MAIS 202 Deliverable 2: Who's the Artist?

Agnes Totschnig

1 Problem Statement

The aim of this project is to train a machine learning model to predict the artist of a painting.

2 Data Preprocessing

As proposed in Deliverable 1, we will be using the dataset of paintings of 50 influential artists available on kaggle, cf. reference [1]. I decided to start with only the top four artists for which there is more than 330 paintings available, notably Van Gogh, Degas, Picasso and Renoir. For the next deliverable I would like to collect more paintings to include the top ten artists and have a better class balance. The dataset was split in 80% training and 20% test.
Finally the images were resized to 100 × 100 pixel and the colors were rescaled to 1. To perform some data augmentation, I used horizontal and vertical reflections and slight shear of the images. I was also thinking of taking random small samples of the paintings, as the style of the artist should be recognizable everywhere on each painting.

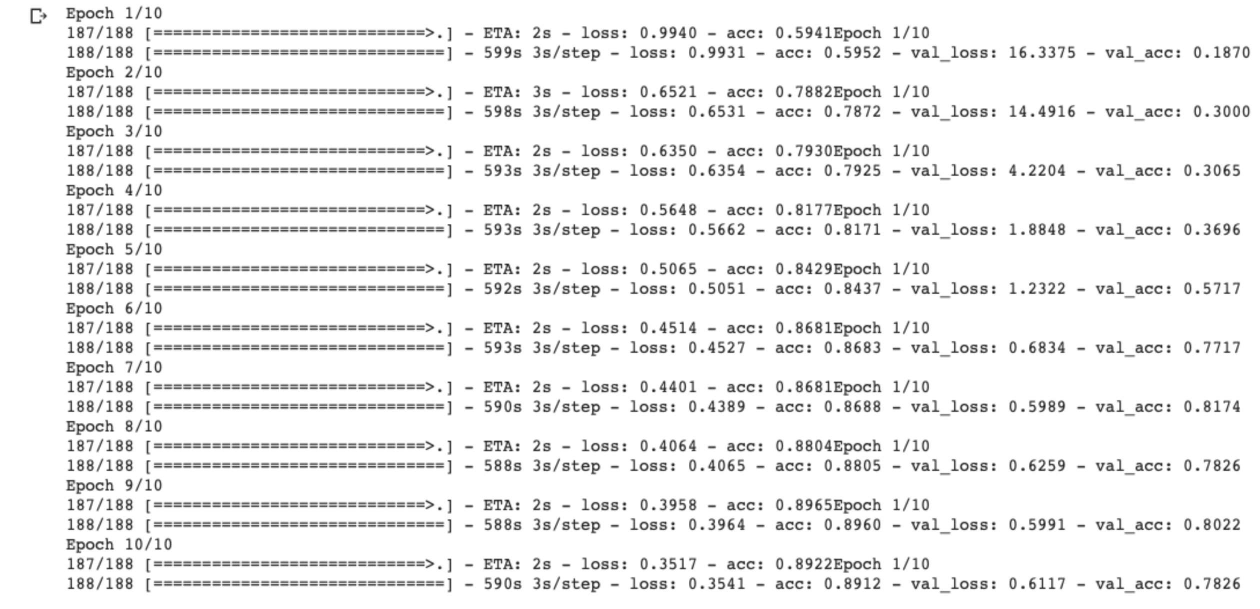
3 Machine Learning Model

To get started, I tried the approach proposed in [5] using a convolutional neural network. It is based on a `ResNet50` architecture with imported weights, pretrained on ImageNet. I used the Keras `ImageDataGenerator` for preprocessing and the Keras models for training.

As usual, between the layers of the neural network, batch normalization layers are added to keep the mean activation close to 0 and the standard deviation close to 1. The activation function used is the ReLU (Rectified Linear Unit) as it is a popular choice in image processing. The added layers at the end of the model are initialized following a normal distribution. Moreover, `ReduceLROnPlateau` is used to reduce the learning rate when the model's improvement is slowing down, as measured through the validation loss. Before the next deliverable, I would like to add class weights, to make up for the class imbalance between the number of paintings per artist.

4 Preliminary Results

The model was trained for 10 epochs, which took approximately two hours of time. The validation method used is to compute the loss as the categorical crossentropy, which is useful for classification problems where only one label is correct, as it is the case for this problem, and the validation metric chosen is the accuracy, which might be problematic considering the fact that our classes are somewhat imbalanced.



```
Epoch 1/10
187/188 [============================>.] - ETA: 2s - loss: 0.9940 - acc: 0.5941Epoch 1/10
188/188 [==============================] - 599s 3s/step - loss: 0.9931 - acc: 0.5952 - val_loss: 16.3375 - val_acc: 0.1870
Epoch 2/10
187/188 [============================>.] - ETA: 3s - loss: 0.6521 - acc: 0.7882Epoch 1/10
188/188 [==============================] - 598s 3s/step - loss: 0.6531 - acc: 0.7872 - val_loss: 14.4916 - val_acc: 0.3000
Epoch 3/10
187/188 [============================>.] - ETA: 2s - loss: 0.6350 - acc: 0.7930Epoch 1/10
188/188 [==============================] - 593s 3s/step - loss: 0.6354 - acc: 0.7925 - val_loss: 4.2204 - val_acc: 0.3065
Epoch 4/10
187/188 [============================>.] - ETA: 2s - loss: 0.5648 - acc: 0.8177Epoch 1/10
188/188 [==============================] - 593s 3s/step - loss: 0.5662 - acc: 0.8171 - val_loss: 1.8848 - val_acc: 0.3696
Epoch 5/10
187/188 [============================>.] - ETA: 2s - loss: 0.5065 - acc: 0.8429Epoch 1/10
188/188 [==============================] - 592s 3s/step - loss: 0.5051 - acc: 0.8437 - val_loss: 1.2322 - val_acc: 0.5717
Epoch 6/10
187/188 [============================>.] - ETA: 2s - loss: 0.4514 - acc: 0.8681Epoch 1/10
188/188 [==============================] - 593s 3s/step - loss: 0.4527 - acc: 0.8683 - val_loss: 0.6834 - val_acc: 0.7717
Epoch 7/10
187/188 [============================>.] - ETA: 2s - loss: 0.4401 - acc: 0.8681Epoch 1/10
188/188 [==============================] - 590s 3s/step - loss: 0.4389 - acc: 0.8688 - val_loss: 0.5989 - val_acc: 0.8174
Epoch 8/10
187/188 [============================>.] - ETA: 2s - loss: 0.4064 - acc: 0.8804Epoch 1/10
188/188 [==============================] - 588s 3s/step - loss: 0.4065 - acc: 0.8805 - val_loss: 0.6259 - val_acc: 0.7826
Epoch 9/10
187/188 [============================>.] - ETA: 2s - loss: 0.3958 - acc: 0.8965Epoch 1/10
188/188 [==============================] - 588s 3s/step - loss: 0.3964 - acc: 0.8960 - val_loss: 0.5991 - val_acc: 0.8022
Epoch 10/10
187/188 [============================>.] - ETA: 2s - loss: 0.3517 - acc: 0.8922Epoch 1/10
188/188 [==============================] - 590s 3s/step - loss: 0.3541 - acc: 0.8912 - val_loss: 0.6117 - val_acc: 0.7826
```

As can be seen from the training and validation results above, the model reaches a maximal performance of 89.6% on the training set and 81.7% on the validation set. For the next deliverable, I will make a more in depth analysis of the results, such as using a confusion matrix, to better estimate what kind of optimization I could do on the model.

5 Next Steps

For the next steps, I will work on all the aforementioned points. Furthermore I would like to try varying different hyperparameters and maybe also different neural network structures, but I am very limited in this regard by the long training time of the model. Another interesting approach to the problem was suggested in [4] using a multi-class SVM.

References

1. *Best Artworks of All Time* database from `https://www.kaggle.com/ikarus777/best-artworks-of-all-time`

2. *Art Challenge - A Quiz Game of Famous Painters* by `artchallenge.ru`

3. Nitin Viswanathan, *Artist Identification with Convolutional Neural Networks*, Stanford University

4. Alexander Blessing and Kai Wen, *Using Machine Learning for Identification of Art Paintings*, Stanford University

5. Supratim Haldar, *DeepArtist : Identify Artist from Art* on `https://www.kaggle.com/supratimhaldar/deepartist-identify-artist-from-art`