Age Classification Based on Decision Trees and Neural Networks

Azam Arvin Marzieh Adeli Zahra Ebrahimi Anmol Majithia

1. Abstract

The age classification is an essential task in computer vision to determine a person's age based on facial images. Despite its applications in various fields, age classification poses challenges due to facial feature variations, aging patterns, and similarities between age groups[13]. Early methods relied on manual feature extraction using techniques like LDA and PCA, followed by traditional machine learning models such as SVM. However, these methods had limitations and yielded inaccurate results. This report introduces a systematic approach to address these challenges using supervised and semi-supervised decision trees and deep neural networks, including models like ResNet18, MobileNet, and ShuffleNetV2 trained on UTKFace datasets. The proposed methodology includes data preprocessing, hyperparameter optimization, and handling class imbalance. Through extensive experimentation, the optimized models achieved impressive accuracy in age classification, with ShuffleNetV2 consistently outperforming the other models, highlighting the proposed methodology's effectiveness.

2. Introduction

The age classification predicts an individual's age group based on information extraction from facial images utilizing specific approaches. It plays a significant role in numerous applications, such as healthcare and marketing. Age classification encounters difficulties due to its similarities with other tasks in interpreting facial images, such as face detection, feature extraction, and classification. Using human facial images in age classification has many applications in face recognition, targeted advertising, age-specific content filtering, biometrics, and personalized user experiences[2]. Accurate age estimation is a challenging problem for artificial intelligence systems due to the inherent variability in facial appearances, such as pose variations, expressions, changes in lighting conditions, occlusions, and image distortions caused by blur, noise, or other factors like makeup, and the complex nature of age-related changes. Despite individuals within a particular age group sharing the same chronological age, their appearances can differ significantly due to diverse factors such as their environment, lifestyle, and genetics. The intricate nature of these complexities poses significant difficulties for artificial intelligence systems in accurately determining a person's age using their facial features. Moreover, the need for pre-classified datasets and noisy images adds more complication to the age classification task. Nevertheless, despite the challenges involved, age classification remains a crucial application of facial detection technology in diverse sectors such as marketing, healthcare and security.

Over the years, various machine learning approaches have been employed to tackle the challenges associated with age classification in human faces, ranging from traditional techniques like decision trees to more sophisticated deep learning models. Furthermore, researchers have developed large datasets comprising pre-classified images specifically for training age classification algorithms and enhancing their accuracy, or techniques like data augmentation and normalization are widely employed [1]. Moreover, specific techniques use facial texture and shape features to improve the precision of age classification. Despite significant efforts, accurate age classification based on facial characteristics remains a complicated issue in artificial intelligence research[**?**]. Existing literature like Logistic Regression, SVMs, and VGG-16 have been used to tackle the challenges of age classification. While these methods have shown encouraging results, they also possess drawbacks, such as computational complexity for large datasets, lack of age-specific features and sensitivity to feature representation[5].

This report presents a comprehensive study addressing age classification challenges using three approaches, including supervised and semi-supervised decision trees and convolutional neural networks(ResNet18, MobileNetV2, and ShuffleNetV2) on the UTK Face dataset[10]. Our methods for decision trees include preprocessing and manipulating image data to work with decision trees and hyperparameter tuning to find an ideal max depth and evaluation. Our decision tree analyzes a dataset of facial images and their corresponding age labels. We extract features from each image by converting them to grayscale and flattening the resulting arrays. The age labels are encoded into two categories based on predefined age ranges. The dataset is then split into training and testing sets, and a decision tree classifier is trained using the training data. The trained classifier is used to predict the age labels for the testing set, and the accuracy of these predictions is computed. Doing this, we realized that although Decision Trees are great for simple tasks and very helpful in understanding as it is an explainable ML model when it comes to images, their simplicity leads to them falling a bit short of current bleeding edge technologies like Convolutional Neural Networks. Our approach for convolutional neural networks includes data preprocessing, exploratory data analysis, training, hyperparameter tuning, and evaluation. Our convolutional neural networks are trained using the Adam optimization, and we used 80% of our data for training, 10% for validation and the rest of the 10% for testing. We also applied a data augmentation technique to train our models with more data to prevent overfitting and we used a batch size of 32 and the Cross-Entropy loss function. Optimized convolutional neural network models achieved an overall accuracy for the age classification task, and test accuracy is reported for each model. Our results indicate the efficiency of our proposed approaches for training and optimized models for our age classification task.

2.1. Related Work

Early age estimation methods depended on calculating ratios between different measurements of facial features. These features include the eyes, nose, mouth, chin, and wrinkles. The process involved locating these features and measuring their sizes and distances to determine the ratios using manually designed rules and these ratios were then used to classify the face into specific age groups. Hayashi et al. proposed a technique based on wrinkles and skin texture analysis using the Hough transform to study facial features[7]. In the past few years Deep learning techniques have demonstrated remarkable advancements in automatic feature learning and image classification task. Widely used models like VGGNet and ResNet have shown promising results for their outstanding performance on facial recognition tasks[8]. Rethe et al. presented a methodology for estimating the age of individuals based on their facial images using Convolutional Neural Networks to learn discriminative features from facial images[11]. Recently K.A.Hossain et al. proposed a methodology for age estimation from face images by combining deep learning techniques with geometric elements and boosting ensemble methods[9].

In recent times, there have been notable progressions in utilizing artificial intelligence for age classification. Initial techniques involved assessing proportions between distinct facial characteristics to categorize faces based on age groups. More advanced approaches employ deep learning models to estimate age directly from facial pictures. Nevertheless, accurately identifying facial features remains a difficult issue, which hinders the suitability of these methods for analyzing images from social platforms in real-life scenarios.

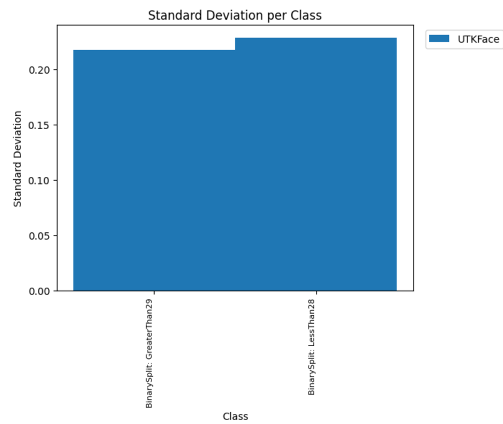

Figure 1. Standard Deviation Per Class for UTKFace Dataset

3. Methodology

3.1. Dataset

UTKFace is a large-scale face dataset that contains 23,706 face images of people of different ages, gender, and ethnicity. It was created by the University of Tennessee Knoxville(UTK) and is commonly used in computer vision for age classification, gender classification, and ethnicity recognition. This dataset is challenging due to variations in the lighting, poses and quality of images. The gender distribution is roughly balanced, with 48% male and 52% female. Each image in the UTKface dataset is stored as a JPEG file shown below as a table and is accompanied by metadata containing three labels: age, gender, and ethnicity. The age Labels range from 0 to 116 years, with an average age of 33 years, covering a broad spectrum of ages(see Table. 1). The gender labels are binary, classifying each face as either male or female. The ethnicity labels classify the subjects into five categories: White, Black, Asian, Indian, and Others(see Table. 1). The UTKFace dataset was collected from various sources, including online sources, image search engines, and social media platforms[4].

Dataset	NB of Images	NB of Classes	Format
UTKFace	23,706	2	JPEG

Table 1. Specifications of the UTKFace Dataset

Overall, we divided UTKface Dataset into three parts: Training, Validation, and Testing, with a distribution of 80%,10%, and 10%, respectively. Certain preprocessing and filtering techniques were employed to prepare the data, including resizing all images to a standardized size and normalizing the pixel values. We calculated the standard deviation for each class of our dataset, and the mean of standard deviation values is 0.22, which shows that this dataset is easy to be classified since higher values show spreading out of a class that leads to more difficult classification tasks

Figure 2. Sample of Augmented Image of UTKFace Dataset

with probable more inaccurate results. We applied preprocessing techniques like image resizing and normalization on our dataset. A sample of image dataset with applied augmentation technique is shown above(see Figure. 2).

3.2. Decision Trees

In our study, we applied the Decision Tree algorithm to image data by converting the images to grayscale and flattening the pixel values. These flattened pixel values were used as features, while the age of the subjects represented the labels.

The insights gained from our experiment were also valuable in the context of semi-supervised Decision Trees, with only 20% of the data labelled and the remaining 80% left unlabeled. We generated five different decision trees and, during each iteration, assessed the confidence of predictions for the unlabeled data. Any prediction above a certain threshold was added to the labelled group, reducing the size of the unlabeled group. Initially, we began with approximately 4,000 labelled data points. Eventually, we successfully labelled over 23,000 data points and achieved an accuracy of 65%.

To enhance the accuracy of our models, an additional approach was employed. Given that our dataset exclusively consisted of facial images, we attempted to implement the Haarcascade face detector to further refine the cropping process, ensuring that each face was as similar as possible across all photos. However, during the implementation, we discovered that the default Haarcascade Face Detector provided by openCV exhibited limited accuracy. Consequently, many images were erroneously diagnosed as devoid of any faces, resulting in excluding those images from the dataset. Given this situation, we were faced with a decision between marginally cleaner data, as the original dataset was already quite clean, and a substantially larger dataset (7000 vs. 23000). Ultimately, we opted for the latter, as the augmenting the dataset with a greater volume of data typically leads to an optimal outcome for models.

3.3. Convolutional Neural Network

Our project utilized three popular convolutional neural networks (CNN) models, namely ShuffleNetV2, ResNet18, and MobileNetV2 since these models are widely used for binary classification tasks due to their high accuracy and high performance. ResNet18 is a specialized variant derived from the ResNet architecture suitable for image classification and is designed to overcome the differences associated with training extremely deep learning[3]. ResNet18 offers a notable advantage in terms of computational and memory requirements and also proves to be highly accessible for deployment on devices that have limited computational resources. MobileNetV2 is a convolutional neural network (CNN) model designed to strike a balance between the size of the model and its accuracy[12]. It was specifically designed to address the computational limitations of mobile and embedded devices while maintaining good accuracy. ShuffleNetV2 is an advanced convolutional neural network model that balances accuracy and computational efficiency and it offers versatile configurations that can be customized to suit specific requirements, enabling a flexible trade-off between accuracy and efficiency. The remarkable performance of ShuffleNetV2 is attributable to its ingenious design principles, including channel shuffling for effective information exchange and group convolution for parallel computations. The approach employed in this methodology utilizes group convolutions and channel shuffling techniques to decrease the number of floating-point operations (FLOPS) needed during training while still preserving high accuracy. Overall, our selected neural network models are designed to tackle different challenges. ResNet18 focuses on achieving high accuracy, MobileNetV2 emphasizes efficiency in mobile devices, and ShuffleNetV2 balances accuracy and computational cost, allowing us to choose the best model for our classification task. The chosen CNN models' computational complexities for training and validation stages, in terms of the time taken for one epoch of training and the number of FLOPS calculations, can be described as follows:

ResNet18, when trained for a single epoch, necessitated approximately 990.436 billion FLOPS. In contrast, ShuffleNetV2, on the other hand, utilized about 393.202 billion FLOPS and MobileNetV2 demanded around 4018.565 billion FLOPS for the same training duration. When trained and validated for 100 epochs, ResNet18 required approximately 6978 seconds, averaging 69.78 seconds per epoch. ShuffleNetV2, on the other hand, took around 8620 seconds in total, with an average of 86.2 seconds per epoch for training and validation. MobileNetV2's training and validation process took about 8056 seconds, averaging 80.56 seconds per epoch.

To sum it up, ShuffleNetV2 achieved the highest accuracy amongst all our models, including decision trees and convolutional neural networks. Still, training and validation takes the most time compared to other convolutional neural networks. ResNet18 obtained an accuracy between MobileNetV2 and ShuffleNetV2, but it was relatively the fastest regarding training and validation time. MobileNetV2 takes less time than ShuffleNetV2 for training and validation tasks, but it has the least accuracy among our proposed neural networks. Obviously, all of the proposed convolutional neural networks got better accuracy than our decision trees. Moreover, regarding the accuracy of the age classification task in our decision trees, the Supervised decision tree outperforms the Semi-Sepervised model.

3.4. Optimization

Regrarding decision trees, upon running the model, we observed overfitting, where the training accuracy exceeded 90%, while the testing accuracy remained around 40%. To address this issue, we conducted multiple model iterations on a small subset of the data, adjusting the hyperparameter for maximum depth from 1 to 20. This revealed a decline in accuracy beyond a depth of seven. Consequently, we set the model to this depth and retested it, resulting in a test accuracy of 67%.

We used the Adam optimization algorithm to train our convolutional neural networks for the age classification task[6]. A learning rate of 0.01 was chosen, and a fixed batch size of 32 was used due to limited GPU machines. Various hyperparameters were employed to optimize our Convolutional Neural Network, including the batch size, learning rate, and loss function. We used the Cross-Entropy loss function for our binary class classification task.

To validate and optimize our convolutional neural networks, we used a training and validation set to train our model and fine-tune the hyperparameters. The test set was employed to evaluate the final performance of the model on unseen data. During the training phase, we continuously monitored our model's performance on the validation set and adjusted the hyperparameters accordingly. We followed an iterative approach, starting with a baseline model that used default hyperparameters. We then systematically varied the hyperparameters to identify the optimal values. We tuned our learning rate and increased it gradually from 0.001 to 0.1 while observing the training loss. Then the learning rate was set to the best value, and training loss started improving. In the last step, we evaluate the performance of our optimized decision trees and convolutional neural networks using various metrics, including precision, recall, F1 score, and the confusion matrix.

4. Results

4.1. Experiment Set up

Our decision tree approach began with data preprocessing; then, we dedicated 80% of the data to training, 10% to validation, and 10% to testing. Then we applied our decision tree models to that, and we had overfitting problems. To resolve this issue, we tested what effect we would have by changing the hyperparameters like min_samples_split, max_depth and min_samples_leaf. We tested a subset of the whole data. We trained multiple models with changing hyperparameters and saw the most significant improvement with max_depth. We found the best result with a depth of 7, where we got our highest accuracy. For semi-supervised, we used the same max_depth from above and started by taking 20% of the data as labelled and deleting the labels for the remaining 80%. On each iteration, we checked the classification confidence for each prediction, and every value with a confidence of more than 0.7 was removed from unlabeled and added to labelling.

Our initial step for convolutional neural network models was pre-processing the dataset, and our data splitting was the same as the decision trees approach. We used the data augmentation technique to obtain more data, get better results and prevent overfitting by image shifting, rotations and flipping. We used the Adam optimization algorithm with a batch size 32 due to the lack of high-speed GPU and Cross-Entropy loss function. It is observed from our optimization technique that we got the best result for a learning rate of 0.01 for our dataset.

The performance of the optimized CNN models was assessed using various metrics, including ROC/AUC, confusion matrix, precision, recall, and F1 score and for our decision trees we used the same metrics except ROC/AUC. These metrics collectively provided a comprehensive evaluation of the model's classification accuracy and its ability to correctly classify different age groups. Additionally, they provided insights into the misclassification patterns of the model across different classes. Our optimized Convolutional Neural Network models and decision trees achieved an overall accuracy for age classification, and the test accuracies for our dataset are as follows: Resnet18 achieved an accuracy of 81.72%, while MobileNetV2 and ShuffleNetV2 reached 80.72% and 83.07%, respectively. In terms of our decision trees, we got an accury of 67.47% using supervised approach and 67.47% using semi-supervised approach.

These results demonstrate the efficacy of the proposed methodologies for training and optimizing Convolutional Neural Network models and Decision Trees for binary-class age classification. The proposed methodologies involved pre-processing the data augmentation, training, evalution and hyperparameter tuning.

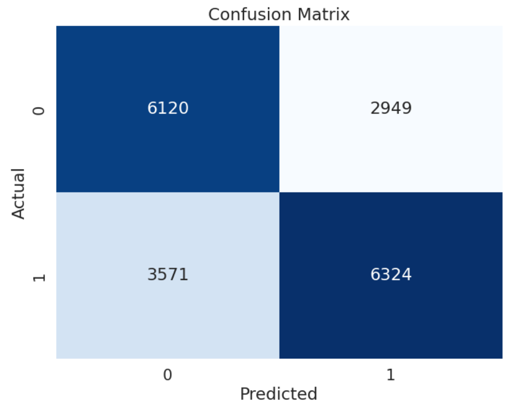

Figure 3. Confusion Matrix of Semi-Supervised Decision Tree on UTKFace Dataset

4.2. Main Results

We trained our Semi-Supervised Decision tree on our UTKFace dataset, and with each iteration, we are getting more samples. The number of our samples on the Semi-Supervised decision tree increased from iteration 1 to iteration 3 from 4741 to 22620 samples, and our accuracy increased from 81.75% to 91.73%, and then from iteration 4 to iteration 5, we had a slight decrease in accuracy from 91.73% to 90.59%. Finally, we got our confusion matrix result above(see Figure. 3).

Moreover, by testing a subset of the data, it was found that increasing the max_depth to 7 yielded the best results(see Figure. 4), with training accuracy reaching around 80% and testing accuracy at 67.47%. In the semi-supervised approach, 20% of the data was initially labeled, and on each iteration, unlabeled samples with a classification confidence above 0.7 were added to the labeled section. The number of labeled images increased significantly, reaching over 23,000 by iteration 5. The final training accuracy was 90.59%, with a testing accuracy of 65.61%. Overall, what is obvious is that we got better accuracy on Supervised Decision Tree since our accuracy on Supervised Decision Tree is 67.647% which shows better performance than the Semi-Supervised approach.

Comparing our Convolutional Neural Network models, we got our best accuracy on the ShuffleNetV2 since it outperformed other neural networks and achieved an accuracy of 83.07%. We trained ShuffleNetV2 on 100 epochs with a batch size of 32, and our training loss decreased from 0.6473 from the first epoch to 0.3021 on the last epoch, and our training accuracy got better from 62.48% to 85.95%, which obviously shows that our model is performing well on the training set and it reached to the 83.43% of the validation accuracy in the last epoch((see Figure. 5)). As we discussed in the previous section we trained our dataset on MobilenetV2 and Resnet18 too and based on our results as

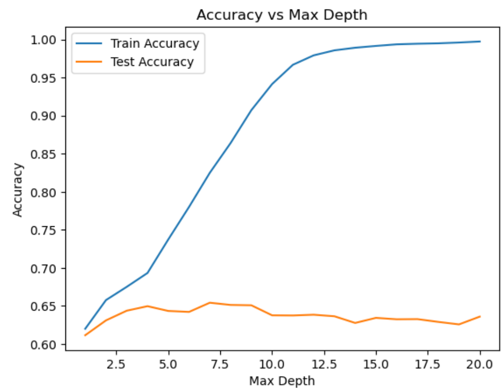

Figure 4. Accuracy Vs Max Depth of Decision Tree on UTKFace Dataset

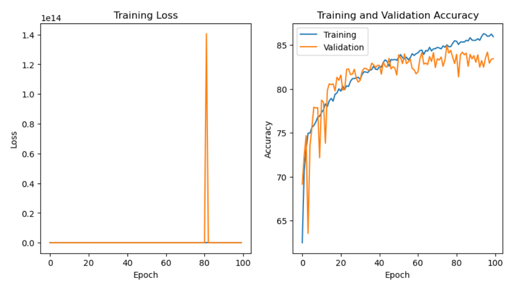

Figure 5. Training Loss and Training and Validation Accuracy per Epoch for ShuffleNetV2

we explained before Resnet18 takes less time to be trained on our dataset. For all trained models we gained the confusion matrix and the details of the our metrics are given in the shown table(see Table. 2).

Model	Precision	Recall	F_Score	Acc
Supervised	0.67	0.61	0.63	0.67
Semi-Supervised	0.63	0.67	0.64	0.65
MobileNet	0.72	0.46	0.56	0.80
ResNet	0.83	0.80	0.81	0.81
ShuffleNet	0.82	0.83	0.82	0.82

Table 2. Precission, Recall, F_Score and Accuracy of Our Proposed Methodologies

During the process of tuning the learning rate, we began with an initial search using a wide range of learning rates (0.001 to 0.1) on a logarithmic scale. After examining the initial outcomes, we conducted a more detailed search within the same range (0.001 to 0.1) to fine-tune our results. It was discovered that a learning rate of 0.01 delivered the most optimal performance across all three datasets and three CNN models.

4.3. Ablative Study

In order to comprehensively analyze the impact of various hyperparameters on model performance, an ablative study was conducted. The study involved tweaking 2 different hyperparameters such as the number of classes for training, number of images per class training. The objective was to understand how these adjustments influenced the overall performance of the model. The following observations were made: Number of Classes for Training: Increasing the number of classes for training had a noticeable effect on the model performance. As the number of classes increased, the model faced more complexity in distinguishing between different classes, leading to a slight decrease in accuracy. Number of Images per Class Training: The number of images per class during training had a significant impact on the model's ability to learn and generalize. Increasing the number of images per class resulted in improved accuracy, as the model had more examples to learn from and could better capture the patterns and features specific to each class.

Code of this project is available at: 6721_Project.

References

[1] Mohamed El Ghmary, Younes Ouassine, and Ali Ouacha. Automatic license plate recognition with yolov5 and faster-rcnn. In Mohamed Lazaar, El Mokhtar En-Naimi, Abdelhamid Zouhair, Mohammed Al Achhab, and Oussama Mahboub, editors, *Proceedings of the 6th International Conference on Big Data and Internet of Things*, pages 351–361, Cham, 2023. Springer International Publishing. 1

[2] Yun Fu, Guodong Guo, and Thomas S. Huang. Age synthesis and estimation via faces: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(11):1955–1976, 2010. 1

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 3

[4] Van-Thanh Hoang and Kang-Hyun Jo. *Modified Stacked Hourglass Networks for Facial Landmarks Detection*, pages 648–657. 01 2019. 2

[5] Olga Ilina, Vadim Ziyadinov, Nikolay Klenov, and Maxim Tereshonok. A survey on symmetrical neural network architectures and applications. *Symmetry*, 14(7), 2022. 1

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4

[7] Sandeep Kumar, Sukhwinder Singh, Jagdish Kumar, and K. M. V. V. Prasad. Age and gender classification using seg-net based architecture and machine learning. *Multimedia Tools Appl.*, 81(29):42285–42308, dec 2022. 2

[8] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks, 2017. 2

[9] Seyed Mousavi and Vyacheslav Lyashenko. Age estimation in make-up (cosmetics) using color and depth images. 11 2019. 2

[10] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation, 2016. 1

[11] Rasmus Rothe, Radu Timofte, and Luc Van Gool. Deep expectation of real and apparent age from a single image without facial landmarks. *International Journal of Computer Vision*, 126, 04 2018. 2

[12] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. 3

[13] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep convolutional network cascade for facial point detection. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3476–3483, 2013. 1