Lab 12: CNN

Adrian S. Volcinschi M.
Universidad de los Andes
https://uniandes.edu.co/en
as.volcinschi@uniandes.edu.co

Abstract

The image classification problem has been around and will be for a long time. In this paper the now a days classic approach was implemented: Convolutional Neural Networks. More specifically, a ResNet-50 was implemented to classify 10 labels from the CelebA dataset including male, female, colors of hair, and more. The toughest part about implementing this kind of solutions are the resources and the dataset/train algorithms. With the implemented ResNet-50 using 3 epochs an F-Measure of 0.561 was achieved.

1. Introduction

The problem of image classification is a problem that comes from way back. It has gone through many stages like Texture models [7], HOG models [1], and nowadays Neural Networks such as AlexNet [6] in the early 2012. This problem consists of assigning a label to an entire image based on the contents it has. For example, labeling a car image as a car can be considered a classification, but labeling a car image as a Ferrari is also as good. This is were coarse and fine grained classification definitions come in handy. A coarse grained classification can be understood as labeling an image based on the more high-semantic level information such as car. A fine grained classification can interpreted as the Ferrari example, where the Ferrari implies a car label but it is much more specific. In this specific task of classification we are classifying various attributes of a celebrity in a picture of them. This can therefore be understood as a fine-grained classification problem that is going to be taken upon using convolutional neural netowrks (CNN's) in the CelebA dataset. A classic approach to this problem would be to create a representation space for the features we want to extract and then use some classifier such as a Random Forest or a Support Vector Machine to discriminate between these features to obtain a label. Since the performance of Neural Networks is much higher than the ones of the classic approach, this paper is going to be centered around them.

Convolutional Neural Networks are defined as deep Neural Networks that have convolutional layers inside their architectures in what's called the hidden layers. Convolutional layers are layers where a preset kernel, or filter, of an odd size convolves over the input along the X and Y dimensions but also the depth or Z dimension and strides with a certain step to subsample if it is desired. This is what enabled image classification to boost up in performance over the years with different architectures such as VGG [9], GoogLeNet [10], ResNet [3], and DenseNet [4].

Going into more detail for the chosen architecture (ResNet-50) it is important to consider this is a special kind of architecture since it is not exactly optimizing the neurons themselves but the residuals. When an input X goes through a residual block, the output is going to be X + a function of X that is the residual block itself. With this you can easily optimize to get a residual function F = 0 rather than a complex F function and use and identity mapping [5].

2. Materials and Methods

To train the celebrity attributes classifier the CelebA [8] dataset was used. This dataset consists of 202,599 pictures of 10,177 different celebrities in different scenarios. Each picture has a corresponding one-hot vector with 40 binary labels in regards with 40 categories such as male, female, wavy hair, pointy nose, etc. For this paper, the whole dataset was used following the partition proposed by the author: 162,770 train, 19,867 validation, and 19,962 test. Additionally, only 10 categories were used to simplify the problem. The selected categories are: Eyeglasses, Bangs, Black Hair, Blond Hair, Brown Hair, Gray Hair, Male, Pale Skin, Smiling, and Young. Some of the train images are shown in Figure 1

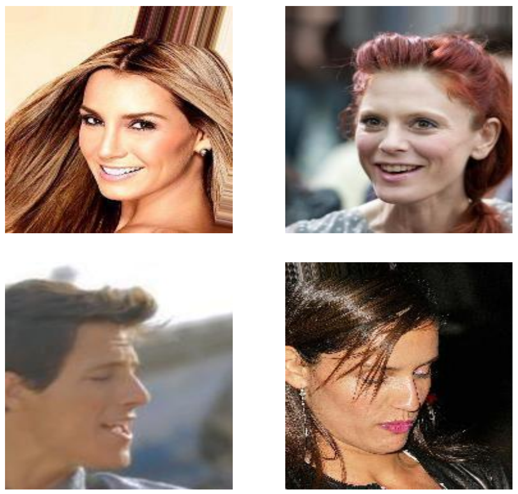

Figure 1: Pictures of the FER 2013 dataset

2.1. Custom Network

The first attempt to solve this problem consisted in a custom neural network based on convolutional layers, Stochastic Gradient Descent (SGD), ReLu's, max pooling, Softmax Cross-Entropy as a non-linear function, and finally a Cross-Entropy loss function. The first layer consisted of a convolutional layer which took the RGB image and convolved it with 64 3x3 filters. Then a second convolutional layer did the same with 128 3x3 filters. After the second layer, a max pooling and ReLu were added to break the continuous flow of convolutional layers between layers two and three. After these non-linearities a third convolutional layer with 32 3x3 filters was used again. Finally, the output of this third convolutional layer went through a max pooling and ReLu into a fully connected layer with 10 outputs corresponding to the 10 categories.

2.2. ResNet-50

For this paper a ResNet-50 architecture was implemented to solve this problem. I began by using the pretrained weights of the ResNet-50 on ImageNet since it has shown that it improves early convergence [2]. Due to time and available resources, all experiments were done in the complete dataset using a batch size of 100 and iterating through 3 epochs only. It was implemented with an SGD optimizer with default pytorch parameters. Finally, since the problem poses a 10 category classification, a Cross-Entropy Loss Function was used to measure the loss along the training process.

2.3. Evaluation Method

For the multiclass classification the F-Measure was selected as the evaluation metric. This was done using the test server to get an accuracy and F-measure for the .txt file that the code generated which had a 1 for positive and 0 for negative result for each category.

3. Results

3.1. Custom Network

When the custom network was implemented, the convergence was really bad. The algorithm took too much memory and didn't got close to a reasonable loss value which is around 1. Since it took so much resources for nothing, no further studies were done with this network.

3.2. ResNet-50

The trained ResNet-50 that was trained with the whole dataset and during 9 epochs gave a resulting F-measure of 0.522. The graph of loss in train and validation can be seen in Figure 2. When the same model was trained with only 1 epoch it gave an F-measure of 0.561.

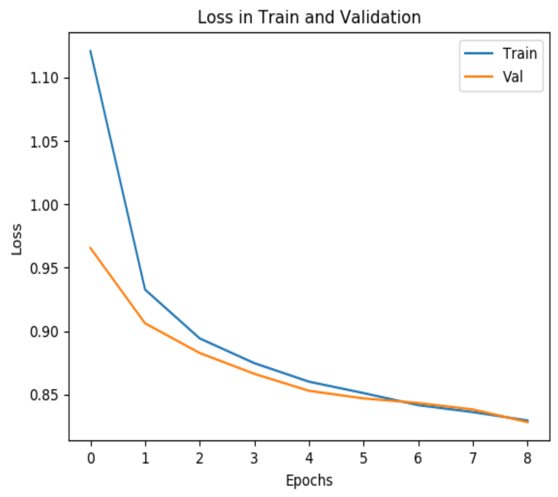

Figure 2: Loss in train and validation sets of the CelebA dataset with a ResNet-50

This led me to first think the model was doing overfitting but the validation losses kept going down as epochs occurred. Later on the optimizer was changed from an SGD optimizer to an Adam one. The model was trained for even more epochs (12) and the result went further down with a F-measure of 0.392. This is not making much sense so a more detailed depuration of the code was realized. Upon doing it, the only source of error left was the dataloader. When the dataloader was inspected a subindex for the 10 labels chosen was in the wrong order. Additionally, the transform.resize module with only one int as a parameter resized only the height and rescales the width but not to the same size. This could have also been a source of error for the model since the input was not being of the sime size always.

3.3. Jitter

In the computer vision area the jitter, or also called Data Augmentation, is the transforms that are performed over the images before using them to train, validate or test. In this case a horizontal flip with a probability of 0.5 was used alongside a normalization over all the RGB channels with values for the mean: [0.485, 0.456, 0.406] and for the variance: [0.229, 0.224, 0.225]. A short study was done upon the impact of these transforms using the SGD ResNet-50 with only 2 epochs. The results can be seen in Fugure 3.

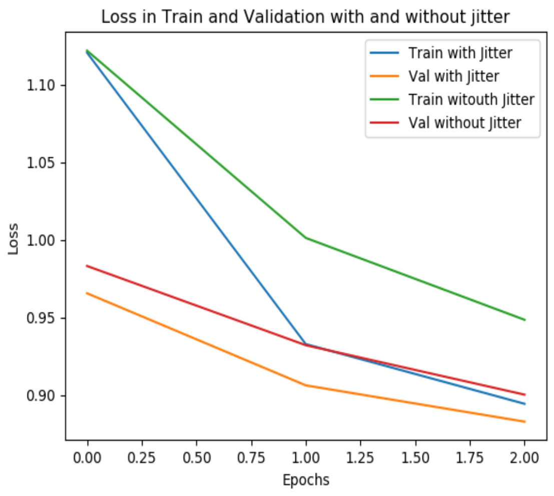

Figure 3: Loss in train and validation sets of the CelebA dataset with a ResNet-50 for Jitter study

As it can be seen, the jitter or data augmentation reduces the loss over all the epochs spectrum showing that implementing data augmentation is a good practice to get better results overall and even more when there are more epochs so the changes can be more.

4. Conclusions

As it was shown, a deeper network gives considerable better results than a network with few layers. Additionally, the data augmentation helps improve the loss a little bit which in return can be better overall performance. Besides this, further architecture tweaks can boost the performance even higher and trying other Neural Networks such as Variational Auto Encoders could help with this problem. Finally, the correct implementation of this CNN's is not an easy task ranging from the dataset loading which can be hard to do to the architecture itself and the correct formulation of the train and test algorithms.

References

[1] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *international Conference on computer vision & Pattern Recognition (CVPR'05)*, volume 1, pages 886–893. IEEE Computer Society, 2005.

[2] K. He, R. B. Girshick, and P. Dollár. Rethinking imagenet pre-training. *CoRR*, abs/1811.08883, 2018.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[4] G. Huang, Z. Liu, and K. Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016.

[5] P. Jay. Understanding and implementing architectures of resnet and resnext for state-of-the-art image classification: From microsoft to facebook [part 1], feb 2018.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[7] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43:29–44, 2001.

[8] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[9] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[10] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.