Age Detection in Image Classification

Aluya Omofuma and Angelica Wiltz

Goals: Model Comparison/ Model Improvement

- Comparison of CNN model and Pretrained model (VGG16)
  - We chose to do a model comparison to evaluate the difference between pretrained models and models built from scratch
- Improvement of CNN model
  - We obtained the data from kaggle.
  - We aimed to improve on this model

Dataset

- UTKFace Dataset
- Aligned and Cropped Faces
- GitHub repository created by graduate students from The University of Tennessee
- Approximately 24,000 face images that were webscraped from Wikipedia and IMDB
- Labels of face images are embedded in file name

- Age
- Gender
- Race
- Date and Time

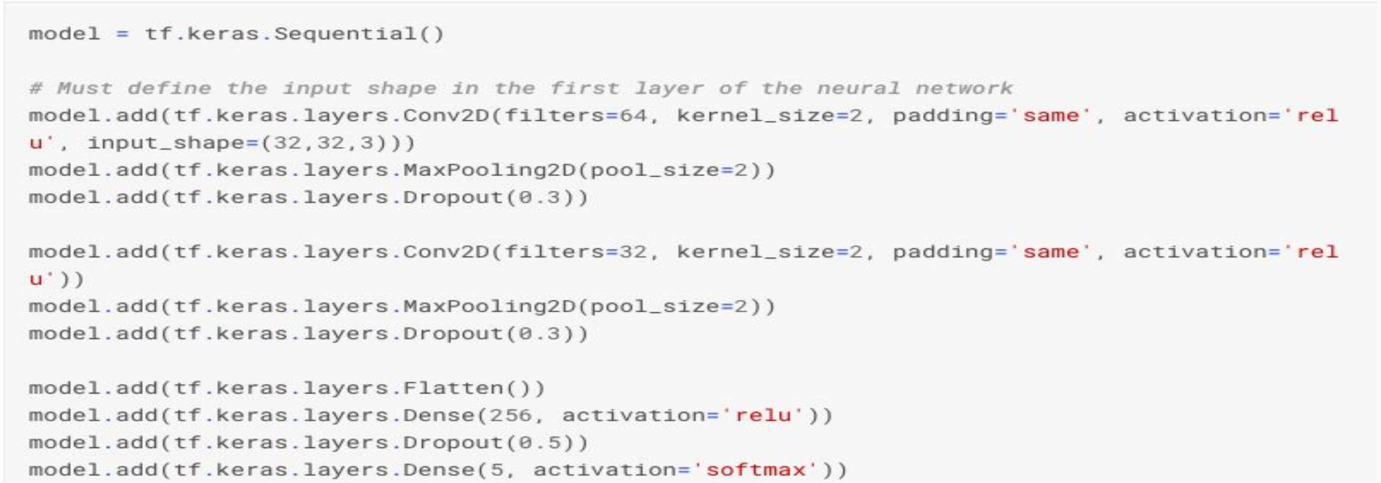
Data

Data Preprocessing

- Indexed age from filename on the images
- Created six classes for age classification - increased from four classes
- Resized images from 200x200 to 32x32
- Converted data from lists into arrays
- Applied keras' one-hot encoding to categorical labels with 6 labels
- Split data into training, testing, and validation sets 70/15/15

CNN Model

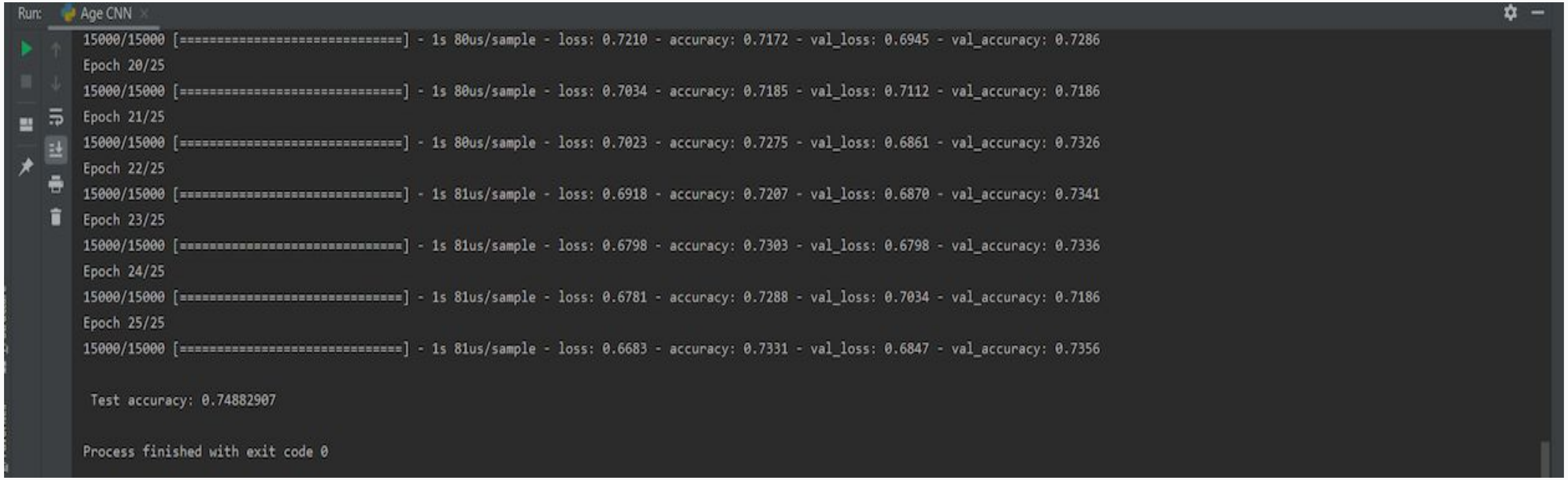
- Resized images 32x32
- Increased classes for age groups
- 3 Layers



```
model = tf.keras.Sequential()

# Must define the input shape in the first layer of the neural network
model.add(tf.keras.layers.Conv2D(filters=64, kernel_size=2, padding='same', activation='rel
u', input_shape=(32,32,3)))
model.add(tf.keras.layers.MaxPooling2D(pool_size=2))
model.add(tf.keras.layers.Dropout(0.3))

model.add(tf.keras.layers.Conv2D(filters=32, kernel_size=2, padding='same', activation='rel
u'))
model.add(tf.keras.layers.MaxPooling2D(pool_size=2))
model.add(tf.keras.layers.Dropout(0.3))

model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(256, activation='relu'))
model.add(tf.keras.layers.Dropout(0.5))
model.add(tf.keras.layers.Dense(5, activation='softmax'))
```

Results - CNN

- 73% accuracy for the training set, 0.66 loss
- 74% test accuracy

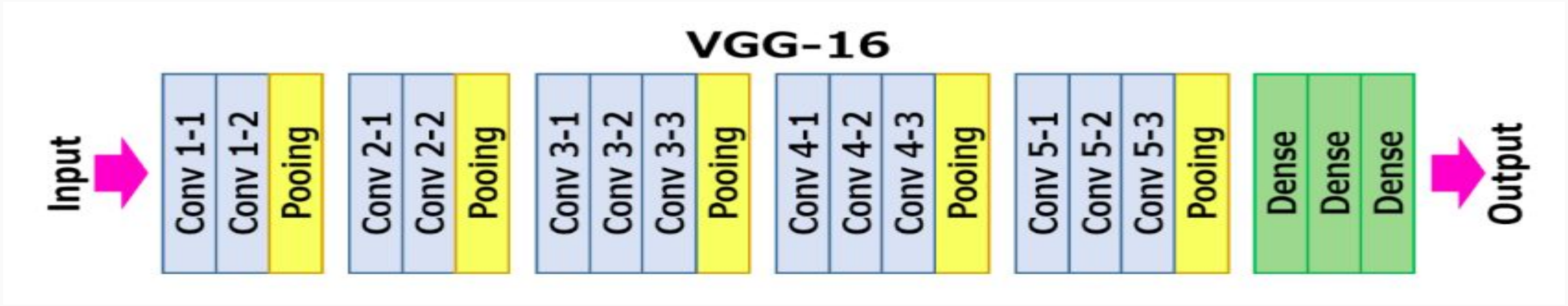


Run: Age CNN

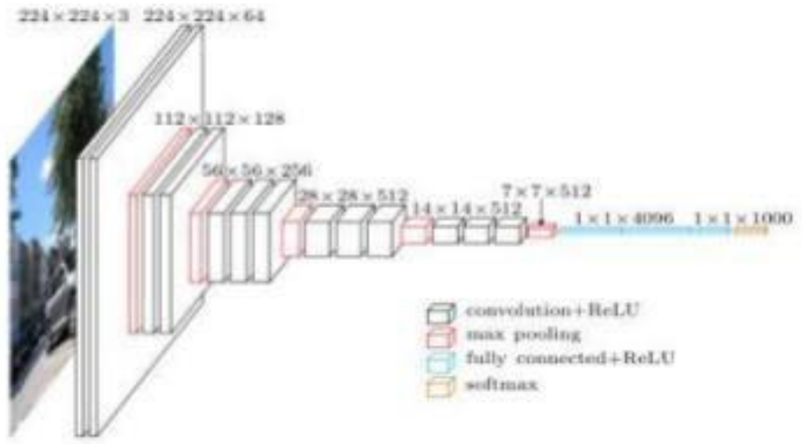
```
15000/15000 [==============================] - 1s 80us/sample - loss: 0.7210 - accuracy: 0.7172 - val_loss: 0.6945 - val_accuracy: 0.7286
Epoch 20/25
15000/15000 [==============================] - 1s 80us/sample - loss: 0.7034 - accuracy: 0.7185 - val_loss: 0.7112 - val_accuracy: 0.7186
Epoch 21/25
15000/15000 [==============================] - 1s 80us/sample - loss: 0.7023 - accuracy: 0.7275 - val_loss: 0.6861 - val_accuracy: 0.7326
Epoch 22/25
15000/15000 [==============================] - 1s 81us/sample - loss: 0.6918 - accuracy: 0.7207 - val_loss: 0.6870 - val_accuracy: 0.7341
Epoch 23/25
15000/15000 [==============================] - 1s 81us/sample - loss: 0.6798 - accuracy: 0.7303 - val_loss: 0.6798 - val_accuracy: 0.7336
Epoch 24/25
15000/15000 [==============================] - 1s 81us/sample - loss: 0.6781 - accuracy: 0.7288 - val_loss: 0.7034 - val_accuracy: 0.7186
Epoch 25/25
15000/15000 [==============================] - 1s 81us/sample - loss: 0.6683 - accuracy: 0.7331 - val_loss: 0.6847 - val_accuracy: 0.7356

 Test accuracy: 0.74882907

Process finished with exit code 0
```

VGG16

- Pretrained on ImageNet (15 million images, 22,000 categories)
- 92.7% top -5 test accuracy - ImageNet Large Scale Visual Recognition Competition 2014
- Twelve convolutional layers, 2D and 3D, 3x3 filters , stride = 1, Relu activation
- One flattened convolutional layer, two fully connected layers
- Input size of 224x224x3



VGG16 Pre-Trained Model



The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes.

Citation: https://arxiv.org/pdf/1409.1556.pdf

VGG16 Model

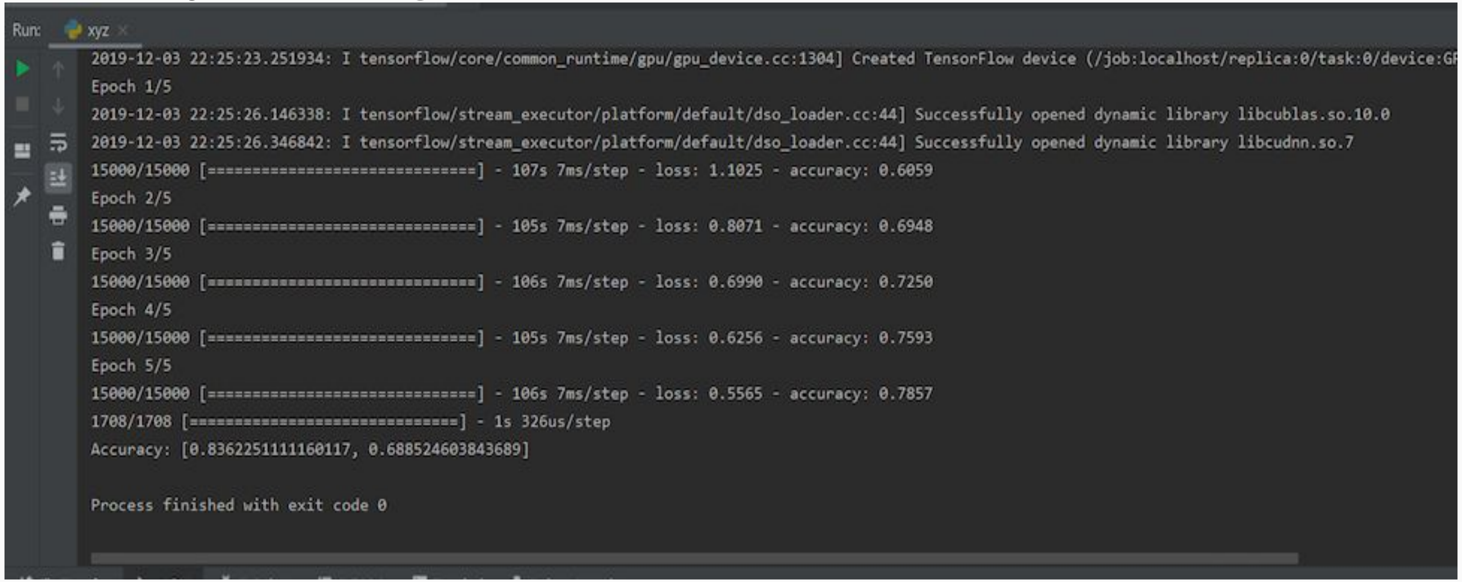
- Resized images 224x224 to match input size of VGG16 model
- Split training and testing samples by 15000/7000
- Added 2 Dense layers with 5000 neurons + relu activation
- Added 1 Dense layer to calculate the 6 classes using a softmax function, sums classes to 1
- Adam optimizer with learning rate = 0.001
- Preprocessed the input using the keras VGG16 preprocessing input function
- 5 epochs, loss = categorical cross entropy, metrics = accuracy
- 50% accuracy, 1.4 loss

VGG16 Model - Improved

- Resized input to 32x32 due to low memory
- Eliminated additional dense layers
- Used different batch size from CNN model - low memory capacity
- Added “imagenet” weights to VGG model
- Assigned Global Average Pooling 2D to the final dense layer, minimized overfitting/reduced numbers of parameters, computed the average of the output
- Fit model to the VGG16 model inputs and output predictions

Results - VGG16

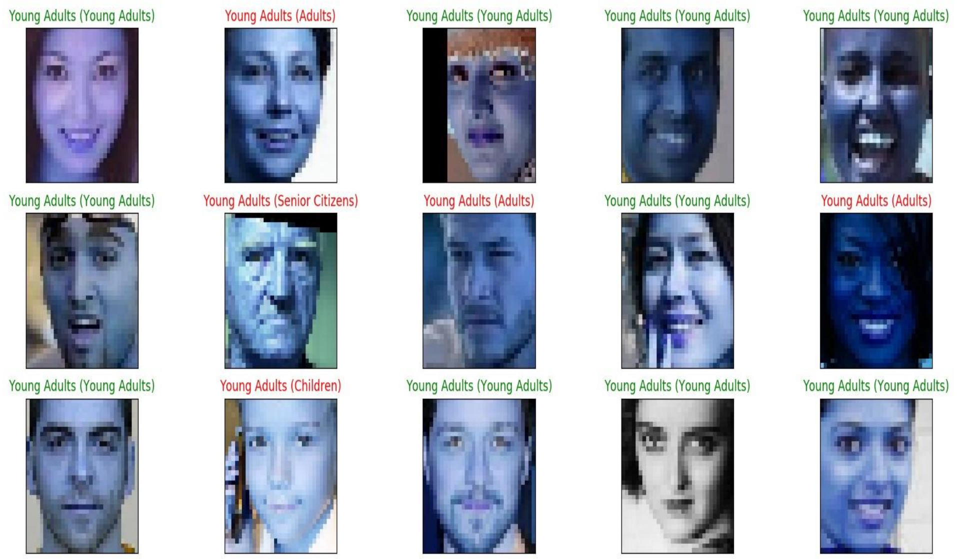
- 78% accuracy on training models, .55 loss
- 68% accuracy on testing set, .83 loss



```
2019-12-03 22:25:23.251934: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1304] Created TensorFlow device (/job:localhost/replica:0/task:0/device:GP
Epoch 1/5
2019-12-03 22:25:26.146338: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcublas.so.10.0
2019-12-03 22:25:26.346842: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcudnn.so.7
15000/15000 [==============================] - 107s 7ms/step - loss: 1.1025 - accuracy: 0.6059
Epoch 2/5
15000/15000 [==============================] - 105s 7ms/step - loss: 0.8071 - accuracy: 0.6948
Epoch 3/5
15000/15000 [==============================] - 106s 7ms/step - loss: 0.6990 - accuracy: 0.7250
Epoch 4/5
15000/15000 [==============================] - 105s 7ms/step - loss: 0.6256 - accuracy: 0.7593
Epoch 5/5
15000/15000 [==============================] - 106s 7ms/step - loss: 0.5565 - accuracy: 0.7857
1708/1708 [==============================] - 1s 326us/step
Accuracy: [0.8362251111160117, 0.688524603843689]

Process finished with exit code 0
```

Classification Plot

Age Range	Label
0-6	Infants
7-12	Children
13-19	Teenagers
20-40	Young Adults
40-65	Adults
65+	Senior Citizens


Young Adults (Young Adults)
Young Adults (Adults)
Young Adults (Young Adults)
Young Adults (Young Adults)
Young Adults (Young Adults)
Young Adults (Young Adults)
Young Adults (Senior Citizens)
Young Adults (Adults)
Young Adults (Young Adults)
Young Adults (Adults)
Young Adults (Young Adults)
Young Adults (Children)
Young Adults (Young Adults)
Young Adults (Young Adults)
Young Adults (Young Adults)

Conclusion

- For the UTKFace dataset, our pretrained model computed a higher accuracy for the training set (78%) but a lower overall accuracy for the testing set
- Our CNN model computed a lower accuracy for the training set but a higher accuracy for the testing set
- Adding more bin sizes (classes) helped to improve the model
- CNN model improved accuracy from the initial model on Kaggle which was 63% to 74%
- CNN model trained at a faster rate compared to VGG16 model

References

1. “Very Deep Convolutional Networks for Large-Scale Image Recognition” - Karen Simonyan and Andrew Zisserman https://www.robots.ox.ac.uk/~vgg/publications/2015/Simonyan15/simonyan15.pdf
2. “Deep Expectation of apparent age from a single image” - Rasmus Rothe, Radu Timofte, Luc Van Gool https://www.computer.org/csdl/proceedings-article/iccvw/2015/5720a252/12OmNviZlBv
3. “Age Group Classification with CNN” https://www.kaggle.com/sangarshanan/age-group-classification-with-cnn