2019SP_MSDS_458-DL_SEC56 Week10- Assignment4

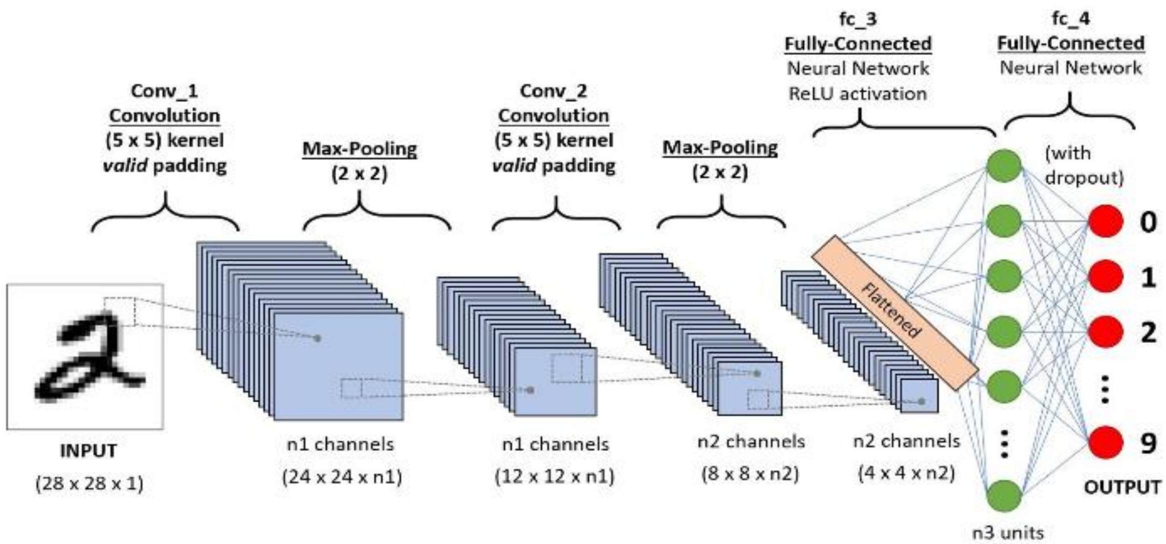
Prabhat Thakur

Deep Learning - Convolutional Neural Network (CNN) for Gender Classification

Abstract: The goal of this assignment is to research an application of AI in the area of computer vision. I worked on gender classification problem using deep convolutional neural network (CNNs). Gender classification has become relevant to an increasing amount of application, particularly since the rise of social platforms and social media. In this paper I used deep-convolutional neural network. A significant increase in prediction performance can be obtained using CNN models.

Introduction: In this assignment, I attempted to utilized face recognition capabilities from cvlib library to first extract the faces from the image and then pass it to gender detection model. Face recognition techniques described in the last few years have shown that tremendous progress can be made by the use of deep convolutional neural networks (CNN) .The data used in the assignment is gathered from Google Images. The dataset consists of around 2200 face images (~1100 for each class). The size of the training data set is small which could result in over fitted model.

Literature: A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.



A CNN sequence to classify handwritten digits

Source: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

A ConvNet is able to successfully capture the Spatial and Temporal dependencies in an image through the application of relevant filters. The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and reusability of weights. In other words, the network can be trained to understand the sophistication of the image better. The role of the ConvNet is to reduce the images into a form which is easier to process, without losing features which are critical for getting a good prediction.

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data through dimensionality reduction. Furthermore, it is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training of the model. There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel. In this research max pooling is used.

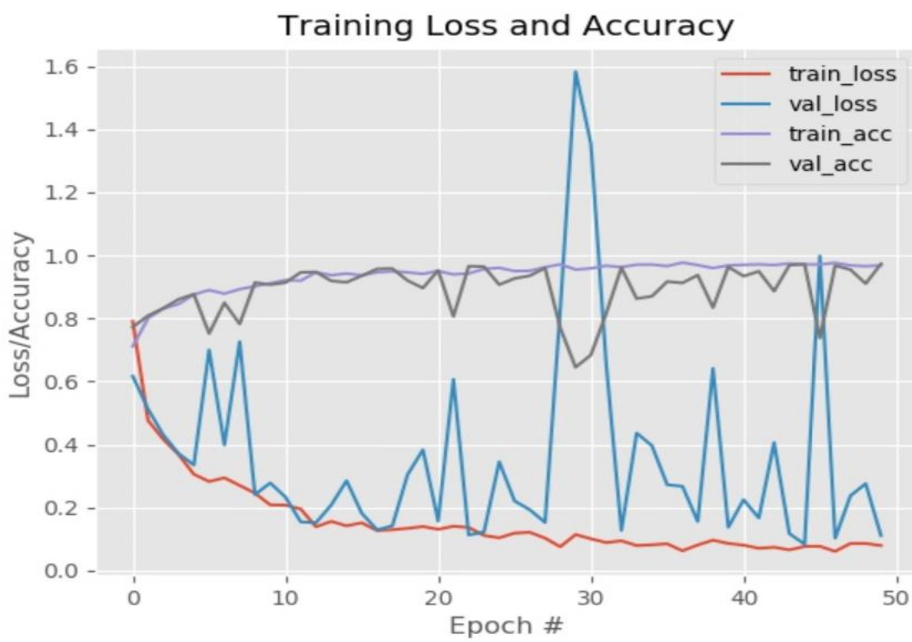
The high-level features as represented by the output of the convolutional layer is then passed to Fully-Connected layer for learning non-linear combinations. The Fully-Connected layer is learning a possibly non-linear function in that space. Now that we have converted our input image into a suitable form for our Multi-Level Perceptron, they are than flatten into a column vector. The flattened output is fed to a feed-forward neural network and backpropagation applied to every iteration of training. Over a series of epochs, the model is able to distinguish between dominating and certain low-level features in images and classify them using the binary crossentropy Classification technique.

There have been several papers over the years studying the topic of gender classification. A detailed survey of gender classification methods can be found in [3] and more recently in [7]. One of the early methods for gender classification [5] used a neural network trained on a small set of near-frontal face images. In [6] the combined 3D structure of the head (obtained using a laser scanner) and image intensities were used for classifying gender. SVM classifiers were used by [4], applied directly to image intensities.

Methods: One of the first applications of convolutional neural networks (CNN) is perhaps the LeNet-5 network for optical character recognition. Compared to modern deep CNN, their network was relatively modest due to the limited computational resources of the time and the algorithmic challenges of training bigger networks.

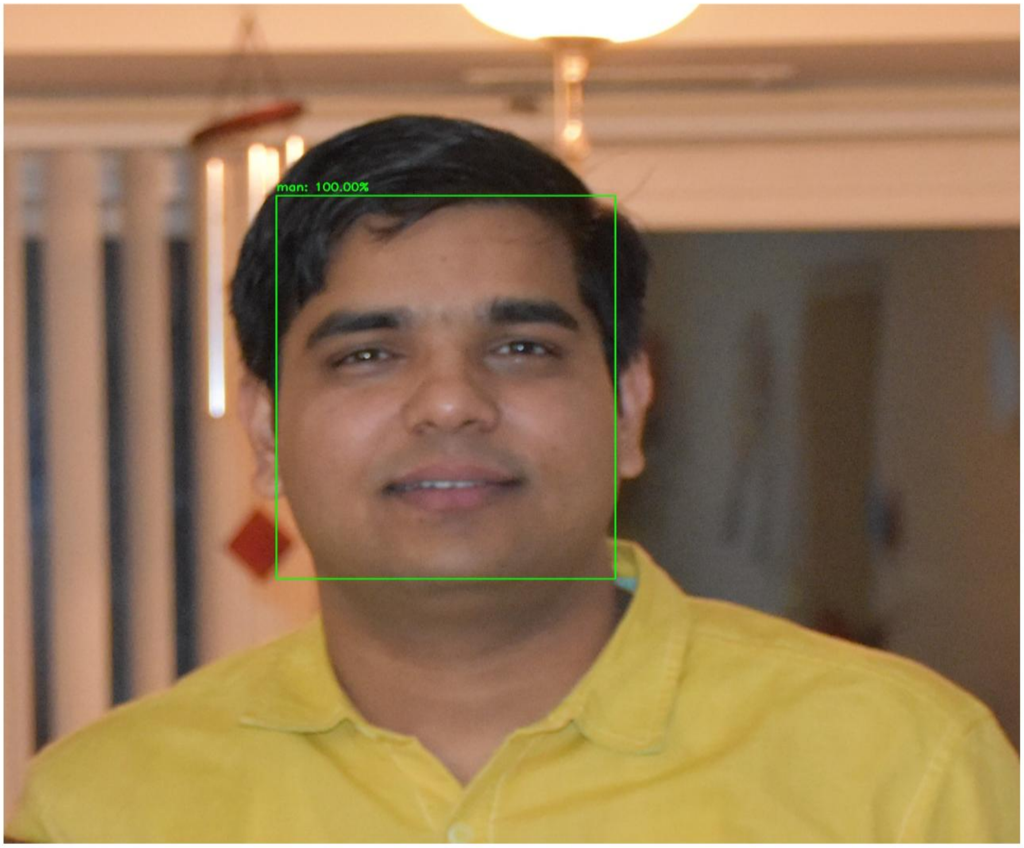
In this assignment, the keras model is created by training SmallerVGGNet from scratch on around 2200 face images (~1100 for each class). Face region is cropped by applying face detection using cvlib on the images gathered from Google Images. All three color channels are processed directly by the network. Images are first rescaled to 96 × 96 and fed to the network. Image augmentation is used for generalized model output. Refer Index figure 1 for complete network structure.

Results: The model is trained using 50 epochs with 64 images per batch. Learning rate is set to 0.001. The model achieved around 96% training accuracy and ~94% validation accuracy. (20% of the dataset is used for validation). Below is the model training loss and accuracy plot for both train and validate data set.



Each epoch took about 300 seconds to train. Due to the resource constrain model was trained using 50 epochs only, from the plot it seems that model should have been trained from more epochs.

Below is the model prediction on my image. I also tried some images with multiple faces mix of man and woman. The model did not perform well on those images.



The presence of the dropout layers, used to prevent overfitting, do in fact improve accuracy with the addition of some computational time.

Conclusion: Though many previous methods have addressed the problems of gender classification, until recently, much of this work has focused on constrained images taken in lab settings. Such settings do not adequately reflect appearance variations common to the real-world images in social websites and online repositories. Internet images, however, are not simply more challenging: they are also abundant. The easy availability of huge image collections provides modern machine learning based systems with effectively endless training data, though this data is not always suitably labeled for supervised learning.

Overfitting is common problem when machine learning based methods are used on such small image collections. This problem is exacerbated when considering deep convolutional neural networks due to their huge numbers of model parameters. Care must therefore be taken in order to avoid overfitting under such circumstances.

References

[1] Chollet, F. (2018). Deep learning with Python. Shelter Island, NY: Manning Publications.

[2] A Comprehensive Guide to Convolutional Neural Networks, Retrieved May 02, 2018, from https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[3] E. Makinen and R. Raisamo. Evaluation of gender classification methods with automatically detected and aligned faces. Trans. Pattern Anal. Mach. Intell., 30(3):541–547, 2008.

[4] B. Moghaddam and M.-H. Yang. Learning gender with support faces. Trans. Pattern Anal. Mach. Intell., 24(5):707– 711, 2002.

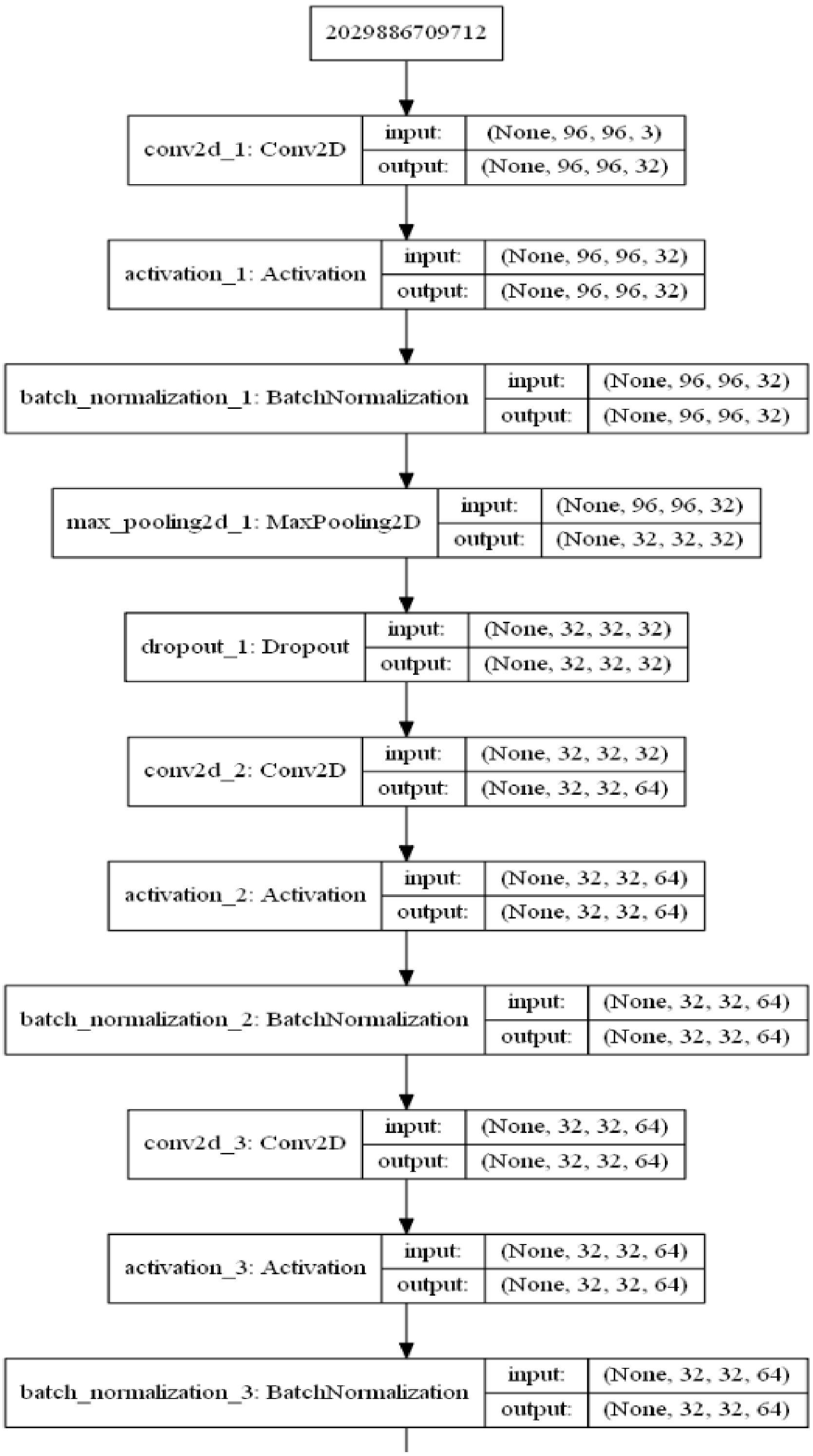
[5] B. A. Golomb, D. T. Lawrence, and T. J. Sejnowski. Sexnet: A neural network identifies sex from human faces. In Neural Inform. Process. Syst., pages 572–579, 1990.

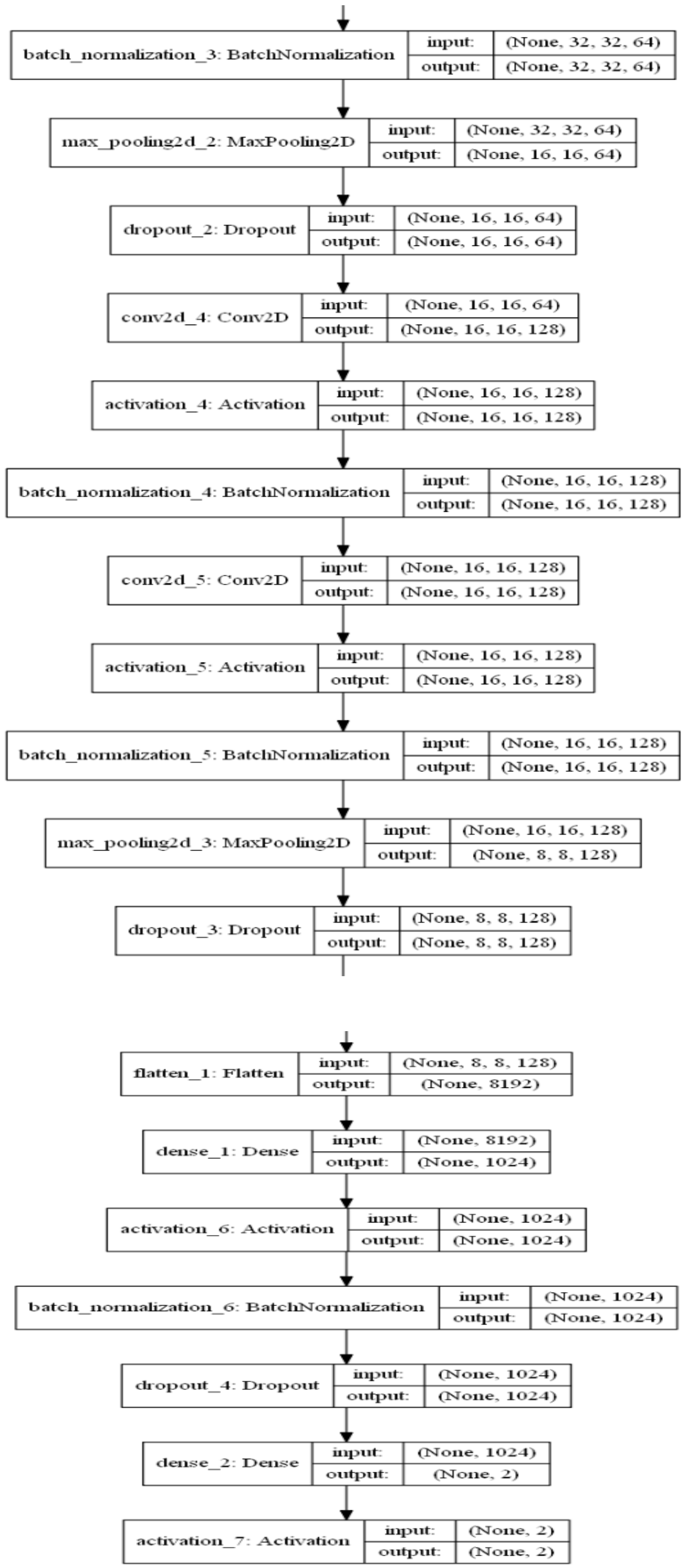
[6] A. J. O'toole, T. Vetter, N. F. Troje, H. H. Bulthoff, et al. Sex ¨ classification is better with three-dimensional head structure than with image intensity information. Perception, 26:75–84, 1997.

[7] D. Reid, S. Samangooei, C. Chen, M. Nixon, and A. Ross. Soft biometrics for surveillance: an overview. Machine learning: theory and applications. Elsevier, pages 327–352, 2013. 2

Appendix:

Figure 1 - Model Structure





batch_normalization_3: BatchNormalization
input: (None, 32, 32, 64)
output: (None, 32, 32, 64)
max_pooling2d_2: MaxPooling2D
input: (None, 32, 32, 64)
output: (None, 16, 16, 64)
dropout_2: Dropout
input: (None, 16, 16, 64)
output: (None, 16, 16, 64)
conv2d_4: Conv2D
input: (None, 16, 16, 64)
output: (None, 16, 16, 128)
activation_4: Activation
input: (None, 16, 16, 128)
output: (None, 16, 16, 128)
batch_normalization_4: BatchNormalization
input: (None, 16, 16, 128)
output: (None, 16, 16, 128)
conv2d_5: Conv2D
input: (None, 16, 16, 128)
output: (None, 16, 16, 128)
activation_5: Activation
input: (None, 16, 16, 128)
output: (None, 16, 16, 128)
batch_normalization_5: BatchNormalization
input: (None, 16, 16, 128)
output: (None, 16, 16, 128)
max_pooling2d_3: MaxPooling2D
input: (None, 16, 16, 128)
output: (None, 8, 8, 128)
dropout_3: Dropout
input: (None, 8, 8, 128)
output: (None, 8, 8, 128)
flatten_1: Flatten
input: (None, 8, 8, 128)
output: (None, 8192)
dense_1: Dense
input: (None, 8192)
output: (None, 1024)
activation_6: Activation
input: (None, 1024)
output: (None, 1024)
batch_normalization_6: BatchNormalization
input: (None, 1024)
output: (None, 1024)
dropout_4: Dropout
input: (None, 1024)
output: (None, 1024)
dense_2: Dense
input: (None, 1024)
output: (None, 2)
activation_7: Activation
input: (None, 2)
output: (None, 2)