4 Literature Review

Age plays important role in the social interaction. Age detection is the most attentive problem in the computer vision. The age of the person can be identified by his/her facial features. Humans can understand the approximate age of the person face by looking at the facial features. Its very simple for human, but its way more challenging for computers to understand the age from the facial features. Computer Vision systems are working on the age classification problems from many years. The previous approaches relied the facial feature dimensions comparison, i.e. calculating the length between the eye brows and trying to get some biological references for solving the problem.

Also, it's not possible always to get well-aligned and good quality images in the real world. For example, the images taken from the surveillance camera will not always have same facial alignments. It may differ with images and it became challenging for classification system to find the age from such a facial image.

Recently developed machine learning system has tried to overcome some of the such challenge. The deep Convolutional Neural Network (CNN) is been widely accepted approach for classification of images. CNN uses the concept of Artificial Neural Network for age classification. In the approach provided in [1], the researcher has used CNN for age and gender classification. In this approach, the dataset used was the Adience benchmark for age and gender classification of unfiltered images. They showed that despite the images were having many different alignments, different sizes, different resolutions and variety of people from different background, the model developed was able to classify age and gender by using Convolutional Neural Network (CNN). They provided the basic CNN architecture for age classification.

In [2], the researcher has tried to improve the existing model in [1] by reducing the number of parameters, increasing depth of the network and modifying the level of dropout used. The result was that it caused the system performance to decrease as compared to earlier architecture. Also, the researcher tried to do age classification based on gender, means using different classifier for male and female. In [3], taking the advantage of the architecture proposed in [1] and [2], TensorFlow was used to train the model.

The research work provided in [1] and [2] formed the basis of the architecture on age classification using CNN of misaligned, low resolution and natural images. The architecture defined in both approaches is been used in our proposed approach. In [2], it was tried to improve the architecture by changing parameters but didn't work as expected. By taking ahead the approaches in [1] and [2], we are trying to improve the training speed because Adience dataset is unstructured and requires lot of time for training. We are making using of TFRecords to improve the training speed and trying to reduce parameters in order to improve the existing network. As compare to approach in [3] we are using TensorFlow in more efficient way and trying to improve the model.

5 Problem description and Architecture

The problem we are talking is about the age classification using Convolutional Neural Network (CNN) and improving the existing model and training speed of the model. We are using Adience dataset for training model as it contains variety of images.

5.1) Convolutional Neural Network (CNN)

Neural network idea is to take large number of training examples as input and develop a system which can learn from the training examples. Convolutional neural networks (CNNs) are the current state-of-the-art model architecture for image classification tasks. CNN is a class of deep, feed-forwarding artificial neural networks, most commonly applied to analysing visual imagery. CNN uses different convolutional layers for classification. The first layer is the input layer where the dataset is fed to the network. The middle layers are called hidden layers, which are responsible for classification. The last layer is the output layer. CNN usually consists of the convolutional, pooling, full-connected layers. Figure 5.1 shows basic CNN architecture.

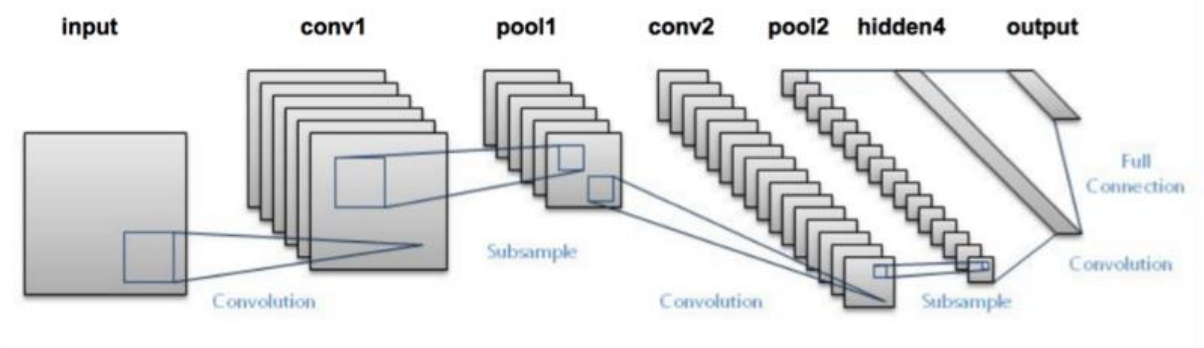


Fig 5.1 Basic CNN architecture [4]

Layers in CNN:

a) Convolutional Layer: Convolutional layers apply convolutional operation to the input images. The input layer usually connects to the convolutional layer. In first convolutional layer, each pixel contributes to single neurons. For instance, if there is 512x512 image, then in first layer there are 2621444 neuros. Convolutional layer convolutes at every pixel and calculates the average weight. The filer usually of odd size (3x3, 7x7) are convoluted with or without zero padding. Stride is also provided to convolutional layer. Stride controls how filter convolves around the input layer. The amount by which filter shifts is the stride.
b) Pooling layer: Pooling layer does the work of down sampling. So, it is some time also called the down sampling layer. This basically takes filter and a stride of same length. It then outputs the, maximum number in every subregion that the filter convolves around.
c) Fully-connected layer: Full connected layers connect every neuron in one layer to every neuron in another layer. Sometime the drop out layer is also applied to avoid the problem of overfitting.
d) ReLu (Rectified Linear Units Layer): The purpose of this layer is to add non-linearity to a system as the convolution is basically is just doing linear operations. Its just element wise multiplication and summation.

5.2) Architecture for age classification

For age detection implemented Convolutional Neural Network having

- 3 Convolutional Layers
- 2 Dense Layer and dropout layer

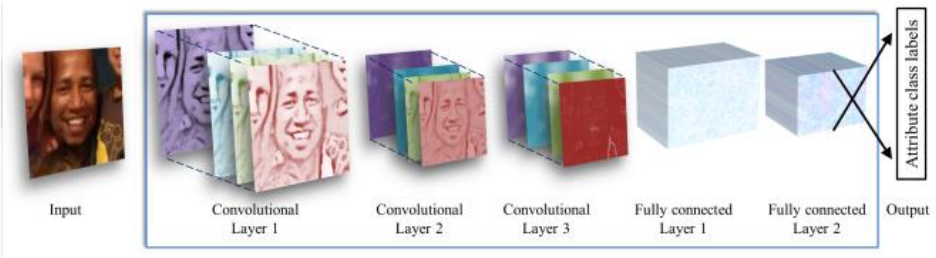


Fig 5.2 Proposed Architecture [3]

5.2.a) Convolutional Layer 1

- Input Image 227*227
- 3x7x7 filter shape,
- 96 feature maps.
- Stride of 4 and same padding.
- Followed by: ReLU activation function, MaxPooling and Normalization

The output layer produced by convolution layer 1 has a shape

[batch_size, 28, 28, 96] *batch_size is decided during run-time by TensorFlow

5.2.b) Convolutional Layer 2

- Input Image 28x 28x 96 from convolution layer 1
- 96x5x5 filter shape,
- 256 feature maps.
- Stride of 2 and same padding.
- Followed by: ReLU activation function, MaxPooling and Normalization

The output layer produced by convolution layer 2 has a shape

[batch_size,14, 14, 256] *batch_size is decided during run-time by TensorFlow

5.2.c) Convolutional Layer 3

- Input Image 14x 14 x 256 from convolution layer 1
- 256x5x5 filter shape,
- 256 feature maps.
- Stride of 2 and same padding.
- ReLU activation function, MaxPooling and Normalization

The output layer produced by convolution layer 1 has a shape

[batch_size,14, 14, 256] *batch_size is decided during run-time by TensorFlow

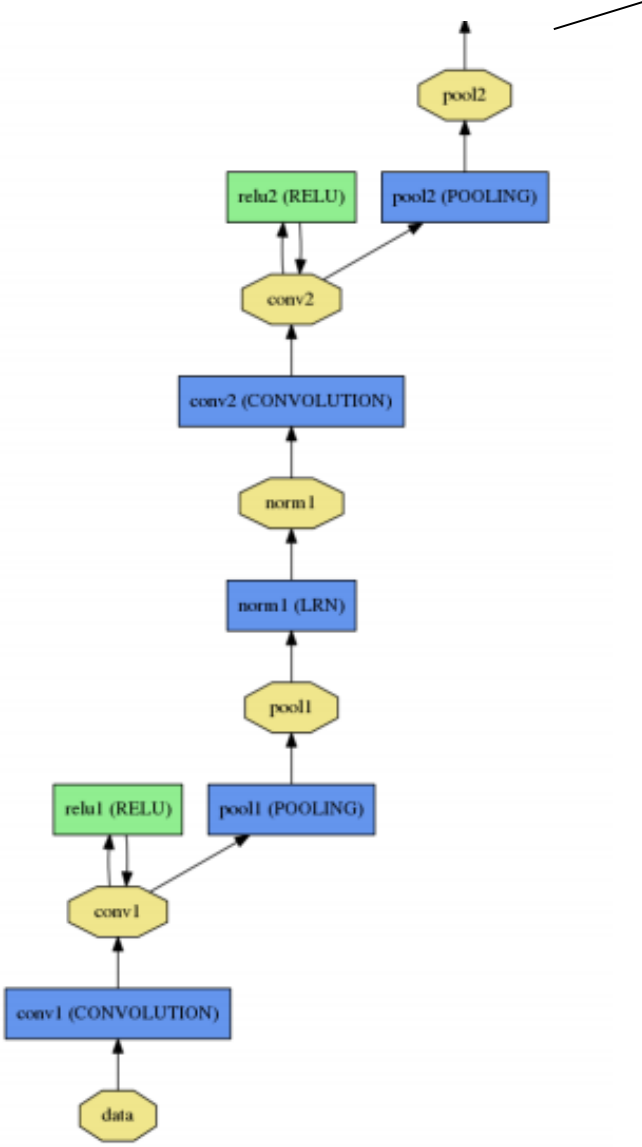
5.2.d) Dense Layers

Dense layer takes the input from the convolution layer 3. In this architecture two dense layers are used. The artificial neural network is more used to problem of overfitting. Overfitting of network means that the network performs well for training examples but not for other examples. So, to avoid the problem of overfitting the dropout layer is added after every dense layer. The layer from out a random set of activations in that layer by setting them to zero. Network should be able to provide the right classification or output for a specific example even after some of the activations are dropped.

- The output of the layer 3 is fed to dense layer.
- First two dense layers having 512 neurons
- Third layer is the important as it is will return raw values from predictions and it has number of neurons equal to number of classes.
- Each dense layer is followed by drop out layer with rate of 0.5

Finally output of the last fully connected layer is fed to a soft-max layer that assigns a probability to each class.

Continued next page



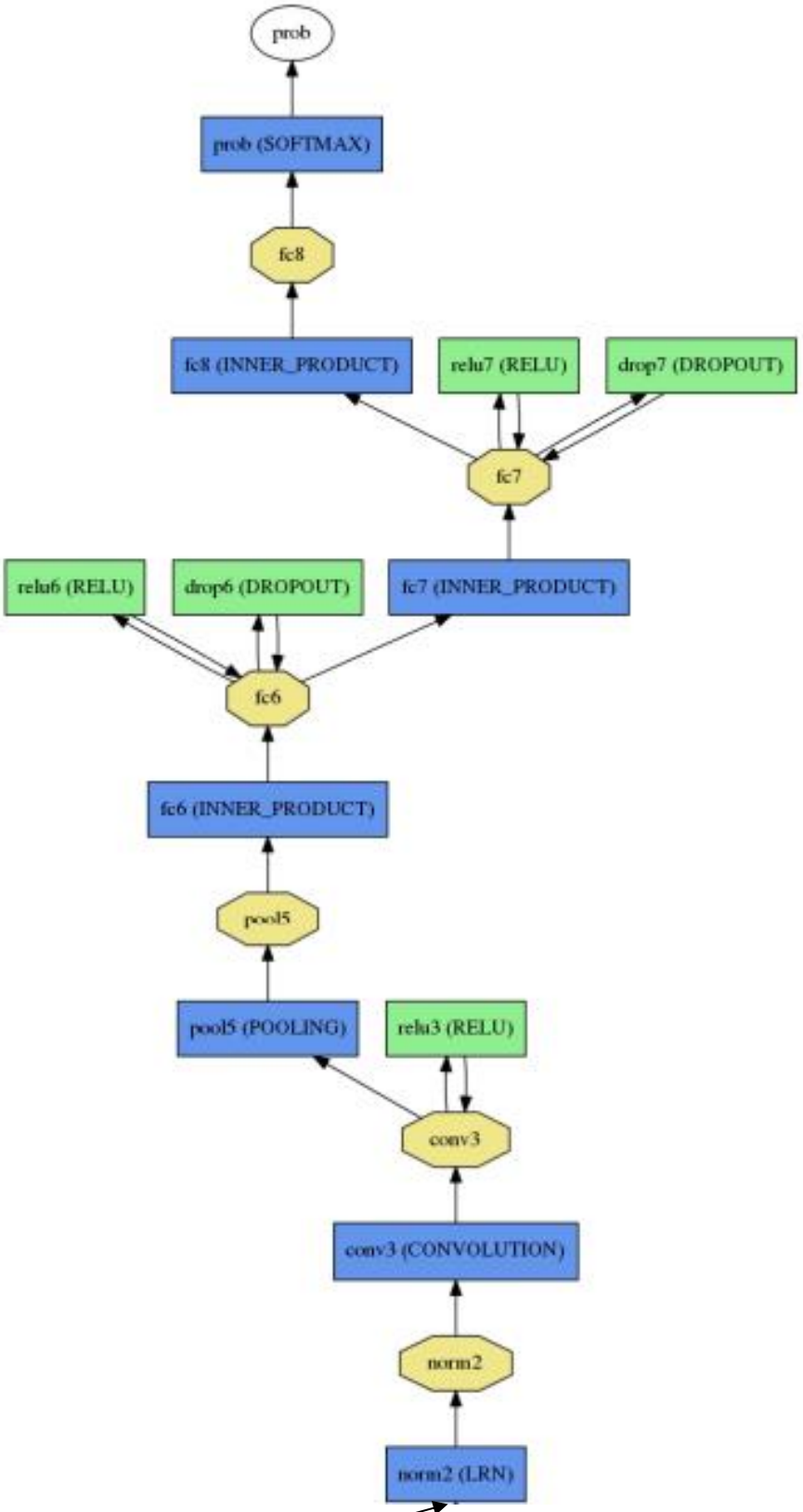


fig 5.2 (c) Flow chart of Architecture

Continued

5.3) Dataset

The dataset used is Adience Dataset which contains unfiltered faces for age classification. The sources of the images included are Flick albums, assembled by automatic upload from iPhone 5(or later) smart phone devices, and released by their authors to the general public under the Creative Common (CC) license [5]

The entire Adience collection includes images around 19k of 2,284 subjects. Each image was having age range associated with it. The dataset was having images of younger generation. Its bit obvious as the social network is been used more by younger generation. For simplicity we reinitialize the dataset with labels to each age range by using Excel operations. Using the below formula, we were able to refine the dataset with the single labels.

```
cc=IF(C2="(0,2)",1,IF(C2="(4,6)",2,IF(C2="(8,12)",3,IF(C2="(15,20)",4,IF(C2="(25, 32)",5,IF(C2="(38, 43)",6,IF(C2="(48, 53)",7,IF(C2="(60, 100)",8,0)))))))))
```

The table shows the total number of images in each category.

0-2 (1)	4-6 (2)	8-13 (3)	15-20 (4)	25-32 (5)	38-43 (6)	45-53 (7)	60- (8)	Unlabelled (0)
1427	2162	2294	1653	4897	2350	825	869	3023

Table 5.3 Ages and number of images in that group

The challenges faced with this dataset were that the many images had no labels. It was very difficult to know the exact count and which images has which labels. Also, dataset contained many parameters which were useless for us, as we were more interested in ages. So some pre processing was done to drop this extra parameters from the dataset. The biggest problem we were going to face that the age distribution is skewed towards younger age. This raised our concern over accuracy of the model.

6 Training and Testing

The training and testing phase include dataset preparation for feeding the model and implementation of proposed architecture. Our first aim was to improve the training speed. We used TFRecords for improving the speed of training.

6.1) Dataset preparation

TFRecord file format is a simple record-oriented binary format developed by the Google that many TensorFlow applications use for training different models. We first focused on converting the dataset into TFRecord format. It was crucial and challenging task to convert the dataset to the TFRecord format. We developed the python script to stored the dataset (Audience Benchmark Dataset) in TFRecord format and stored images in byte format and labels in int64 format.

```
feature = {
    'image_raw': _bytes_feature(img.tostring()),
    'label': _int64_feature(label)
}
```

For any successful implementation its very necessary that the dataset is divided into training, testing and evaluation. It is the important step in the machine learning. So we divided that dataset into **60% training 20% testing and 20% validation.**

Advantage of the TFRecord

- TFRecords can handle more complex data neatly
- It has faster reading speed when deep neural network architecture is complex
- The tedious work of splitting data into training, validation and testing , shuffling the data can be handel well by TFRecords

Writing to TFRecord:

Writing the dataset to TFRecord includes converting the data to featureSet then feature set to example then to serialized example and finally to TFRecord

Data -> FeatureSet -> Example -> Serialized Example -> TFRecord.

Writing TFRecord

Reading from TFRecord

This is just the reverse of the writing steps.

TFRecord -> SerializedExample -> Example -> FeatureSet -> Data

6.2) Training

The CNN model proposed in this approach was build using the TensorFlow. TensorFlow has its own advantages. It tries to improve the performance of the model. The batch_size of the data is been kept dynamic and it decided during run-time.

The three files from the dataset preparation (train.TFRecord, test.TFRecord, val.TFRecord) was provided to CNN.

The CNN model was trained for 1000 steps. After every step the loss was calculated. Loss function measures how closely the model's predictions match the target classes. To find loss we used softmax cross entropy function.

Figure 6.2a: Graph of the loss in first 1000 steps

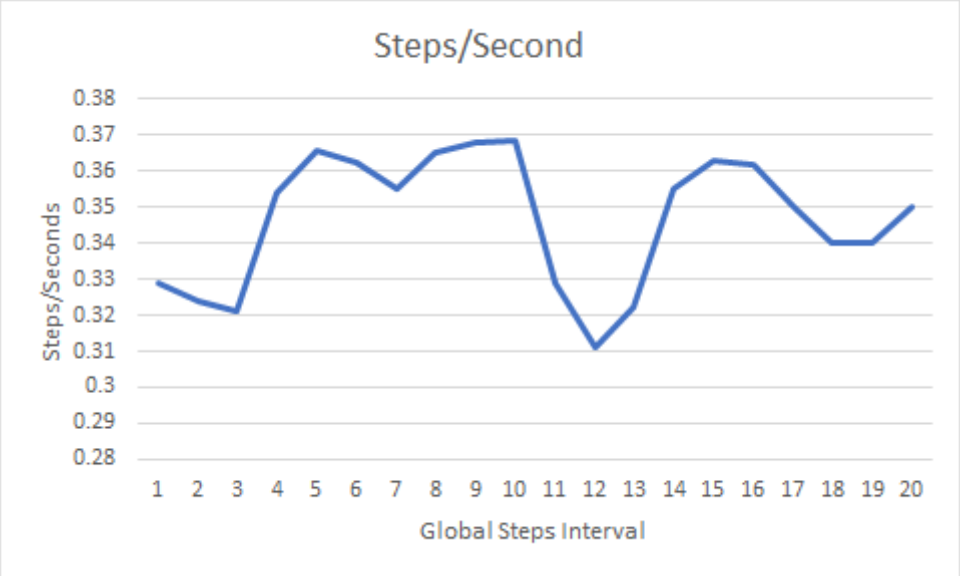


Fig 6.2b: Training Rate

Figure shows steps/second at every Global Step. I was using i7 processor with 8GB RAM. The steps may different from system to system. As it depends on computational power of the system.

7 Result

The model was trained with steps of 10000 and repeated 1000 times to get accuracy around 65%-75%. Tried to improve the accuracy by adding one more convolutional layer, changing the number of neurons in dense layer and reducing dropouts. But I was not able to improve the accuracy. In fact, making changes in the architecture sometime caused the accuracy to drop to the much undesirable range.

Example 1:

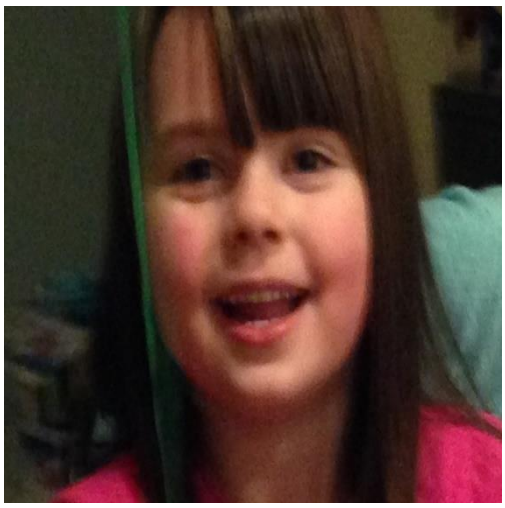


Image with label =2 (Age Group 4-6)

The input to CNN was a TFRecord file. This is how TFRecord file look if printed in JSON format.

```
features {
  feature {
    key: "image_raw"
    value {
      bytes_list {
        value: "\307\347\376\310\347\376\310\347\376\310\352\3
      }
    }
  }
  feature {
    key: "label"
    value {
      int64_list {
        value: 2
      }
    }
  }
}
```

This are the probabilities obtained from the CNN model.

probabilities = [[
0.00460506 -- > Label 0
0.06160506 -- > Label 1
0.16631052 -- > Label 2
0.03797467 -- > Label 3
0.12801285 -- > Label 4
0.11332455 -- > Label 5
0.05861919 -- > Label 6
0.03009178 -- > Label 7
0.03315931]] -- > Label 8

It can be seen the image has max probability 0.16631052 with label 2 i.e age is in the range [4-6] Ref: Table 5.6

Example 2:

Image with Age Group 22

probabilities = [[0.2194914 0.05463617 0.09182423 0.04252291 0.13438655 **0.2606217**
0.11845216 0.03707999 0.04098488]]

It can be seen the image has max probability **0.2606217** with label 5 i.e age is in the range [25-32] Ref: Table 5.6

Example 3:

Image with label 8 (Age 60 and above)

probabilities = [[0.14710806 0.04152571 0.04801511 0.02708164 0.06386861 0.15110134

0.43744025 0.03705486 0.04680435]]

It can be seen the image has max probability 0.43744025 with label 6 i.e age is in the range [38-43] Ref: Table 5.6

As the dataset is skewed towards the younger age, it may be the cause of the misclassification of older person images.

References

[1] :Hassner, T. (2018). Age and Gender Classification Using Convolutional Neural Networks. [online] Available at: https://talhassner.github.io/home/publication/2015_CVPR

[2] Ekmekji, A. (2018). Convolutional Neural Networks for Age and Gender Classification. [online] Cs231n.stanford.edu. Available at: http://cs231n.stanford.edu/reports/2016/pdfs/003_Report.pdf

[3] https://github.com/dpressel/rude-carnie

[4] https://hackernoon.com/visualizing-parts-of-convolutional-neural-networks-using-keras-and-cats-5cc01b214e59

[5] https://talhassner.github.io/home/projects/Adience/Adience-data.html#agegender