A CNN Model for Classification of Paintings According to their Artists.

Abhisek Panda
(150026)
apanda@iitk.ac.in

Raktim Mitra
(150562)
raktim@iitk.ac.in

MLG-8

Abstract

Many of the machine learning problems like object detection, face recognition, text recognition on images, complex semantic filtering, etc. are quite simple for common humans and yet very hard to learn by a computer model. However, in this project, we explore a more complex problem which is challenging even for humans, i.e. recognizing the painter of a given painting. If a person has not previously seen the concerned painting, it is generally very hard to answer. The task of identifying a given art's true painter is generally carried out by an expert in this domain, which in general, is a slow and expensive process, and also error-prone. In this project, we investigate if the celebrated Convolutional Neural Networks are capable to tackle this problem. Given the sheer complexity of the problem, we present the problem as a multi-class classification problem where the network has to classify a given painting among 300 different painters and build a model to solve the learning problem.

Our experiments suggest that CNNs are highly suitable for identifying the artists of various paintings which is accomplished by learning the minute details about their art styles.

1 Introduction

Each day, a huge amount of data is uploaded on the internet for the world to view, amounting to nearly 50000GB/second(2). Artistic paintings form a big part of this set(3). As these databases become bigger, there is an urgent need to automate the process of tagging these images with proper labels like artists, since doing it manually is very slow, expensive and requires considerable expertise.

This project is an attempt to extract the unique signature of an artist reflected through his/her drawing style by using Convolutional Neural Networks. No previous work, to the best of our knowledge, has tried using CNNs for classifying paintings based on their artists for such a large scale dataset.

2 Problem Statement

Given a training and a test dataset containing paintings of 300 different painters, guess the painters of the test images among the 300 possible painters.

Evaluation Metrics: Exact prediction (top 1 guess), top 5 guess, top 10 guess, top 20 guess.

Casting as a classification Problem

We cast the problem of guessing painter of a given painting as a classification problem:

- **Input**: A $3 \times 256 \times 256$ RGB image (A painting).
- **Output**: A vector $v \ \epsilon \ \mathbb{R}^{300}$ where each v_i corresponds to a painter and $\forall i, j \epsilon [300] \ \ v_i > v_j$ means painter i is more likely to have painted the picture than painter j.

3 Previous Work

We discuss some published works on problems similar to the problem we focus.

- Eva Cetinic, Sonja Grgic, in their 2013 work(5) attempted to tackle the problem by image feature extraction and building a classifier based on that. They trained several models including one multilayer perceptron with one hidden layer. Their work is illuminating but very limited in the dataset they used, it contained only 500 images from 20 painters.
- A 2016 paper(4) by Sang-Geol Lee and Eui-Young Cha addresses the problem of painting classfication on basis of genre using convolutional neural network. Basically they made classes of painters containing painters of similar styles (impressionism, post-impressionism, surrealism and expressionism) and their model classified paintings to one of these classes. Although this answers a good way of classifying paintings of similar style, it does not help to detect individual painters as individual painters will have more minute differences in their artwork than classes of painters of different styles.

In general, most works done on this problem other than those mentioned above are mainly based on feature extraction(6) (or using generic image features like SIFT AND HOG) and applying techniques like SVM, Hierarchical clustering. In most cases the main drawback is the limited dataset, less number of painters, less number of training images and/or only similar kind of paintings being considered. Thus, the core problem remains to be able to extract useful features from the very high dimensional images.

However, Convolutional Neural Networks(CNNs) have proved to be very effective in learning very complicated features and hence, we experiment with this model to address the problem at hand.

4 Data Acquisition and Pre-Processing

The dataset used in this project was originally hosted at the Kaggle competition: "Painter By Numbers"(9). The original data-set consisted of 1584 different artists in the training set with 79433 training images. The test set consisted of images from different painters, some of which were present in the training set too, and some others were not. Each image was in Joint Photographic Expert Group(JPEG/JPG) format in both training and test dataset. There was no fixed resolution and all images were of varying sizes. 1 csv file was also provided for the training set, with 1 entry per training image. It contained information like the painter's ID(unique for each painter), painting title, school of painting, genre and year of painting.

However, the resources available at our hands did not permit us to use such a large scale dataset. So, we had to trim the training dataset to comprise of the top 300 artists in terms of number of paintings available. Of course, this step was also repeated for the test set so that test set consisted of images from these 300 artists only.

The data statistics of the new training set are as follows:

- Total number of images: 48351
- Mean Number of Paintings per artist: 161.17
- Standard Deviation of Number of Paintings: 95.51
- Number of Paintings by anyartist: 413
- Number of Paintings by any artist: 72

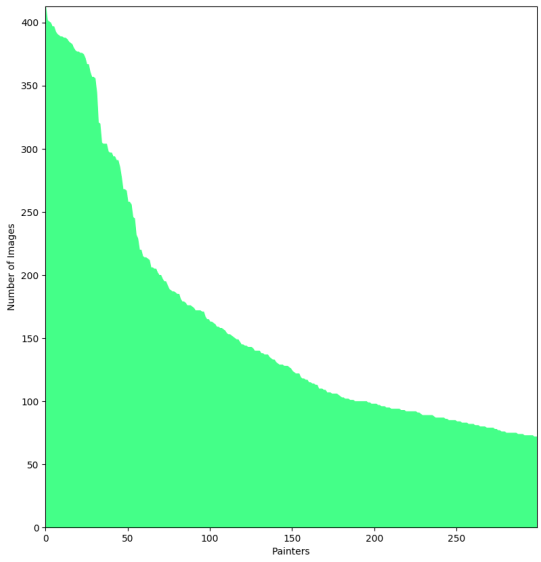

Figure 1: The frequency of images per class in the new training set comprising of top 300 painters in terms of number of their paintings

Challenge of Evaluation

The evaluation was originally to be carried out by Kaggle competition organizers, hence no meta data about the test images was provided. Fortunately, after the competition was over, a csv file was provided, which had detailed information about the original painter of each image, both in training and test set. However, this posed another problem since the training images were annotated with artist IDs whereas the test images were annotated with artist names, and there was no file stating which ID belonged to which name. To overcome this problem, we designed a python script that created a new table storing the name to ID conversion by querying against both csv files with *filename* as common field. This method only ensured that test set is restricted to only those painters who are available in training set, and didn't tamper with the images in any way whatsoever, so the test set was unseen by us during training.

Data Processing and Augmentation

Each image, both in training set and test set, was first resized to $276px \times 276px$ size by matching the smaller dimension among height and width to $276px$ and then taking a center crop w.r.t the other dimension. This served as the base image for augmentations.

Data Augmentation is a powerful strategy for improving the accuracy of any Machine Learning model and it helps in reducing overfitting. Since some artists had very few images in training set, we augmented the dataset by intoducing 7 images per image which included the original image, the image rotated by $90°$, $180°$ & $270°$, mirror image, water image and the transposed image. Each of the 7 images was of size $256px \times 256px$, which was selected by taking a random crop of the earlier $276px \times 276px$ base image.

To facilitate the coding part, all images in both training set and test set were organized in a hierarchical order, with all images of an artist grouped in a directory, the directory name being the ID of the artist(this file struture was separate for training and test set).

Apart from the above mentioned augmentations, each training image was distorted by zooming and applying a sheer before feeding into the model. This was done to reduce overfitting. However the zoom and sheer levels were kept very low since they tend to change the artistic style of the paintings.

Training and Validation split

The validation set was created by selected 1/6 fraction of images randomly from all the augmented images. After this division, the training set contained 281997 images and the validation set contained 56460 images. The test set comprising of these 300 artists consisted of 13712 images.

5 Model Architecture

We decided to use Convolutional Neural Networks(CNNs) for the problem at hand. This decision was based on the huge success of CNNs in image classification tasks. However, this problem was different from the problems that the CNNs have solved in the past in the following ways:

- It is very difficult to predict the true artist of a painting even for humans unless we have a clear understanding of the artistic styles of the concerned artist. However, this may often lead to incorrect results since an artist can paint multiple images that have absolutely no visible correlation amongst them.
- Each artist has some kind of uniqueness to him/her which is reflected in his works. For some artists, it may be obvious like using bright colors or using a particular shading frequently, however, in most cases, this unique identifying feature is quite subtle. Most humans cannot perceive the pixel level minute details of the image and this is where factors like brush strokes, line styles, etc come into picture(6).

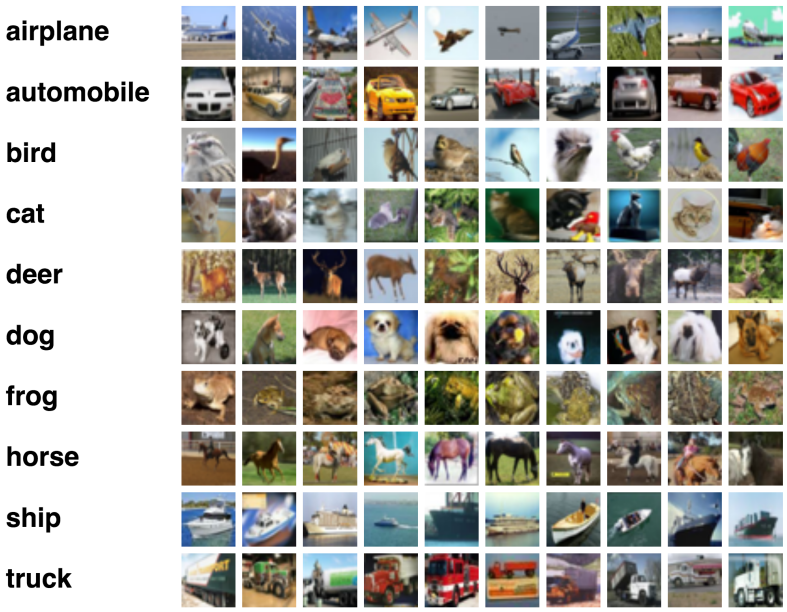

Figure 2: A sample from CIFAR10 dataset(10). Observe that it is very easy for humans to identify images belonging to the same class for a majority of these images.

To overcome the above problems, the following steps were taken during model designing:

- The kernel filter size was kept very small. Only 2×2 and 3×3 kernels were used for every layer. Since the kernels recognise hidden features in the images, restricting them to small sizes enabled the network to capture fine and more delicate details in the paintings.
- Only 1 max pooling layer was used at the beginning of the network to downsize the image from 256×256 to 128×128. Max pooling layers were avoided in later part of the network because they lead to loss of valuable information about the image by discarding 75% of data.
- The channel depth of each layer steadily increases as we from input towards output. As the convolutional network gets deeper, it captures more and more intricate details, hence

Figure 3: 2 images from the test set. These images depict trees and nature in general, however with highly different styles. Yet, these are painted by the same artist, Martiros Saryan. Most of the images in the dataset exhibit this unusual behaviour of appearing very different to human eyes, although they are created by the same person.

by increasing the channel depth towards output, we aim to capture more information about these details in the images.

- Batch Normalisation (7) layers were used after each convolutional layer. Batch Normalisation has provided faster convergence in most networks, hence we decided to incorporate it into our network too.

We used the categorical cross entropy function as the loss function to propagate gradients back. Mathematically, it is given as:

$$L(y, \hat{y}) = \frac{1}{N} \sum_{i}^{N} y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$$

Parametric RELU(PRelu) function was used as activation function for each layer except the final output layer, for which we used the softmax activation. PRelu is mathematically defined as :

$$PRelu(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{otherwise} \end{cases}$$

where α is a parameter that is learnt by the network.

Dropout was used in the fully connected layers to reduce overfitting. L2 regularization was also introduced for aggressive regularization. Adams optimizer was used to regulate the convergence. The learning rate was initialized as 0.00008.

6 Setup And Experiments

The entire model was implemented using Keras (8) library in python, using Tensorflow backend. The code was heavily inspired from (11), the model which won the original Kaggle competition.

The training was carried out on GPU servers provided by Department of Computer Science & Education, IIT Kanpur. The GPU used was NVIDIA 1080GTX with 8GB VRAM. A batch size of 16 was used for training and each epoch took nearly 10000s to complete. The training was done for 75 epochs, which took a little over 9 days for completion.

7 Evaluation on Test Data

The accuracies obtained by our model on the test set are as follows:

Category	Accuracy
Exact Prediction	0.5543
Top 5 prediction	0.7553
Top 10 prediction	0.8225
Top 15 prediction	0.8560
Top 20 prediction	0.8790

Analysis

The performance of our model is indeed very satisfactory compared to previous works. For instance, (14) obtained an exact prediction accuracy of 42.2% on the same dataset by training a CNN from scratch, however comprising of only top 57 painters. Hence our model is certainly a major improvement over it.

The error in predictions can be attributed to close similarities in the context of paintings, i.e., the picture they portray to us. For example, there are a few painters who have exclusively painted portrait images of people. Any portrait image from any other painter is highly likely to be classified as being painted by these few painters , which is technically incorrect, although our model has no way of knowing it. In a way, it has also learned to classify the paintings based on the scenes they depict.

8 Future Work

In future, we would like to test if we can extend the model such that, "Given any two paintings, the model will predict whether they are works of the same painter or not". This is the original problem statement of the *Painter by Numbers* problem, but that would require more computational resources as we will have to work on 1584 classes and almost 1.6 times the data we used in this project. We can also try to train the model to detect paintings from painters it has never seen before.

As an alternative to this discriminative approach, it would be very interesting to train a generative model which can generate fake paintings of an artist, by learning his/her art style.

We also expect the accuracy to go up by introducing adversarial attacks in the paintings. For example, juggling around a random number of rows and columns may not pe perceivable to humnas, but it greatly influences the model prediction. This way, the model is forced to learn intrinsic artistic details of the painting, instead of simply recognising the depicted scenes.

9 Conclusion

This project was an attempt to utilise the marvellous power of Deep convolutional neural networks for overcoming an extremely tough task of recognising an artist through his/her paintings. Overall we are impressed with the top-k performance of our model right now and would like to optimise it further and find out more ways to improve classification accuracy. We conclude with a note of positivity and hope for future improvement of our solution to tackle the problem in question.

References

[1] Entire source code and model architecture: `https://github.com/Abhipanda4/CNNpainter`

[2] Internet data usage statistics: A Blog on Corporate News
`https://blog.microfocus.com/how-much-data-is-created-on-the\`
`-internet-each-day/`

[3] DeviantArt: An online artwork, videography and photography community. https://www.deviantart.com/

[4] Lee, Sang-Geol & Cha, Eui-Young. (2016). *Style classification and visualization of art painting's genre using self-organizing maps*. Human-centric Computing and Information Sciences. 6. . 10.1186/s13673-016-0063-4.

[5] Cetinic, E & Grgic, Sonja. (2013). *Automated painter recognition based on image feature extraction*. Proceedings Elmar - International Symposium Electronics in Marine. 19-22.

[6] Sablatnig, Robert & Kammerer, Paul & Zolda, Ernestine. (2002). *Hierarchical Classification of Paintings Using Face- and Brush Stroke Models.* Proc. 14th Int. Conference on Pattern Recognition. 1. . 10.1109/ICPR.1998.711107.

[7] Sergey Ioffe and Christian Szegedy. (2015). *Batch normalization: accelerating deep network training by reducing internal covariate shift*. In Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37 (ICML'15), Francis Bach and David Blei (Eds.), Vol. 37. JMLR.org 448-456.

[8] Chollet, François and others, Keras(2015), GitHub Repository, https://github.com/fchollet/keras

[9] Kaggle Challenge: Painter By Numbers https://www.kaggle.com/c/painter-by-numbers

[10] *Learning Multiple Layers of Features from Tiny Images*, Alex Krizhevsky, 2009.

[11] inejc/painters: Winning solution for the Painter by Numbers competition on Kaggle, Github Repository, https://github.com/inejc/painters

[12] Stanford University CS231b project: Understanding Visual Art with CNNs

[13] Stanford University CS231b project: Using Convolutional Neural Networks to demystify aesthetic works of art

[14] Stanford University CS231n project: Artist Identification with Convolutional Neural Networks