Project Documentation: Face-Based Age Estimation Using Regression

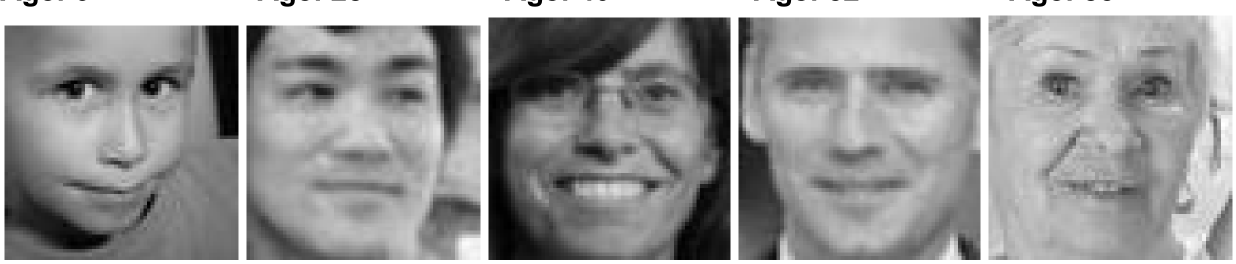
GR. 5

1. Project Overview

This project implements a deep learning solution to predict the exact age of a person based on facial images. It treats the age estimation task as a regression problem, predicting a continuous age value using a Convolutional Neural Network (CNN), specifically the ResNet-18 architecture. The solution employs Mean Squared Error (MSE) loss during training and evaluates performance using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). The dataset used is UTKFace, it contains 23 707 images of faces with annotated ages.

Some example images with age (after preprocessing):

Age: 6 **Age: 28** **Age: 40** **Age: 52** **Age: 85**



2. Project Structure and File Descriptions

The project is structured into clearly defined files:

Directories:

- data/raw: Contains original, unprocessed images.
- data/processed: Contains processed (resized and grayscale) images used for training.
- outputs/models: Stores the best-performing model checkpoints.
- plots: Stores generated plots for visual analysis.

Main Scripts:

- **train.py**:
  - Trains the ResNet-18 regression model.
  - Implements data loading, model training loop, validation checks, and saves the best model based on validation loss.
  - Produces training and validation loss plots.
- **evaluate.py**:
  - Evaluates the trained model's performance on the validation dataset.
  - Generates detailed performance metrics (MAE, RMSE).
  - Produces multiple visualizations to analyze model predictions, residuals, and errors.

Supporting Modules:

- **data_loader.py**:
  - Defines AgeRegressionDataset class for loading and transforming images.
  - Provides a utility function get_dataloaders() to return PyTorch DataLoader objects for training and validation.
- **models/resnet18.py**:
  - Implements a customized ResNet-18 CNN model tailored for regression.
  - Modifies the final fully connected layer for single continuous output.
- **utils.py**:
  - Includes various utility functions for setting random seed, plotting visualizations (e.g., predictions vs. actual, error distributions, loss curves), and analysis.
- **config.py**:

    - Stores configuration parameters such as directory paths, hyperparameters (e.g., epochs, batch size, learning rate), and random seed.

3. Detailed Instructions for Running the Code

Step 1: Data Preparation

- Ensure raw images are placed in data/raw.
- Run the preprocessing script to convert and resize images:

python preprocess.py

Step 2: Train the Model

- Execute the training script:

python train.py

- Model checkpoints are automatically saved in outputs/models.
- Training loss plots are saved in the plots directory.

Step 3: Evaluate the Model

- After training completes, evaluate the model using:

python evaluate.py

- Evaluation script prints performance metrics (MAE and RMSE) and saves detailed analysis plots in the plots directory.
- Alternatively the train and evaluate scripts can be combined by calling main.py

4. Results and Analysis

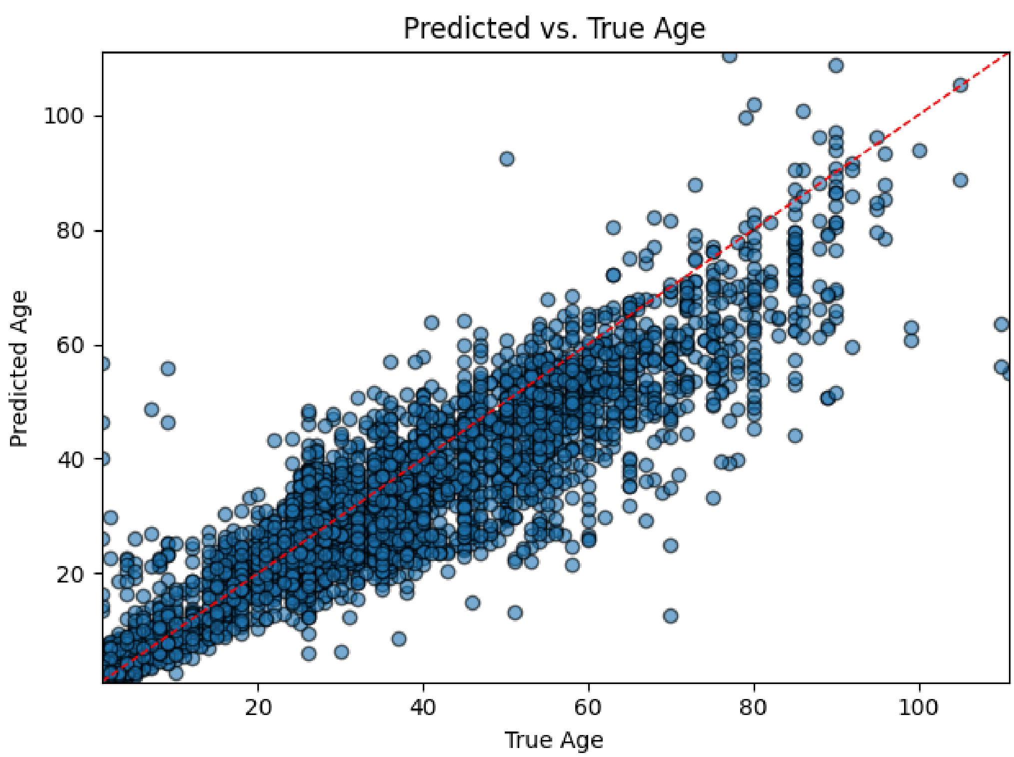
The trained model was evaluated on a validation dataset which contained 4740 images , producing the following key metrics:

- **Mean Absolute Error (MAE)**: 6.086
- **Root Mean Squared Error (RMSE)**: 8.821

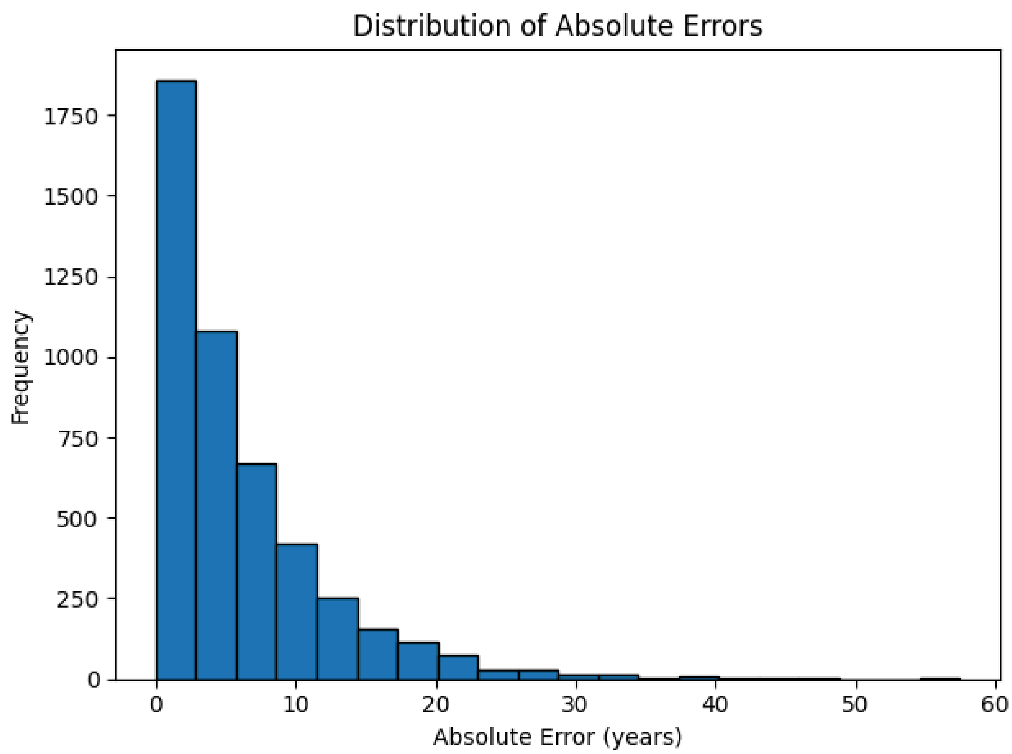
The result means that on average, the predicted ages are around 6 years off from the actual values. While not perfect, these results are reasonable for age estimation from face images and show that the model captures general age patterns fairly well. There's still room for improvement, especially in reducing larger errors, but it's a strong starting point.

Visualizations

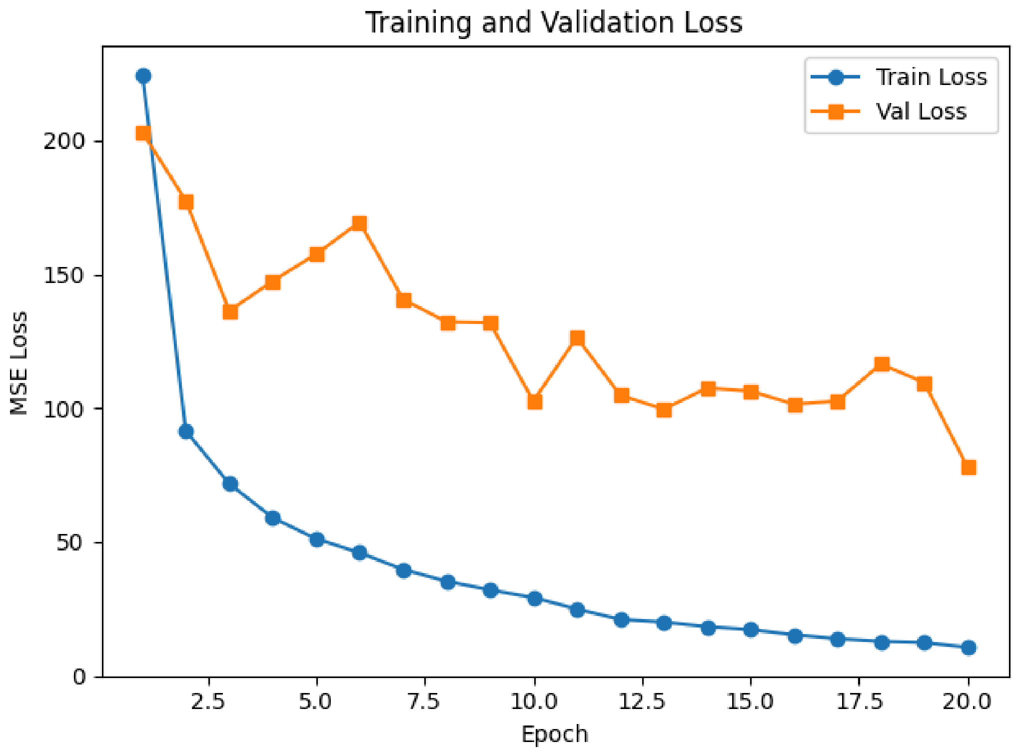
- **Predicted vs. Actual Ages** (plots/pred_vs_true.png):
  - Illustrates the model's prediction accuracy by comparing predicted ages against true ages.



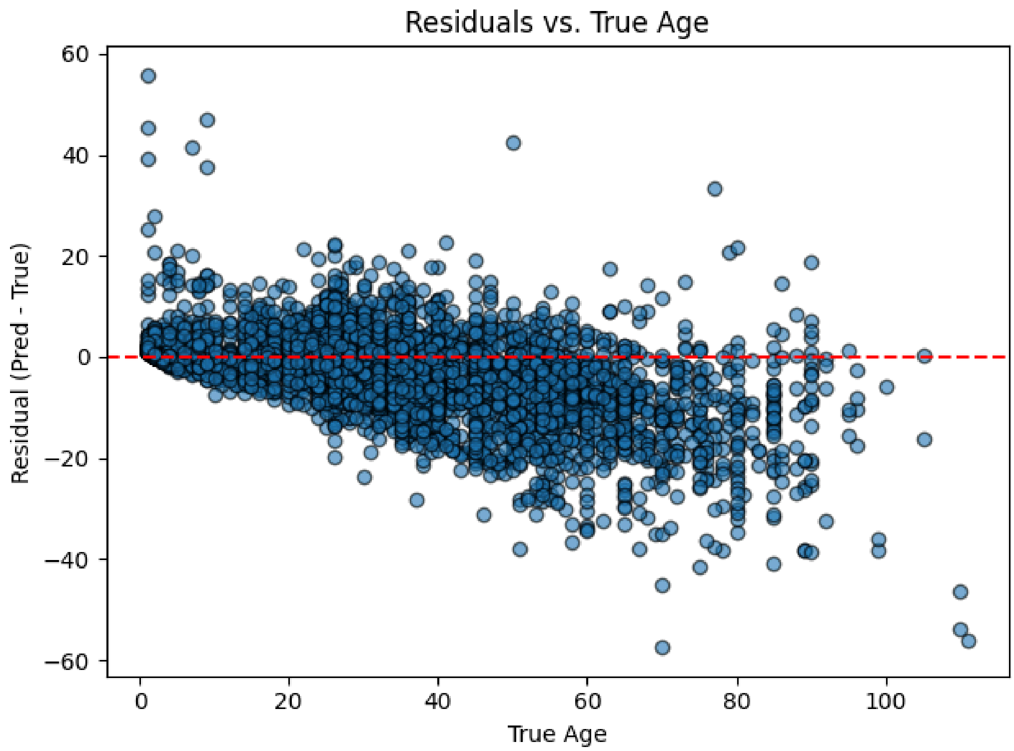
- **Error Distribution** (plots/error_dist.png):
    - Provides insight into the frequency of absolute prediction errors.
    - We can observe that most of the errors are within 0-3 years.



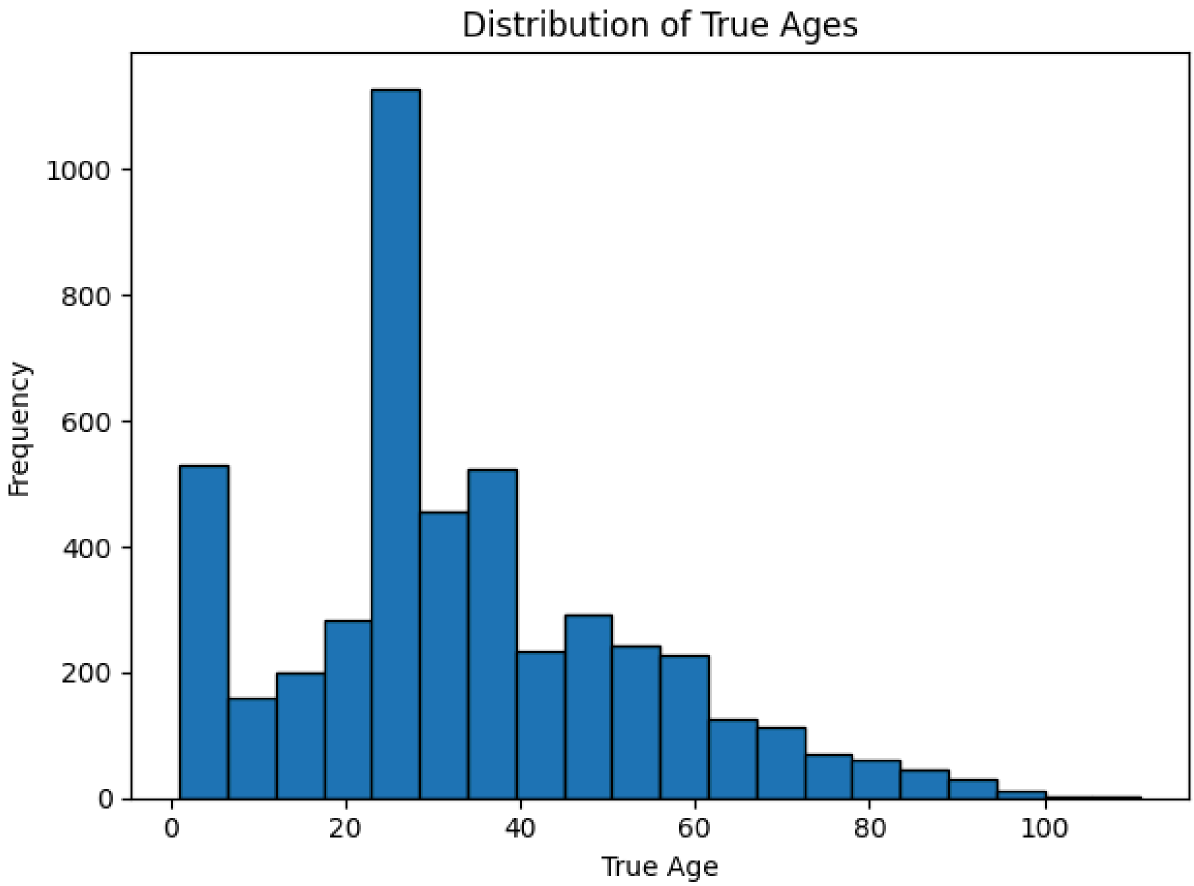
- **Training and Validation Loss Curve** (plots/loss_curve.png):
    - Shows the progression of the model's learning and potential overfitting patterns
    - We can observe on the Val loss that the model steadily learns, although comparing 2 plots the difference between train and val loss may suggest overfitting



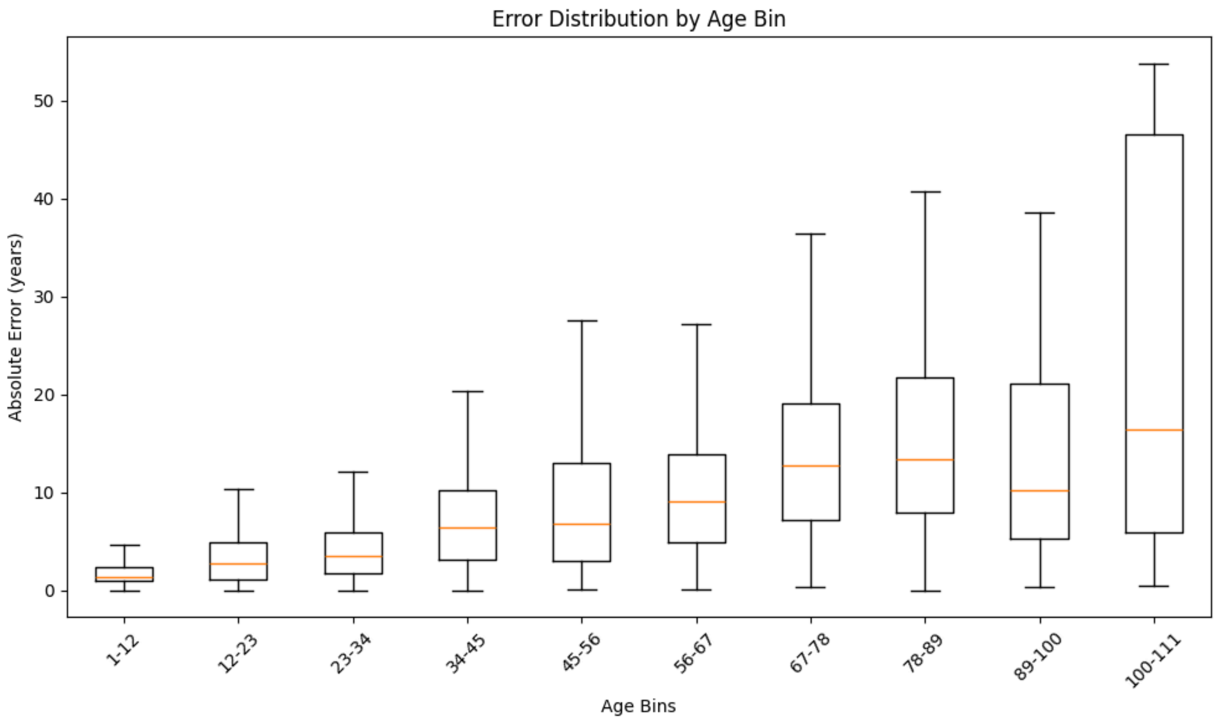
- **Residuals Analysis** (plots/residuals_vs_true.png):
    - Examines prediction residuals (errors) against true age values to identify biases or trends.
    - It can be observed that the higher the true age, the predictions stay low, it may be because of the unbalanced dataset



- **True Age Distribution** (plots/true_age_dist.png):
    - Highlights age distribution within the validation set.
    - On this plot we can see the distribution which suggest very unbalanced dataset in terms of ages



- **Error by Age Bins** (plots/error_by_age_bin.png):
    - Analyzes prediction accuracy across different age groups, highlighting specific areas of strength or weakness.
    - Easy to observe that the higher this plot has a similar but reversed pattern to the previous plot, the less samples we get in training, the larger the error is.



5. Challenges and Solutions

- **Filename Parsing and Data Loading**:
    - Challenge: The annotations were inside the images names instead of csv, which was new to me
    - Solution: Implemented text splitting for extracting and assigning the age to each image

- **Finding an environment**
    - Challenge: Find an environment with enough computing power to run the training
    - Solution: Used google colab with bought credits for CPU usage

6. Conclusion

The ResNet-18 model we built does a solid job predicting someone's age from a photo. It's simple to understand and gives measurable results. We followed a clear process preparing the data, evaluating performance, and generating helpful visualizations which makes it easy to reproduce or improve in the future.

With a Mean Absolute Error (MAE) of around **6.09 years**, the model performs close to how well people typically guess each other's age. In fact, one study found that participants were off by a median of **5.15 years** when estimating age from facial images without any disguise ([link to study]) so our model has similar MAE.