighting (yot, 2023), pose, and expressions (Ganel et al., 2022), which also posed difficulties for our model to perform accurate prediction. Therefore, obtaining a comprehensive dataset that covers a wide range of ages, ethnicities, and conditions was challenging. This scarcity of diverse and accurate training data made it difficult for our models to learn and generalize effectively. Despite this challenge, our models demonstrated a commendable ability to extract meaningful age-related features from the available data, showcasing their adaptability and robustness in handling data limitations. By using transfer learning (VGG19, which is a pre-trained model with great complexity), we obtain a stable result compared to the baseline model.

11 Conclusion

In summary, based on the difficulties we have faced, our model can be further improved. We used VGG19 as the pre-trained transfer learning model for feature extraction. However, VGG19 is trained on ImageNet which may not be ideal for human faces recognition. Therefore, for further improvements, we can utilize pre-trained model specialized in human faces recognition to mitigate existing inaccuracy in the teenager demographic. Furthermore, we will try data augmentation to balance the data scarcity in the teenager age group, expanding the model's training data in the age group, thus learning more age-related facial feature to reduce its inaccuracy in this particular age group.

References

Age verification for social media, 2022. URL `https://www.yoti.com/social-media/`.

Age verification for social media, 2023. URL `https://www.yoti.com/wp-content/uploads/Yoti-Age-Estimation-White-Paper-March-2023.pdf`.

Tzvi; Ganel, Carmel Sofer, and Melvyn A Goodale. Biases in human perception of facial age are present and more exaggerated in current ai technology, 2022. URL https://pubmed.ncbi.nlm.nih.gov/36581653/.

Xin; Geng, Zhi-Hua Zhou, and Kate Smith-Miles. Automatic age estimation based on facial aging patterns, 2007. URL https://ieeexplore.ieee.org/abstract/document/4359348.

Young H Kwon and Lobo Niels da Vitoria, 1998. URL https://www.sciencedirect.com/science/article/pii/S107731429790549X.

Niklas Lang. Breaking down convolutional neural networks: Understanding the magic behind image recognition, 2021. URL https://towardsdatascience.com/using-convolutional-neural-network-for-image-classification-5997bfd0ede4#:~:text=The%20Convolutional%20Neural%20Network%20(CNN,suited%20for%20this%20use%20case.

Fazle Rabbi. Faical age. https://www.kaggle.com/datasets/frabbisw/facial-age, 2019.

Phyo-Kyaw; Sai, Jian-Gang Wang, and Eam-Khwang Teoh. Facial age range estimation with extreme learning machines, 2014. URL https://www.sciencedirect.com/science/article/pii/S0925231214011400.

[1]

[1]The reference list is organized using iclr2022_conference format.