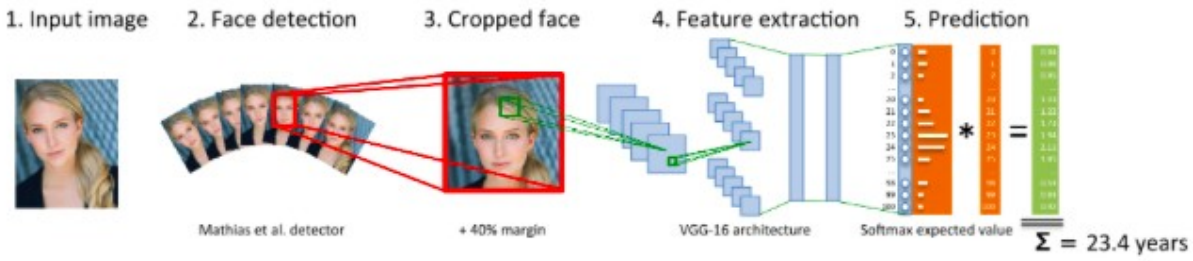
Final Report
Junsuk Oh

Image Classification by AgeWith Deep Learning

Junsuk Oh, Devyani Gupta, Wan-Ting Tsai, Henry Schwartz, Chance Wren



Abstract:

This report will explain, demonstrate, then conclude the significant aspects of Group 8's Age Detection with Deep Learning Analytics Project. The contents of this document are listed in order below:

Problem Identification

More and more children and minors are relying on the internet as a source of reliable information. As the number of devices that can access the internet rises in every household, more people are voicing their concerns regarding the lack of authorization control on numerous websites with sensitive content that may pose significant risks to minors. Although many websites have their own age policies, enforcing these policies is a monumental challenge.

It is our responsibility to create a safe online environment for people of all ages to be able to use the internet without exposure to various risks lurking online. In order to do so, our group will develop an age detection algorithm that will receive an image as an input to estimate the age of the individual in the image, in order to determine whether this particular individual is old enough to access and view contents from the website that he or she is trying to gain access.

Our goal is to use this algorithm to be implemented in various websites with sensitive information, including some social media websites. Whenever a user tries to access the website, instead of the old and ineffective pop-up that asks the user “Are you over 21?”, the website asks for authorization to briefly access the device’s camera, in order to capture an image of the user and determine whether the user is old enough to access the website.

Of course this algorithm with have obvious weaknesses such as having a user old enough simply gain access to certain age restricted content for them, using a static images of individuals who do meet the age requirement, and some users may even push back a refuse to relinquish their physical identity to gain access to certain materials. We can address several of these concerns by having open and easy to understand privacy and data regulations stating that images used for age verification will not be stored or kept and/or we could seek to implement our age recognition algorithm on sites that already thrive on sharing photographic content like Instagram for instance to simply flag certain profiles for review based on predicted ages.

Data Selection and Data Source

Raw Data 1

Our first and primary dataset will include over 34,000 images of 100 Indian actors, male and female, from more than 100 videos. This dataset was acquired from a similar data science project whose goal was to detect the relative age of individuals based upon their facial attributes within their photo. An individual's relative age was classified as either Young, Middle or Old. One of the most unique attributes of this dataset is its high amount of variability in terms of photo scale, subject poses, facial expressions, illumination, age, resolution, occlusion, and makeup. In addition, this high degree of variability can help other face related applications.

Raw Data 2

Since our project is contingent on large data sets of images. Finding datasets that contained many pictures of children proved quite difficult and this could be slightly due to the fact that they are children themselves. Our second datasets contained 3,828 images of 1,010 celebrities. For each identity at least one child/young image and one adult/old image was present. This dataset was provided by Simone Bianco, Assistant Professor of DISCo (Department of Informatics, Systems and Communication) at the University of Milan-Bicocca. Professor Bianco was able to intelligently scrape the internet for images of individuals that were publically available. Although the images in the dataset are limited in terms individuals and image quality, it gives our group a great starting point in terms of dataset, primarily because a majority of the images are segmented, aligned and crop to best show the individual's face. Please see figure 1 below:

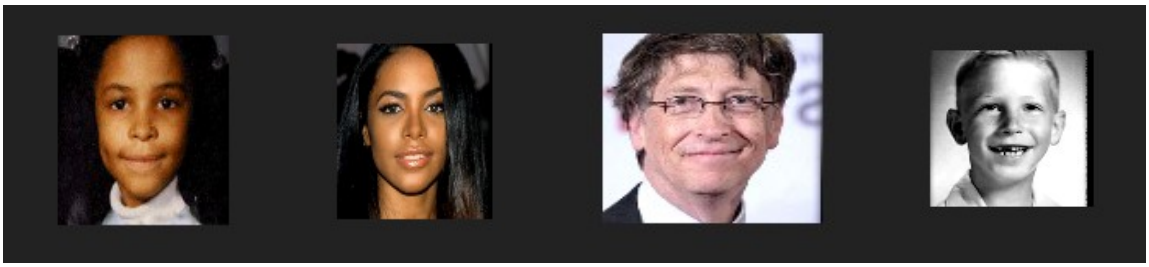


Figure 1: Aaliyah Haughton (young, left, old, right) & Bill Gates (old, left, young, right)

Although Simone Bianco originally used this dataset for face verification across large age gaps, our team can use some if not all of this dataset as a portion of our training data due to the diversity people and ages present in the dataset.

The combined use of datasets 1 and 2 will add not only an extra challenge when it comes to our data cleaning and uniformity, but this extra layer of variability will help to ensure the accuracy of our models will increase as well.

Data Cleaning and Point Deletion Overview

We deleted images that did not contain majority of facial features, images too blurry to identify important facial features and images which contained multiple faces. We could have extracted each of the faces in the photo, but this process is extremely difficult, especially when the quality of images is low or not all facial features are visible, simply because we needed to zoom into the photo to extract the faces of people in the back in a group photo. Zooming in reduces the number of pixels that can be used to hold information on facial features, which reduces the quality of the data. We did not want to risk negatively impacting the accuracy of our model, so majority of our data involved direct headshots. Since we are trying to control internet content access by using images, our final product wouldn't need to accept images that contain multiple faces as an input when it is implemented on a website. If it receives an image with multiple faces, it would simply count the number of faces the image contains, it would ask for a webcam headshot of the main

user. For our business implication purposes, these images possess little value, since our algorithm will be estimating ages of a single individual per image. Therefore, these images will be removed from our dataset for data normalization purposes.

Data Segmentation for Alignment and Cropping

It is common for images to contain faces with various sizes, lighting, and angle alignments. These differences make the data difficult to analyze in its raw form. Therefore, the data needs to go through the process of normalization. The normalization process will consist of detecting facial landmarks in each image, scaling the size of faces so that all faces are relatively the same size, centering all the faces and aligning the angle of faces so that both eyes lie on a horizontal line.

In order to successfully detect facial landmarks in each image, we need to detect the pixels that consists of the individual's face from an image. We achieve this by using a python library called OpenCV, specifically a pre-trained face detection classifier called Haar Cascade Classifier. Haar Cascade Classifier can process images quickly with high object detection rates because complex classifiers are layered in a cascade formation, which allows the model to disregard areas of the image that are unlikely to contain a facial landmark, and focuses on areas with higher probability. Haar Cascade Classifier uses Haar features, which are kernels that detect whether a feature it is searching for is present in an image. Predefined features in the classifier locates various facial features, such as eyes, nose, cheeks and mouth.

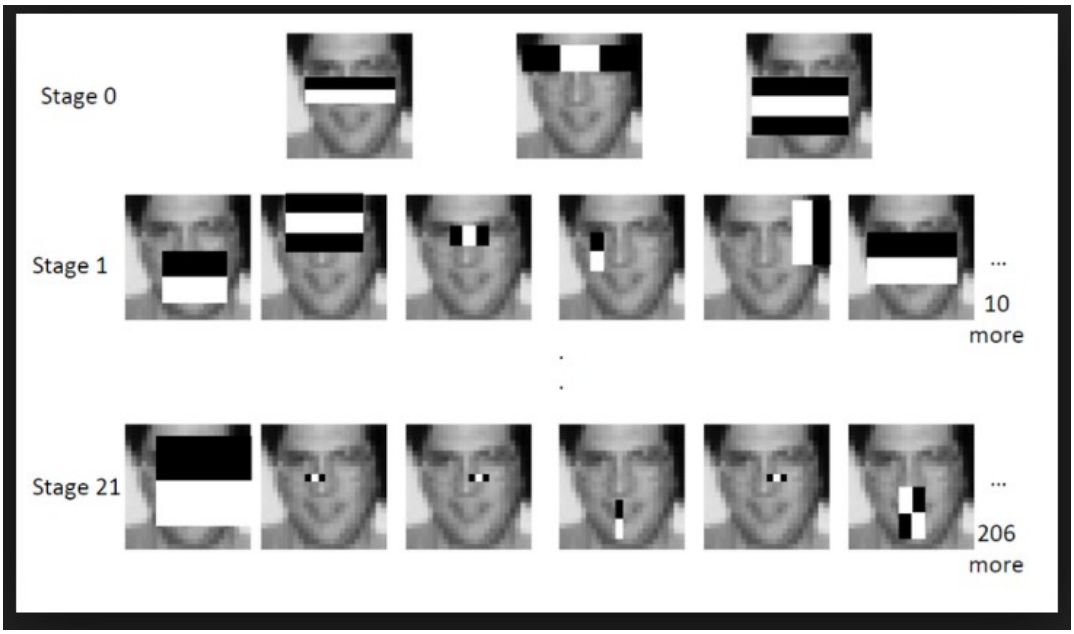


Figure 2: How Haar Cascade Classifier detects specific facial features (Pandey, 2018)

For example, a feature that detects the eye region of the face will utilize a common pixel characteristic of facial images: the eye region is darker than the nose and the cheek regions nearby. Above image is a screenshot of the eye detection feature of Haar Cascade Classifier at work.
Then, we detect the key facial structures of the face. For this process, we utilize the dlib library, which includes a pre-trained facial landmark detector. This method uses the training dataset which includes the labeled (x,y) coordinates of prominent regions of the face, and measures the distance between the input pixels to find the most probable distances between these coordinates. Below is an example of how the dlib's facial landmark detector labels coordinates for the overall facial structure.

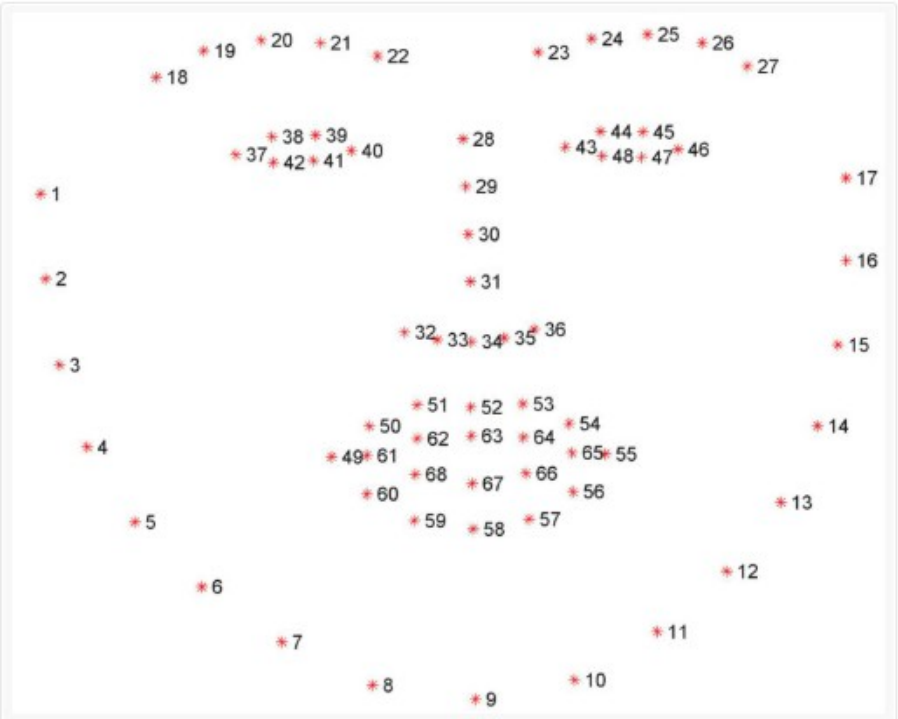


Figure 3: A sample of label coordinates for the overall facial structure (Rosebrock, 2017)

We center and scale the faces with the following parameters: face width in pixels, face height in pixels, and left eye position in coordinates (x,y). First, desired width and height of face is used to unify the size of the faces in all photos. Usually, the width and height values are the same so that the output will consist of square images.

Image lighting also relates to this cleaning issue. The different lighting of certain images will have to be normalized as well for the same reasons. Our strategy for handling this is to convert all the images to a grayscale format. We will achieve this using the image module from Pillow (PLS), a python imaging library.

Initial Algorithms Selection

- Classification and Clustering:
  Instead of feeding the entire image as an array of numbers, the image is broken up into a number of tiles, the machine then tries to predict what each tile is. Finally, the computer tries to predict what's in the picture based on the prediction of all the tiles. This allows the computer to parallelize the operations and detect the object regardless of where it is located in the image.

- KNN:
  It is by far the simplest machine learning/image classification algorithm, which makes it very easy to implement. Although KNN usually performs worse than some of the other machine learning algorithms, they are suitable for datasets with a lot of noise and large number of data points. Since our dataset has these characteristics, we are hoping to see that our data is a part of a niche group of datasets which work very well with KNN. However, KNN is known for its high computational cost and weakness for datasets with high dimensionality, so it's hard to forecast how it will perform with our data.

- SVM:
  This algorithm plots each data point in a hyperplane with N-dimensional space; value N is determined by the number of features. SVM is alone is usually enough for binary classification but for more complex analysis, SVM is usually modified using the 1AA approach, the most common SVM multiclass approach in the industry. We will modify SVM if we decide that the dataset is too complicated for SVM without modification. SVM has a broad range of implementation in the real world, most prominently used for handwritten character analysis, text categorization and stock market price prediction.

- Neural Network:
  Due to the difficulty of estimating the exact age, we are going to try developing a Neural Network Model to estimate the age to be within certain ranges. Neural network models are the frequently used mechanism for age classification already in this field. Since the human brain is the best classifier in pattern recognition as it has been trained and learned for several centuries. The main emphasis of an NN model is to extract, feed and train the computer system mechanism for pattern recognition by the help of the original nature functions of human brain.

Final Algorithm Selection

During our research to find the best model in the model selection phase, we realized that combining multiple deep learning models into a single model is popular with image analysis due to its high accuracy, achieved by a strength of a model compensating for a weakness of another model. The complex nature of image data is also very adequate for deep learning models. We selected RNN, CNN and LSTM and combined them into a single model. Further discussion of the final model is further discussed in the Model Analytics section.

Outcome Anticipation

- Our goal is to build an application which can help create a safe online environment for people of all ages to be able to use the internet without exposure to various risks lurking online.
- With our application, we are trying to implement a stringent verification system which prohibits minors getting access to explicit content and offensive speech, and also prevent cyber bullying.
- The face recognition system will take an individual's picture and will allow him/her access depending on if he/she lies in a certain age group and also it will be able to flag already created accounts by minors if there are any.

ETL and Database

Extract

In short, the extract stage in the standard ETL process is when data is typically read from a database. For the purposes of our project and due to the small sizes, a limited data formats our dataset will consist of an external database may not be necessary.

Alternatively, even though our data is small simple and easy method for storing our data could be the Google Cloud Platform. Using this method would allow us a simplistic method for storing and pulling our data. This can be easily managed through the Google Cloud Platform's command line or online console via the web.

Per the purpose of our project, to detect the relative age of individuals in photos, our datasets are large when it comes to the number of photo files we have, but small in terms of sheer file size. Our dataset of photos will consist anywhere between 34 - 38,000 separate photo files, whose overall size will only be between 48-55 Megabytes, no larger than a 5 min standard definition YouTube video.

The largest bulk of the data, which is sourced from the Indian Movie Face Database (IMFDB) The IMFDB is a popular data source for facial recognition projects due to its large volume (34,512 records) and quality of images that are selected for their high resolution and format. This data will act primarily as validation data as the labels are correctly identified. The three categories our age recognition software will categorize are "young", "middle-aged", and "old." Within the validation data there are 6,706 young pictures, 10,804 middle pictures, and 2,396 old pictures.

Introduction on IMFDB

Indian Movie Face database (IMFDB) is a quite large face image database which is consisting of **34512** images about **100** Indian actors collected from more than 100 videos. All the images are **manually** selected and cropped from the video frames related to scale, pose, expression, illumination, age, resolution, occlusion, and makeup.

IMFDB was inspired by some image recognition project, like Labeled Faces in the Wild (LFW) and Public Figures (PubFig), related to face datasets complementing them in following ways that faces in IMFDB are collected from Indian movie videos, and IMFDB is built by manual selection and cropping of video frames resulting in a large spectrum of poses develop efficient algorithms to handle pose.

IMFDB has kept the following design guidelines in mind while building the database.

- To ensure diversity in appearance of actors, movies are selected from 5 Indian languages namely, Hindi, Telugu, Kannada, Malayalam, and Bengali.
- For each actor, movies are selected such that they give wide variations in age.
- Cropping of faces: Faces are cropped with a tight bounding box. In order to maintain consistency across images, this project followed a heuristic of cropping the face from forehead to chin.
- For every image, annotation is provided for following attributes:

    - Expressions: *Anger, Happiness, Sadness, Surprise, Fear, Disgust*
    - Illumination: *Bad, Medium, High*
    - Pose: *Frontal, Left, Right, Up, Down*
    - Occlusion: *Glasses, Beard, Ornaments, Hair, Hand, None, Others*
    - Age: *Child, Young, Middle and Old*
    - Makeup: *Partial makeup, Over-makeup*
    - Gender: *Male, Female*

Transform

Transformation is the process of converting the extracted data from its previous form into the form it needs to be in so that it can be placed into another database. The transformation of data typically occurs by using rules or lookup tables or by combining the data with other data. In our case we would only need a single database to store our original photos, but transformation of the photos as well as our cleaning with take place on our machines locally or directly upon the original data stored in the database.

Our larger photo dataset of 34,000 photos will consist of only JPEG files while our second and smaller dataset which will add another layer of variability to our dataset will only consist of just under 4,000 photos in the PNG file format. In response to the differing data file types we may need to convert our second dataset of PNG photo files to JPEGs simply because it will be easier for our models to incorporate files of the same data file type.

Dataset Manipulation and Cleaning

Our primely goal is to apply these images in the dataset onto running deep learning models. Changing photo file type into a single universal format (some are jpg., and others maybe .png now). Thus, we need to make them consistent and we try and use a single format of .jpg that works for our models. Also, converting the images to a consistent grayscale pixelation format will be achieved through a simple python script applied to the entire dataset (including both validation and test data). In this section, awe apply R to convert files as needed. And as soon as we got the correct formats in all, we use Databricks to run needed models.

Load

The cropping of images is also important regarding the consistency and cleanliness of our dataset. However, the cropping of the images is often easily done during the “Load” process. To be more specific, using a filter within DeepLearningStudio (DLS) that crops all the images to a certain dimension. When using other analytical tools such as databricks, the cropping was completed with another python script as the grayscale conversion is. Besides, we’ll split our data in to 20,000 as the testing dataset; and 14,000 as the validations.

Database

According to our team research on the dataset storage session, there are basically two types SQL and NoSQL database for user to evaluate and pick a suitable one up. Theoretically, the NoSQL is the best option for storing large data of videos and images.

Since the IMFDB dataset contains the image files in all, sorting the data on the filesystem is the best recommended approach, while storing the names and paths (if any) to those files in the database might lead to performance and recovery issues as the database grows very quickly in size. We could store our images as BLOBS in Azure, yet then we have to read the BLOB and save it as a file before we use it. And it could bother us since the BLOB is going to take up as much storage as the file. Thus, based on the results of research related to database chosen, we are going to the common conclusion that it's better to store images in plain files rather than in a database.

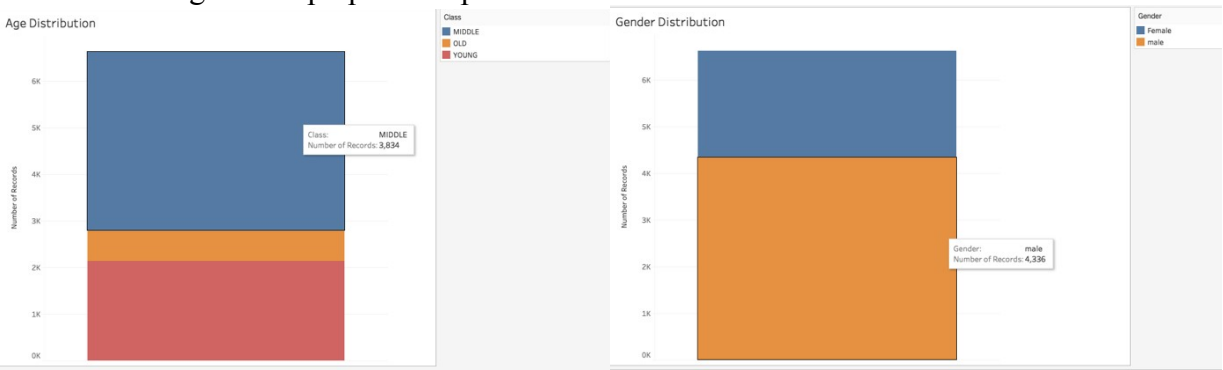
For this storage option, each team member will minimize the CPU usage, the number and the time of internet operations when uploading or downloading images. And storing the files in plain files into our own notebooks could have almost the best performance under intensive read/write workloads with this large IMFDB files. In addition, we are going to keep two copies of each image file in separate datacenters, like google could and dropbox cloud center, for backup option and recovery. This way the datacenter site isn't impacted by all of the requests needed to get those images, resulting in a more performant site for our team to own. We also utilized the deep learning studio cloud, which facilitated the analysis process by reducing computational cost associated with data loading.

In closing, our preferable and reasonable approach to storing the IMFDB images is to have them stored in our personal file system and using a datacenter platform of Google Cloud Platform to organize metadata about them (names, descriptions, etc), as well as links to the actual data such as full paths to their file names.

Data analytics & Visualization

Dataset Visualization and Preparation

With the help of tableau, we can get a holistic picture of our original dataset. According to the left plot of Age Distribution, we can see there is major meddle age class in our original dataset. Besides, in the right plot of Gender Distribution, we can get the idea that the male has a significant proportion to our original dataset. Noting that there is vague label information in the original dataset, so we were pre-processing the gender class manually and double check with the class label of age in the preparation process.



Beside, we had preparation of images for optimal input dimensions.

Original Image *Autocrop Function* *Cropped* *Grayscale Function* *Final Image*

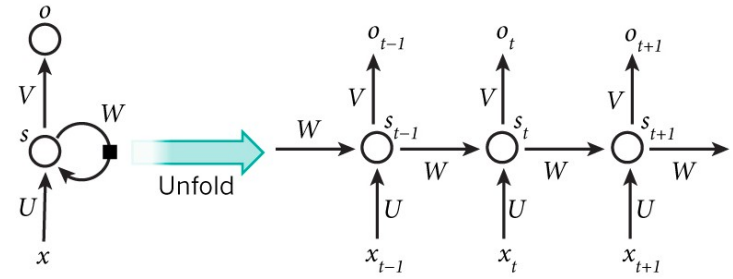
Model Analytics

CNN

Convolutional Neural Networks (CNNs) are one of the categories of Neural Networks family which has proven very effective in areas such as image recognition and classification. And Convolutional Neural Networks have been successful in identifying faces, objects and traffic signs apart from powering vision in robots and self-driving cars these days. In general, the more convolution steps we have, the more complicated features our network will be able to learn to recognize the features from the original dataset. Thus, in our data analytical model, we put layers of convolutional neural network to train our model.

RNN

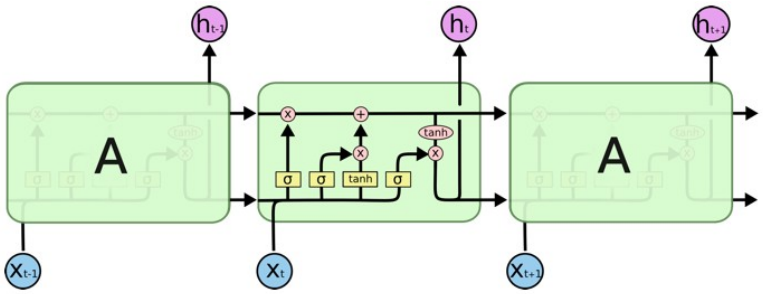
Recurrent Neural Networks (RNNs) are popular models that have shown great promise in many machine learning tasks. The purpose behind RNNs is to solve the traditional assumption problem that all inputs (and outputs) are independent of each other. Thus, RNNs is making use of sequential information in each task to improve the performance of this Neural Network Model. RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being depended on the previous computations. Another way to think about RNNs is that they have a "memory" which captures information about what has been calculated so far. In theory, this chain-like nature of neural network reveals that recurrent neural networks are closely related to sequences and lists. And this is quite a matching architecture to use for image data. Here is what a typical RNN looks like:



A recurrent neural network and the unfolding in time of the computation involved in its forward computation. Source: Nature

LSTM

LSTM is standing for Long Short Term Memory network and it is a special kind of RNN. To be more specific, LSTM is capable of learning long-term dependencies. All recurrent neural networks have a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a simpler structure than LSTM. LSTM still has the nature of this chain like structure, but the repeating module has a quite different structure. Instead of having a single neural network layer. In LSTM, it typically has four layers, and each layer is interacting in a very special way. The key to LSTM is having the cell state and gate. In a LSTM model, it has the ability to remove or add information to the cell state, and this interaction is carefully regulated by gates. Another way to think gate is that the gate is a way to optionally let information through which could positively improve the performance of the analytical model.

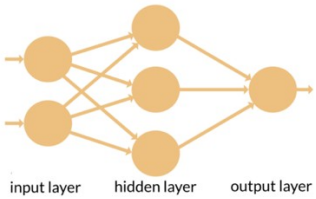


The repeating module in an LSTM contains four interacting layers.

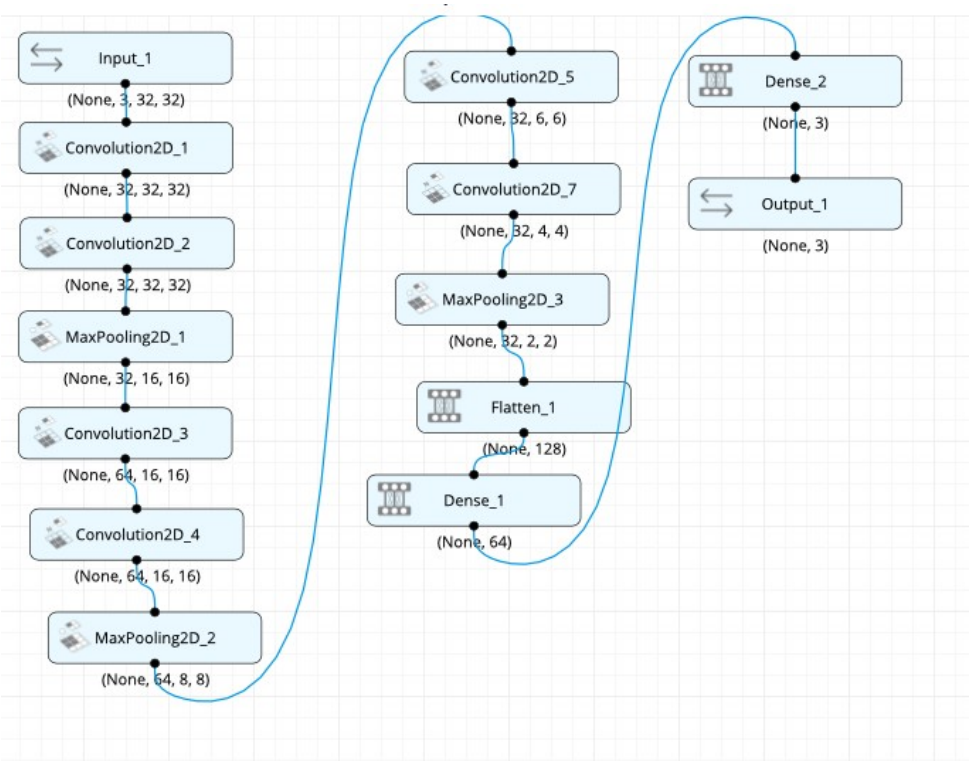
Analytics Visualization

Model presentation

The two visualizations below show the steps the data takes during the deployment of the model. The first being a general summary of how our RNN started at first, and the next being the actual model used in Deep Learning Studio for the final results.



General RNN Model



Our Final Model **Results**

After our training model process, we could achieve almost 80% accuracy with our dataset in age detection.

Our Final Model Loss and Accuracy

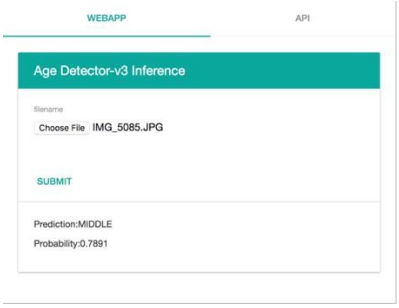
Analytical Discussion

We have observed that there are a few additional ways in which we can improve the accuracy of the model. Better preprocessing of data is expected to significantly increase the performance. For example, converting color images to grayscale would eliminate color feature which creates more noise than signal in making predictions. Also, better standardization of data through facial alignment and image cropping will not only reduce computational cost, but also increase accuracy by allowing the model to better identify key facial features.

Increasing the number of epochs could also increase accuracy, but this increases the risk of overfitting the model, which leads the model to not learn from but memorize the training data. The accuracy of overfit models is expected to perform less efficiently when the validation set includes data points that are do not significantly resemble the data points from the training dataset.

Outcome Application

With the help of Deep Learning studio, the model can be deployed via a html webpage, and we can upload any new image into our model and receive the corresponding prediction and probability of that predicted class.



Conclusion

Our model performs beyond expectations and peaked at an 81% accuracy during a deployment of 25 epochs, lasting roughly 1.5 hours. The first few deployments allowed us perspective into how to better the model for a more efficient and appropriate usage. Although our model can predict whether an individual is in the “middle” or “old” age ranges, its accuracy when categorizing “young” people is less efficient. After further investigation, it was revealed that although there is a substantial amount of “young” labeled images, very few of these images are of children (less than roughly 16 years of age.) This lack of represented children in the image dataset presents a problem when the model tries to evaluate other pictures of children.

A remedy we will implement will be scraping the web for images of younger people, properly cleaning the images as we did the rest of the data, and then finally adding it to the training dataset. Hopefully this new infusion of crucial data will help the model properly identify young people as that is arguably the most important age group to be able to accurately identify.

Personal Project Experiences

This project was by far the hardest project I have ever taken a part of since I joined the MSIS program. The most notable difference between the previous projects from other classes to this one was the flexibility of the requirements that really incentivized all the students to push their limits. Most of our group members had no experience working with image datasets prior to taking on this project. Every step of the analytics process, from data extraction to the modeling, was different and far more challenging due to the complex nature of the image data.

When we first decided to choose face analysis as our project, we initially attempted to estimate the exact age of the individuals in images. However, training process proved to be too difficult, and as a result, accuracy of the model suffered. We then decided to classify images and decided to create three classes: “young”, “middle aged” and “old”. Since we had complete control over the direction of the project, when we decided to pivot, it resembled the process I go through when I take on

Data cleaning process was extremely refreshing as well. In previous projects, data cleaning process usually consisted of deleting missing values, removing special characters, changing data types and joining tables. This time, we were doing something completely different. We used python libraries to estimate the coordinates of important facial features and cropping images around the face so that important pixels are not lost during the cleaning process. We also standardized the image size to attempt to increase the accuracy of the model.

```
import argparse
import dlib
import cv2


image = "C:\\Users\\YOGA-700\\eclipse-workspace\\SAT\\zelda.png"
face_detector = dlib.get_frontal_face_detector()
detected_face = face_detector(image)
w,h = 160

for face_rect in detected_face:

    # First crop the face
    left = face_rect.left()
    top = face_rect.top()
    right = face_rect.right()
    bottom = face_rect.bottom()

    new_face_rect = dlib.rectangle(0, 0, right-left, bottom-top)
    cropped_image = image[top:bottom, left:right, :].copy()

```

Figure: face detection using OpenCV and dlib

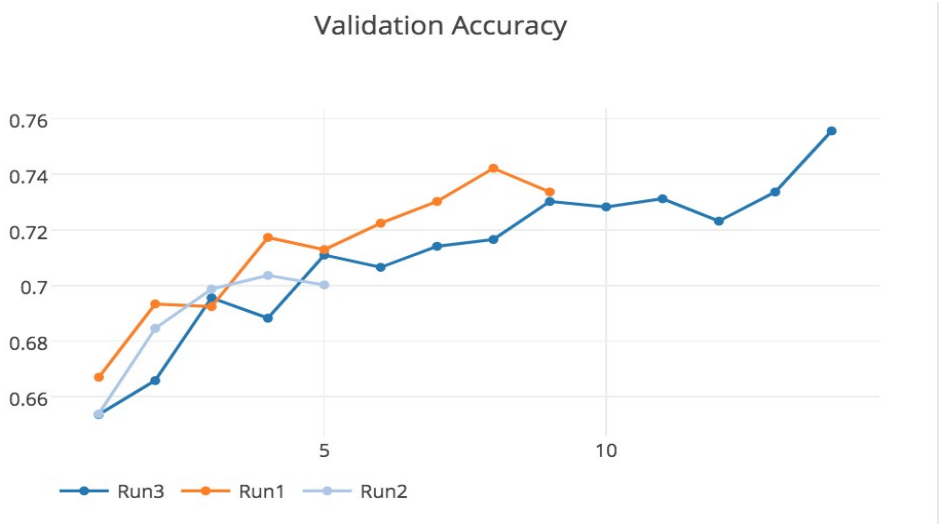
The data cleaning process involved using OpenCV and dlib to detect important facial features and Auto-Crop to crop the images around the face. Then we turned all our images to grayscale to reduce the complexity of the data. Pixels are reduced to 2-dimensional data when colors are removed, and since color is not an important feature in our model, we benefited significantly by removing colors. Finally, we partition the data. The original data is into training and testing data, with 70:30 split ratio.

Selecting appropriate models was also a challenge. We had to go through publications and referee journals to find the most suitable model for this project. In the end, we decided to combine multiple deep learning models into a single model, as it is a common industry practice. Combining models allow a strength of one model to compensate for a weakness of another model, which increases accuracy. We were able to achieve 80% range accuracy in our final model, which I personally believe is high enough to be implemented as a solution to strengthen internet content access control, but could be improved, especially because the model only had 3 clusters.

Model Code

```
def get_model():
        aliases = {}
        Input_1 = Input(shape=(3, 32, 32), name='Input_1')
        Convolution2D_1 = Convolution2D(name='Convolution2D_1',nb_col= 3,nb_filter= 32,border_mode= 'same'
        Convolution2D_2 = Convolution2D(name='Convolution2D_2',nb_col= 3,nb_filter= 32,border_mode= 'same'
        MaxPooling2D_1 = MaxPooling2D(name='MaxPooling2D_1')(Convolution2D_2)
        Convolution2D_3 = Convolution2D(name='Convolution2D_3',nb_col= 3,nb_filter= 64,border_mode= 'same'
        Convolution2D_4 = Convolution2D(name='Convolution2D_4',nb_col= 3,nb_filter= 64,border_mode= 'same'
        MaxPooling2D_2 = MaxPooling2D(name='MaxPooling2D_2')(Convolution2D_4)
        Convolution2D_5 = Convolution2D(name='Convolution2D_5',nb_col= 3,nb_row= 3,activation= 'relu' ,nb_1
        Convolution2D_7 = Convolution2D(name='Convolution2D_7',nb_col= 3,nb_row= 3,activation= 'relu' ,nb_1
        MaxPooling2D_3 = MaxPooling2D(name='MaxPooling2D_3')(Convolution2D_7)
        Flatten_1 = Flatten(name='Flatten_1')(MaxPooling2D_3)
        Dense_1 = Dense(name='Dense_1',output_dim= 64,activation= 'relu' )(Flatten_1)
        Dense_2 = Dense(name='Dense_2',output_dim= 3,activation= 'softmax' )(Dense_1)
```

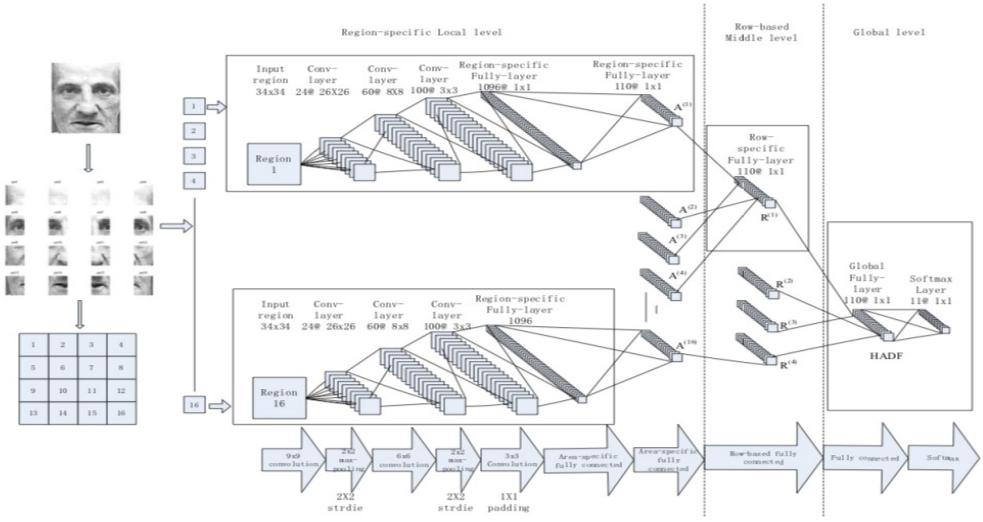
We also attempted to write our program in python using the jupyter notebook. The data cleaning process was relatively easy but incorporating multiple models into one without the help of GUI software was difficult, given the time constraint. However, we decided to individually take on the challenge of writing out the code. The above is the method that constructs our combined deep learning model in python.



We were quite confident with the robustness of our model. Many image recognition and image analysis programs utilize deep learning models. However, we were able to further improve the accuracy of our model. The improvements came from the data cleaning process. We explored with various data cleaning processes to improve accuracy, when we realized that better cropping and face alignment improved our accuracy.

These photos were the photos we used to test the influence of quality of data to the performance of our model. Excluding noise from the data by removing background pixels increased our model by more than 8%. This is also easier than making improvements on the model, so we learned that spending some extra time on the data cleaning process is extremely important to the model performance.

When we were communicating our findings with the class, I learned some of the most important lessons about data analytics. When people are not engaged in the presentation, the findings derived from data analytics loses its value, because people are disengaged, and are not ready to absorb the information. In order to engage the audience, especially the people on the business side that are not as familiar with the technical aspects of data analytics, it is important to use visualization to deliver information in an engaging way.



By showing visuals that allows the audience to quickly understand the context of the process without reading sentences during a presentation is critical in keeping the audience engaged and focused on what the speaker is saying.

References

Chen, B. C., Chen, C. S., & Hsu, W. H. (2015). Face recognition and retrieval using cross-age reference coding with cross-age celebrity dataset. *IEEE Transactions on Multimedia*, *17*(6), 804–815.

Eidinger, E., Enbar, R., & Hassner, T. (2014). Age and gender estimation of unfiltered faces. *IEEE Transactions on Information Forensics and Security*, *9*(12), 2170–2179.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *32*(9), 1627–1645.

Fu, Y., & Huang, T. S. (2008). Human age estimation with regression on discriminative aging manifold. *IEEE Transactions on Multimedia*, *10*(4), 578–584.

Goswami, A. (2018, February 08). Intro to image classification with KNN. Retrieved April 5, 2019, from https://medium.com/@YearsOfNoLight/intro-to-image-classification-with-knn-987bc112f0c2

Rosebrock, A. (2017, April 3). Facial landmarks with dlib, OpenCV, and Python. Retrieved April 5, 2019, from https://www.pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/

Pandey, P. (2018, December 20). Face Detection with Python using OpenCV. Retrieved April 5, 2019, from https://www.datacamp.com/community/tutorials/face-detection-python-opencv

Raw data 1: link for 34512 images of 100 Actors: https://towardsdatascience.com/age-detection-of-indian-actors-with-deep-learning-studio-da10bc91fc57

Shankar S, Moula H, Parisa B, Jyothi G, Menaka K, Radhesyam V, Vidyagouri H, JC K, Raja R, Rajan, Vijay K and C V Jawahar. (2013). Indian Movie Face Database: A Benchmark for Face Recognition Under Wide Variations. National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG).