# Age Estimation via Image Input

Kelan Wu, Benjamin Guzman, Simrat Clair, Benjamin Key

December 2024

## 1 Abstract

While searching for potential problems that could be solved with artificial intelligence, we came up with an idea to create a model that will try to estimate a person's age through facial imaging. The concept has the potential for many different applications and provided us with a clear scope to work with. In order to build a model to solve this problem, we researched the possible models until we settled on the Convolutional Neural Network, which had the potential to process and analyze images one at a time to produce its estimates. We located a dataset with over 20 thousand face images to train the CNN on. After creation of our own model and implementation of other similar models created online, we came across results that were more mixed than we had anticipated. While some of this had to do with our own experience and resources available, we also learned that trying to guess a person's exact age is incredibly difficult even for really advanced AI models. [1]

## 2 Introduction

Age estimation from facial images is a challenging problem. Although humans can often approximate someone's age based on their facial features, replicating this capability with artificial intelligence requires extensive computational resources, robust models, and diverse datasets.

The primary challenge in age estimation lies in the inherent variability in human faces. Genetics, lifestyle, environmental exposure, and image conditions (e.g., lighting, angles, and occlusions) introduce complexities that make precise age prediction difficult. Despite these challenges, advancements in machine learning, particularly with Convolutional Neural Networks (CNNs), have enabled more accurate and efficient solutions for age estimation tasks.

[1] Kelan: Discussed potential models to use. Implemented and trained base CNN model. Wrote introduction, methods, and cnn experiment sections. 25%. Benjamin G: Discussed possible models to compare with. Implemented and trained VGG16. Wrote about dataset and VGG16 results. 25%. Simrat: Discussed possible models to experiment with. Implemented and trained ResNet50. Wrote the background, conclusion, and the ResNet50 results. 25%. Benjamin K: Identified and clarified the problem. Implemented and trained Xception model. Wrote the abstract and Xception results. 25%.

In this project, we aimed to design, implement, and evaluate a system capable of estimating a person's age based on a facial image. Leveraging the UTKFace dataset and TensorFlow, we explored several CNN architectures—including a Basic CNN, VGG16, Xception, and ResNet50—to identify the most effective approach for this problem. Our work seeks to contribute to the growing body of research in age prediction by analyzing and comparing the strengths and limitations of each architecture.

This paper presents an overview of the methods, experiments, results obtained during this study, and insights gained from the challenges encountered throughout the process.

# 3 Background

Age detection from facial images is a crucial task in computer vision with applications in criminal investigations, disaster victim identification, and demographic analysis. Accurately estimating a person's age can help law enforcement narrow down suspects, assist in locating missing individuals, and enable businesses to tailor services based on age demographics.

Traditional methods relied on handcrafted features and machine learning algorithms, which often struggled with variations in facial expressions and lighting. Recent advancements utilize Convolutional Neural Networks (CNNs) like VGG16, Xception, and ResNet50, which automatically extract complex features from images. Our project builds on these approaches by comparing a custom-built Basic CNN with these pre-trained models to evaluate their effectiveness in age prediction. This comparative analysis aims to identify the strengths and limitations of each architecture in accurately estimating age from facial images.

# 4 Data

The dataset that we used was the UTKFace dataset. It is a large-scale face dataset with long age span, with a range from 0 to 116. It consists of over 20,000 face images with annotations of age, gender, and ethnicity. For the purposes of our project, only the age annotation was used. The images cover large variation in pose, facial expression, illumination, occlusion, resolution, etc. The dataset contained two version of each image. One half had the original photos, where the other half were cropped version to make the face the entire picture.

This dataset was chosen thanks to the steps the original creators adding the cropped face to make the reprocessing easier. For the training of all the models, we used the cropped versions to help to get the best out of the models. For validation and testing, the original picture were shown to the models.

The distribution of the ages in the photos are shown below in figure 1. The bars are grouped in blocks of 5 years. For example, 1-5 is represented by a bar, followed by 6-10 and 11-15. The most common age range is 26-30. This is likely as a result of the natural distribution of publicly available pictures. People in

that age range would have been in college when Facebook was getting started and they would have been living their whole lives with digital cameras. They should have the most experience with putting their pictures online. There is also a local max at 1-5 years old. This is probably the result of parent taking a lot of photos of their babies and sharing them.

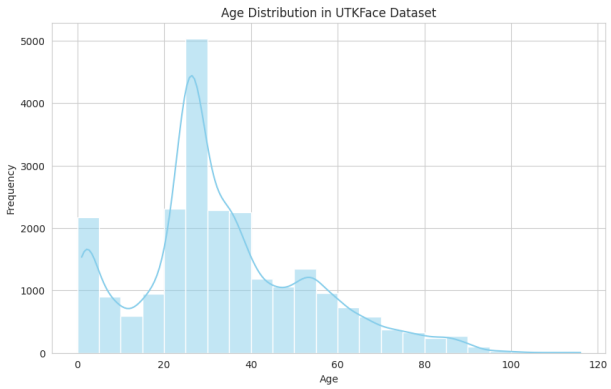


Figure 1: Real vs. Predicted Age

# 5 Method

## 5.1 Pre-processing

For pre-processing, the dataset contains images named with the format where the age is embedded as the first part of the filename. Each file's age is extracted by splitting the filename at the underscore and converting the first part to an integer.

Afterwards, images are read using OpenCV and resized to a resolution of 128 x 128 pixels to ensure consistency across the dataset. Then pixel values are normalized to a range of [0,1] by dividing the image arrays by 255.0. This helps improve the efficiency of model training. Finally, the dataset is divided into training and testing sets using an 80/20 split.

## 5.2 Model Training

For this experiment we are using four different Convolutional Neural Network (CNN) architectures:

1. Basic CNN: A basic architecture for baseline evaluation
2. VGG16
3. Xception
4. ResNet50

We configured each model to have a batch size of 32 and set it to run 20 epochs.

# 6 Experiments

## 6.1 Basic CNN

### 6.1.1 Implementation

For our basic CNN, we imported the Sequential Model from TensorFlow, which lets us stack layers in a linear sequence. We implemented several layers:

- First layer: 32 filters of size 3 x 3
- Second layer: 64 filters of size 3 x 3
- Third layer: 128 filters of size 3 x 3
- Flatten
- Dense Layer: 128 Neurons
- Dropout: 0.5
- Dense Layer: 64 Neurons
- Output

### 6.1.2 Results

After running the model, our results are:


Figure 2: Real vs. Predicted Age

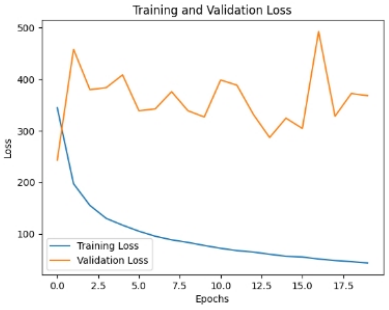


Figure 3: Training and Validation Loss

- Test Loss (MSE): 367.7844846328125
- Test MAE: 15.561653137207031

## 6.2 Xception

### 6.2.1 Implementation

For Xception, we downloaded the weights from Tensorflow to replicate the model, and then trained a fully connected layer on top to make it output a single numerical value. Xception is known for its implementation of depth-wise separable convolution layers.

### 6.2.2 Results

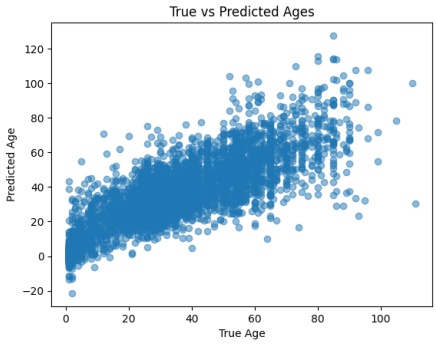


Figure 4: Real vs. Predicted Age

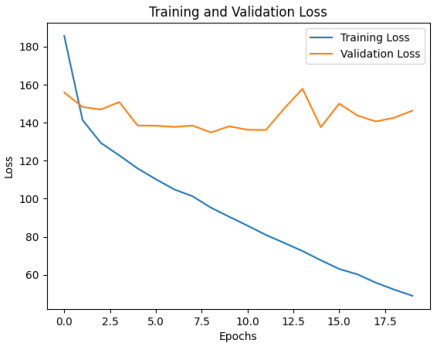


Figure 5: Xception Training and Validation Loss

The results with this version of the model came out with the following statistics:

- Test Loss (MSE): 146.2517852783203
- Test MAE: 8.763976097106934

Since Xception was designed for image classification into a group of 1,000 different categories, it would likely require additional processing power and fine-tuning to improve these rates. However, with the current implementation, it performs much better than our basic CNN model with the same amount of training.

## 6.3 VGG16

### 6.3.1 Implementation

Another model that we used to compare the results of our model is the VGG 16. VGG, named after the research team Visual Geometry Group at Oxford, was developed in 2015. It is a famously deep CNN, with 16 layers. It has 13 convolution layers with ReLu, 3 fully connected layers with ReLu, and finishes off with softmax. It is popular for being simple and for having good performances

with computer vision projects like image recognition. The model takes an input of (224, 224, 3). The first convolutional layer starts off with 64 3x3 filters. The pooling layers use max pulling with a size of 2x2 and stride of 2. By the time the flattening occurs, the resulting a 7x7x512 feature map turns into a vector of size 25088. There are two possible outputs by VGG 16, a class score and a bounding box. We used the class score to approximate age. The bounding box is only good for object localization.

### 6.3.2 Results

After running the model, our results are:


Figure 6: VGG16: Real vs. Predicted Age

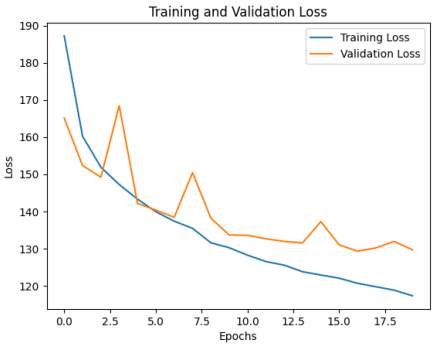


Figure 7: VGG16: Training and Validation Loss

- Test Loss (MSE): 129.6497039794922
- Test MAE: 8.604641914367676

## 6.4 ResNet50

### 6.4.1 Implementation

For the ResNet50 architecture, we loaded a pre-trained model (on ImageNet) and first trained a new dense head for age regression with the base layers frozen. After establishing a baseline, we unfroze the top layers of ResNet50 and fine-tuned them using a reduced learning rate (e.g., 1e-5) to adapt the higher-level filters to age estimation. Early stopping and learning rate reduction ensured stable convergence.

### 6.4.2 Results

ResNet50 showed further improvement. After fine-tuning, we achieved approximately:

- Test MAE: $\sim 13.85$

- Test MSE: $\sim 347.82$

While not perfect, this performance indicates that the model can approximate age to within a certain threshold. The scatter plot of True Age vs. Predicted Age showed a general upward trend, though some clustering and variance remained, especially in higher age ranges.


Figure 8: ResNet50: Real vs. Predicted Age Scatter Plot

# 7 Conclusion

In this project, we evaluated four different models for age estimation from the UTKFace dataset: a Basic CNN, VGG16, Xception, and ResNet50. Our baseline Basic CNN struggled, achieving a high MAE (15.56) and MSE (367.78). Incorporating transfer learning significantly improved results. VGG16 emerged as the top performer with an MAE of about 8.60, followed closely by Xception at 8.76, while ResNet50's performance (MAE  13.85) was more moderate.

These findings highlight the value of leveraging well-established, pre-trained architectures for complex tasks like age estimation. Pretrained models like VGG16 and Xception not only converged faster but also generalized better to unseen data. In the future, refining hyperparameters, employing more targeted data augmentation strategies, or experimenting with specialized loss functions could further boost accuracy. Additionally, the techniques and insights gained here can be extended beyond age prediction to broader applications such as demographic analysis, personalized recommendations, and healthcare assessments, where estimating characteristics from images can provide meaningful value.

# References

[1] https://github.com/mellophone/ai-face-detection