Capstone Project Report

Age Detection Using Convolutional Neural Networks

Karthik Ramesh - 2nd Sep 2018

Definition

Project Overview

Deep learning has definitely come leaps and bounds in the recent years finding its applications across different domains such as facial detection, medical diagnosis, auto completion in texts, image recognition, self-driving cars, recommending products based on historical data and much more. The recent launch of Google Assistant can be thought of one of the great examples of Deep learning.

Convolutional Neural Networks (CNN) is a recent revolution in deep learning that deals with images. Unlike neural networks where the input is vector, for CNN the input is multi-channeled images. CNN, similar to Neural Networks, is made up of neurons with learnable weights and biases. The CNN architecture on a high level receives inputs (in terms of pixels of images), takes a weighted sum of it through each layer followed by corresponding activation function defined and then passes it to the next subsequent layer. This process continues till the output layer where a node will be defined for each possible class that the algorithm is trying to predict for classification problem or the approximate value for the regression problem.

The real world datasets are of lesser quality, different positions, sizes and far from perfection. The CNN should be able to perform well with these datasets and only then it can be deemed fit for real world applications. The academic paper that was also the motivation for the project is "Indian Movie Face Database: A Benchmark for Face Recognition Under Wide Variations" [1].

Problem Statement

The problem statement is to identify the age of the actor/actresses from the image supplied to the CNN architecture. The network is trained from various images that have difference is sizes, shapes, quality and angle. The network maps the image to one of the three classes - Young, Middle and Old. The input images are captured from videos and hence the quality of images vary widely. The CNN should be able to identify and extract the features from these images with high degree of variability in terms of scale, pose, expression, illumination, age, resolution, occlusion, makeup and categorize the images to proper age bucket. The challenges in the mere collection of this dataset and the appicability and exploration of potential of deep learning on such datasets would be really interesting.

Metrics

The accuracy metric of Keras is used to evaluate the model. The data is split between training and validation. The validation set is not exposed for the training and is used to test the model during the training process. This gives an overall accuracy of the model when provided with samples that the model has not seen before.

Accuracy is defined by total number of correct classifications divided by the total dataset population.

Accuracy = (TruePositive + TrueNegative) / (TruePositive+TrueNegative+FalsePositive+FalseNegative)

This accuracy is calculated by the Keras during the training process using the validation data supplied using validation split.

Analysis

Data Exploration

The dataset for this problem is obtained from the ongoing datahack that is being conducted by Analytics Vidya (https://datahack.analyticsvidhya.com/contest/practice-problem-age-detection/ (https://datahack.analyticsvidhya.com/contest/practice-problem-age-detection/)). This is a subset of data obtained from ndian Movie Face database (IMFDB). Indian Movie Face database (IMFDB) is a large unconstrained face database consisting of 34512 images of 100 Indian actors collected from more than 100 videos. The images are manually selected and cropped from the videos that make it high degree of variability in terms described in the problem statement earlier. The IMFDB also provides detail annotation for every image in terms of age, pose, gender, expression and type of occlusion that may help other face related applications [1].

The dataset that will be used for this project consists of 26742 cleaned and formatted images with 9906 images in train and 6636 images in test. The attributes of this dataset are the Unique ID of the image and the age bin - Young, Middle and Old. There are total of 10805 images in Middle class, 2397 images in Old class and 6707 images in Young class.

Model Selection and Justification

Traditional Neural Networks gets complicated when applied to the computer vision problems involving decoding information from images or trying to predict the classes based on the images. When we convert the images to the raw pixels to be able to supply as an input vector to the traditional neural networks, the spatial information, which is critical for images, is lost. Having said that, deep neural networks were still able to provide a decent accuracy for image problems constituting of lesser dimensions. But as the volume and dimensions get bigger, the number of values at each layer gets huge and would require heavy computational power to process.

The Convolutional Neural Network (CNN) helps to bridge this gap. The main building block of CNNs is Convolutional Layer that consists of series of convolutional filters(referred to as kernels). The kernels can be of any size smaller than the original image itself in dimensions. The kernels help to capture a specific phenomenon such as sharpening, blurring, edge detection and much more. Each convolutional filter will help to identify specific feature/edge/shape that can activate that node and propogate the information to the next set of layers. For instance, a 64 *64* 3 image is 64 *64 size and has 3 channels - Red, Green and Blue. The kernel can be of size 5* 5 * 3 which will be convolved the complete image across and down. At each position, the dot product between the image and the filter is taken to get the scalar result.

The activation functions at each layer help to introduce non-linearity into the equation and propogate the output to the next layer. The stack of convolutional layers identify complex patterns at the end as each subsequent layer operates on the output of the previous layers. CNNs identify complex patterns and shapes and use that information to classify the images into appropriate bucket. CNNs are arguably one of the best techniques available for computer vision problems and the efficiency of the technique in such real world images will be an area of potential interest.

Benchmark Model

The Benchmark Model consists of following arrangement:

Input Layer -> Flatten -> Dense Layer -> Output Layer

The Input Layer accepts the vector of pixels as the input. In case of 64 * 64 input image, the Flatten layer will convert the image to vector of size 4096. There are two dense/fully-connected layers similar to neural networks. These fully connected layers, as the name implies, will have connections to all the activations in the previous layer. The final layer consists of three neurons each specific to one of the classes - Young, Middle, Old.

The softmax activation function is used to activate the neuron with the highest probability among the three classes. As explained earlier, one of the drawbacks of applying traditional neural networks to the image classification problems is it will not be able to scale efficiently for images with higher volumes. For example, if the image is 256 * 256 size, the size of the flattened vector would be 65536. Adding few dense layers after that would require heavy computation power and weeks of computation.

The accuracy obtained from our benchmark model described above is 57% and the expectation is the CNNs should be able to perform well and provide better accuracy.

Methodology

Data Preprocessing

The preprocessing techniques involved uniform resizing of images and the ability to gray-scale/colour images to understand network's performance under different scenarios.The images from the database widely vary in size and shape, brightness, quality and viewpoints. The images have to be resized so that we have same dimensions to deal with. Most of the image detection algorithms use normalization to achieve better standardization and resolve any bias during computation which is also a good technique to build better models.

The cv2 package is used to perform this data pre-processing. The main idea of this project is to throw the challenge of images with varying quality to the CNN network and witness how it is able to come up with predictions/classifications. The resizing and standardization techniques are generally across the CNNs in practice and hence the same idea is followed.

Implementation

The objective of this implementation is to overcome the below mentioned challenges and come up with a reasonable accuracy with minimum resource consumption so that it can be widely adopted.

Viewpoint variation: A single object can be oriented in many ways with respect to camera Scale Variation: Variation of size in the real world Deformation: Deformation of real entities in the real world. Occlusion: Images can be occluded, only few pixels are visible. Brightness: There can be variation in the brightness which might make it hard to identify images. Intra-class variation: Intra-class variations within each class(ex., dog. There are lot of varieties of dogs).

The architecture that is proposed is below:

Input -> [Conv2D(100)+BatchNorm] -> [Conv2D(200)+MaxPool+BatchNorm] ->
[Conv2D(300)+MaxPool+BatchNorm] -> Flatten() -> FC(200) -> Dropout(0.2) -> FC(30) -> FC(3)

ConvNet architecture is in the simplest case a list of Layers that transform the image volume into an output volume (e.g. holding the class scores). The layers mentioned above are explained below

ConvolutionLayer:

The Conv2D(number of filters/kernels) represents the convolutional layer along with the number of kernels used. Each kernel is applied in a sliding window fashion across the input image to extract the feature that kernel is looking for. The feature can be edge, sharpening, contrast etc., All the kernels are of 2 * 2 size. The other important parameters to be defined in this layer - kernel size, strides, activation and padding. Each of these are explained in subsequent sections.

Strides:

Strides define the number of pixels to shift across and down during the convolution operation. All the strides are (1,1) that implies to shift 1 pixel across and 1 pixel down after each convolution operation. Higher the stride, lesser the volume of output image.

ReLU Layer:

The activation function used is RELU that stands for Rectified Linear Unit. This activation function introduces non-linearity into the system that was performing linear computation in the convolution layers. ReLU performs significantly better compared to tanh and sigmoid functions that were used predominantly in earlier models. ReLU layer applies the function R(x) = max(0,x). This transforms all the negative activations to zero and linearly increases with the positive activations.

Padding:

The padding parameter is used to provide the zero padding to the borders to include the edges of the original image. If the edges do not contain lot of information, it is safe to omit the information by providing padding=valid. But in our case, as there is a factor viewpoint variation, sometimes the edges may contain useful information and hence padding=same is provided. In general, setting zero padding to be P=(F−1)/2 when the stride is S=1 ensures that the input volume and output volume will have the same size spatially.

Pooling:

MaxPooling is used after convolution layers to reduce the dimensionality of the input. The max pooling operates on the neighbouring set of pixes defined by the pool size and take the maximum pixel only. This seems to work really well in experiments as most of the time the pixel with larger values contain most of the useful information that when reproduced could provide a significantly similar version of the original image in smaller dimensions. Pool size of (2,2) is quite standard and the same is followed here. Pooling layers also move in strides that are defined similar to strides in convolution layer.

One of the primary advantages of CNNs is all the neurons in the network share the same weights. That will tremendously reduce the number of parameters required otherwise. This loops to the point that was made on the benchmark model about scalability. CNNs are highly scalable to images with higher volume and can perform better than traditional neural networks for computer vision problems.

GPU:

I have rented a GPU in FloydHub to get this model executed for the input images. The model, although comparatively less computationally intensive than tranditional neural networks, is still resource intensive to be executed in commodity hardware with normal configuration. The GPU that is used for this execution is of below configuration.

Standard GPU Tesla K80 · 12 GB Memory 61 GB RAM · 100 GB SSD

The training process took approximately 40 minutes to complete for 19000 input images. The random split of 0.2 is used for validation and the validation accuracy is calculated as metrics for evaluating the performance of the model.

Results

Model Evaluation & Validation

The model architecture explained above has provided the validation accuracy of close to 79% with 19k images as input. The benchmark model was able to obtain only 57% maximum whereas the CNN architecture was able to reach 80% accuracy considering the quality of images that were totally random. The challenges that were described in the implementation section are the realworld challeges that CNN has to comply with if it needs to be successfully used across various realtime applications.

This project has exemplified the power of CNN and the scope for its applications in real-time. Few interesting aspects observed during the experiment are listed below:

- The resource consumption is negligible considering the amount images that were fed.
- The architecture is scalable to increase in the volume of pixes and the number of images. When I fed the network 10K images, the network took approximately 30 minutes and with doubling the number of images to 19k, the model took 40 minutes maximum.
- Transfer Learning of VGG16 was not able to match the model accuracy although the last few layers were removed and customized given the images that are provided to the input.
- The model can definitely be improved by have more nodes at the initial and have more convolutional layers to capture much of the information at the initial stage and propogating through later stages.
- Ofcourse, more data could have made an impact on the accuracy.

The code samples and results are captured in the associated ipynb file.

Transfer Learning

Transfer learning is a powerful concept that makes models to leverage the weights of another model that was trained with larger dataset. Given the amount of resources required to train the convolutional model on very large datasets and also considering the fact that the initial set of convolutional layers aim to detect basic components of the image, it is highly possible to transfer the knowledge obtained from another pre-trained network and customize the last few layers to adopt to the classification problem at hand.

There are very good models that are being used in ImageNet competition and keras applications package has most of the popular models such as VGG16, VGG19, ResNet50, InceptionV3, DenseNet and so on. These models are trained for imagenet classification problem on a large dataset and has proven accuracy. In this project, VGG16 model is used to compare the performance of transfer learning weights for this image classification problem.

VGG16 is chosen primarily because it accepts images from size 48 *48. As explained earlier, these are real world images that were taken from videos and hence the quality and dimensions of images may not be perfect. Resizing the images to pixels higher than 64* 64 makes it blur and degrades the quality of prediction.

Conclusion

Reflection

Below is the summary of the process applied for this project:

1. Collect the dataset from the Hackathon competition at https://datahack.analyticsvidhya.com/contest/practice-problem-age-detection/ (https://datahack.analyticsvidhya.com/contest/practice-problem-age-detection/) https://www.analyticsvidhya.com/blog/2017/06/hands-on-with-deep-learning-solution-for-age-detection-practice-problem/ (https://www.analyticsvidhya.com/blog/2017/06/hands-on-with-deep-learning-solution-for-age-detection-practice-problem/)
2. Resize the images to 64 *64. Tried various dimensions and the accuracy dropped when increased to 128* 128.
3. Normalize the images by dividing by 255 to make everything in the same scale.
4. Stack convolution layers followed by Max pooling and BatchNormalization layer. This type of stacking has proven to perform best.
5. ReLU activation function is used as it provides better efficiency compared to Sigmoid and tanh activation.
6. Hired a GPU in floydhub with the below configuration to train the model.

   Standard GPU Tesla K80 · 12 GB Memory 61 GB RAM · 100 GB SSD
7. Compile and Fit train the model in 19k images for 30 epochs for batch size of 500.

The preprocessing step was challenging. I tried to resize the images to volume higher than 64 * 64 but the performance did not improve. I tried grayscale but color pictures performed better than grayscale images. Finding the right combination of network layers was very challenging as i have to balance between maximizing the resource consumption and prevention of overfitting.

The dataset is particularly interesting to showcase the power of CNNs to the real-world images that are taken in not so well equipped environment. The accuracy that the model was able to achieve is good considering the resource utilization and the limited number of images to work on. With some improvement and ofcourse with more data, the model can certainly be improved.

Improvement

Future improvements of this model and thereby related applications can include:

- Train with more data and utilize the model for different classification problems and see how it performs.
- Include more computational power to provide more filters/nodes at the convolutional layer.
- Potential improvement can be reducing the number of parameters required without affecting the accuracy.

References

1. S. Setty et al., "Indian Movie Face Database: A benchmark for face recognition under wide variations," 2013 Fourth National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG), Jodhpur, 2013, pp. 1-5.
2. Felipe Moure C´ıcero, Ary Henrique M Oliveira, Glenda Michele Botelho (2016) Deep Learning and Convolutional Neural Networks in the Aid of the Classification of Melanoma
3. http://cs231n.github.io/transfer-learning/ (http://cs231n.github.io/transfer-learning/)
4. https://keras.io/applications/#vgg16 (https://keras.io/applications/#vgg16)