Convolutional Neural Networks for texture classification

Edgar A. Margffoy-Tuay
Universidad de los Andes
201412566
ea.margffoy10@uniandes.edu.co

Abstract

We present a Convolutional Neural Network architecture suitable to classify texture images, our approach achieves a 90% accuracy when evaluated over the validation set of Ponce's group texture dataset, compared with other classical approaches, such as Texton Dictionary representation, which achieves 60% accuracy.

1. Introduction

Convolutional Neural Networks (CNN) were proposed by Yann LeCun [1] in 1988 to recognize digits. This class of specialized networks are based on a set of convolutional layers to compute a set of filters activations over an image input, which in turn are then aggregated via a set of Pooling layers. This architecture and layout allows to learn new features that cannot be described only by using a set of linear filters, such as the Texton Dictionary framework proposed by Malik et al on [4], on which, each image is represented by a Texton histogram dictionary computed from a set of activation responses between the image and a filter bank that contains different linear and blob detectors.

In this framework, both approaches seem familiar, however, the main difference radicates in the filter initialization process, while a CNN starts with random filters which then are learned via backpropagation, the texton dictionary depends on a set of predefined filter bank. Also, the texton approach only considers first order filters, such as blob and edge detectors, while a CNN not only learns this first order filters, but it also learns more complex filters in latter layers, which improve the higher dimension representation of each image of the dataset. Finally, and more important is related to the classification representation vectors, while on the texton approach the representation relies on calculating K-Means over all the filter activations of an image, which then are given as an input to another classifier such as SVMs and Random Forests, a CNN allows to represent and classify at the same time, by using fully connected layers in the same fashion as a conventional Feed-Forward Neural Network, this allows to stack and propose several reference deep classification architectures, such as AlexNet [3], VGG [5], ResNet [2] and GoogLeNet [6], which achieved state-of-the-art performance on several classification challenges and benchmarks, such as ImageNet.

2. Materials and Methods

The texture classification dataset consists of 1000 texture images of size 640×480 grouped on 25 distinct categories, according to different light illumination conditions. Due to the reduced number of images present on the dataset, it is necessary to sample multiple texture patches of size 32×32, this allows to increase the input image representation space, which should be suitable to train a CNN.

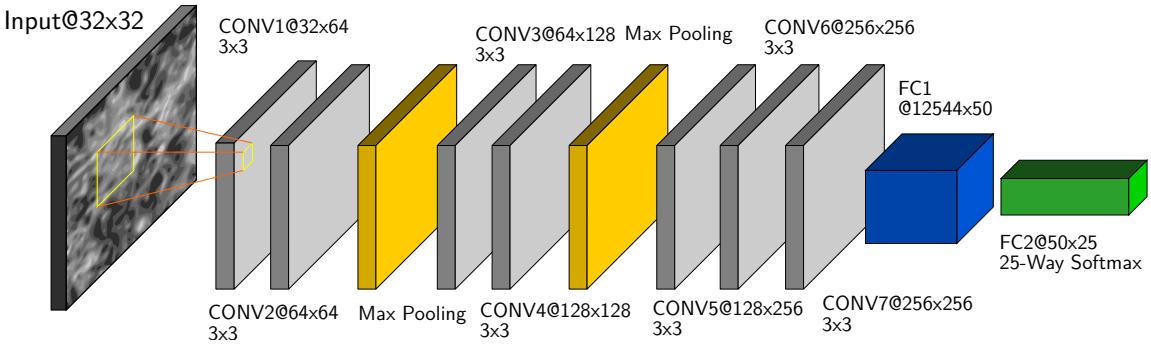

Figure 1: Graphical description of the proposed convolutional neural network architecture

The overall architecture consists of 7 convolutional stacked layers with kernel size 3×3 and stride of 1. The network is trained with a set of 2 fully connected layers that are mapped to a 25-way softmax layer. The non-linearity applied to all the convoltional and fully connected layers correspond to Rectified Linear Units (ReLu), which allow to discard all negative activations while not squashing the positive inputs to a single range. Max Pooling layers of size 2×2 were defined twice between each pair of down-4 layers. The overall network architecture is presented on Figure 1.

The network was trained using random initialization during 100 epochs, the selected optimization method correspond to Stochastic Gradient Descent with momentum, the selected learning rate was fixed at 0.01, and momentum rate of 0.5. To improve the convergence steps, the input set was normalized to have a mean of 0.485 and a variance of 0.229, this allows to unskew the error surface to a more convex form. Also, early stopping was used to prevent overfitting. Finally, the network implementation was done over PyTorch and trained using an NVIDIA Tesla K40 GPU.

3. Results

The network training architecture was modified several times to improve the weight adjustment without incurring in overfitting, initially, the network was proposed with a single Convolutional layer and a single connected layer. Under this model, the network was unable to achieve more than 24% accuracy, this is due to the limitation of a single convolutional layer, which only allows to take an image and extract basic features based only on linear and blob filters, this approach is no different from taking a texture image and classify it via Texton Dictionary representation. To improve the accuracy, a total of four convolutional layers were stacked upon the first defined convolutional layer, this allowed to increase the accuracy of the model, from 24%, up to 60%.

At this point it was necessary to test new training strategies, such as introducing pooling layers and also proabillistic Dropout, while dropout did not improve the accuracy of the model, adding pooling layers between each two stacked layers reduced the total number of convolutional actiavtions, and allowed to increase the total accuracy of model to 63%, however, the total number of parameters trained did not explain and fit the texture data well. To further improve the model training, more convolutional layers were stacked, those layers help to correlate and explain better the lower convolutional features, while the lower convolutional layers help to learn basic features and filters, the upper layers allow to specialize the feature activations that adjust to single input cases.

After adding a set of three more convolutional layers, without any pooling layers, the accuracy increased to 90% over the validation

set and 98% over the training set. As it can be shown on Table 1, the upper layers did improve the classification tasks, while more shallow networks do not perform well.

Network	Acc
CONV1, FC1	24%
CONV1-4, FC1, FC2	60%
CONV1-4 (POOL), FC1, FC2	63%
CONV1-5 (POOL), FC1, FC2	70%
CONV1-6 (POOL), FC1, FC2	80%
CONV1-7 (POOL), FC1, FC2	90%

Table 1: Ablation tests done over the proposed neural network architecture

4. Conclusions

Convolutional Neural Networks are suitable to large scale visual datasets, this model allows to represent an image by adding the responses of several filter layers that are learned via back-propagation, this approach allows to represent an input and classify it at the same time, without any distinction between feature representation and classifier, as it was the usual framework before 2012.

Along the experiments proposed previously it was possible to analyse and identify the function of each of the stacked layers proposed, in each iteration it was possible to observe that adding more convolutional layers on top can help to improve the training convergence, however, this procedure must be controlled to reduce overfitting.

With respect to the total number of layers present on the architecture, it is possible to conclude that the lower layers tend to be similar across all CNN domains, independent of the input dataset, this is due to the fact that these layers tend to learn simple features, such as linear filters, which are canonical for all CNN models, however, the difference between models are constrained by the top layer features, which are more expressive and specialized on the input dataset. This allows to finetune and refine the filter activation responses of a pretrained network over a larger dataset, such as ImageNet to adapt to new and unrelated datasets, with feasible and good results.

Finally, the number of Fully connected layers should be reduced should the model learns a sufficiently good set of features, which can be more expressive and simple than adding a more complex fully connected set of weights that increase both training space and time complexity.

References

[1] Y. L. Cun, B. Boser, J. S. Denker, R. E. Howard, W. Habbard, L. D. Jackel, and D. Henderson. Advances in neural information processing systems 2. chapter Handwritten Digit Recognition with a Back-propagation Network, pages 396–404. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[2] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, NIPS'12, pages 1097–1105, USA, 2012. Curran Associates Inc.

[4] J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and texture analysis for image segmentation. *International Journal of Computer Vision*, 43(1):7–27, 2001.

[5] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[6] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.