Comparative analysis of modifications on Two-Stage Deep Learning approach to the Classification of Fine-Art Paintings*

*Note: Sub-titles are not captured in Xplore and should not be used

Almat Kairbek
The Department of Computational Sciences
Astana IT University
Astana, Kazakhstan
211044@astanait.edu.kz

***Abstract*—This paper introduces the comparisons of new modifications into two-stage deep learning approach that improves the style classification accuracy with existing one. The modification was made in the first stage, instead of dividing the image into five patches, it was divided into nine patches to not lose the data in central part of the each side. A comparison analysis was made in order to identify how increasing the size of WikiArt dataset affects the overall results of the accuracy. The method was tested on VGG-16 and ResNet-50.**

I. Introduction

Image classification has two type of labelling. The first is object recognition as an recognising the objects within an image and the second is semantic categorization known as labeling based on the mood and emotions shown in the image [1]. If the first type is relatively well explored, the second one is still need more research works to get the good results.

Popular paintings are the work of human being. Every artist put their own sights to painting and make an art work in a particular style. Sometimes, identifying the style of art piece is difficult even for the experts of fine arts. In visual arts, style is defined as a set of distinctive elements that can be associated with a specific artistic movement, school or time period [2]. So, style is an important sentiment criteria for the image classification in fine arts. The ability of the machine to classify styles implies that the machine has learned an internal representation that encodes discriminative features through its visual analysis of the paintings [3], which tells us how far machine can even go if the task will be fully successfully achieved.

One of the latest method that was found as a best way for style recognition is Convolutional Neural Network (ConvNet, CNN). CNN is a basic tool for classifying and recognizing objects, faces in photos, and speech recognition. There are many applications of CNN, such as Deep Convolutional Neural Network (DCNN), Region-CNN (R-CNN), Fully Convolutional Neural Networks (FCNN), Mask R-CNN and others. The convolutional neural network by using a special operation - the convolution itself - allows to simultaneously reduce the amount of information stored in memory, so it can better handle images of higher resolution, and to isolate the reference image features, such as edges, contours or faces. In the next layer of processing, these edges and faces can be used to recognize repeatable texture fragments that can be further folded into image fragments.

Essentially, each layer of the neural network uses its own transformation. If in the first layers the network operates with such notions as ”edges”, ”faces”, etc., then further the notions of ”texture”, ”parts of objects” are used. As a result of this elaboration, we are able to correctly classify the picture or identify, in the final step, the object we are looking for in the image.

A classic representative of CNN is AlexNet [4]. The first study in this field that showed that the networks can smoothly arrange the work of art only by using the style labels, without any comprehensive information. It was a big breakthrough in this field, not only it represented a new way of solving this problem, but also the results of identifying art styles were increased. But still it was not enough to fully understand how properly recognize the style in fine arts.

Later, many other works improved the results of AlexNet. The CNN models started become more complex, big, with many different layers. VGG16 [5] was created in order to reduce the number of parameters in the convolutional layers. ResNet-50 [6] concentrated on finding a simpler mapping system by introducing two types of shortcut connections. While the others was going in a direction of increasing the number of layers, Inception [7] worked on making the layers wider. Those were the CNNs developing and spending time on improving the models. The way of how everything needs to work. But there are also other ways of improving overall results of style classification: working with the image itself.

Most of the standard CNN architectures work with fixed, small size paintings, which means that high-resolution art pieces should go through downsampling process. It surely leads to some loss of information. A new study shows that

Identify applicable funding agency here. If none, delete this.

this problem can be faced by using sub-region approach [1], in which initial image divides into smaller regions, and those patches have the same standard size as required by CNN. Also, this study proposes two stages of art style recognition. The first stage proposes dividing image into patches and by using deep neural network get the parameters. The second stage suggests, concatenating the vector parameters into one, and by using shallow neural network labeling image the final result. This paper's work is based on this two-stage deep learning approach.

The remainder of this paper is organized into five sections:some theoretical information of experiment in Section II; proposed method in Section III; implementation of the method and experiment in Section IV; and Section V describes the results and the last Section VI is conclusion.

II. Theoretical Background

This section aims to explore the theoretical background of the key architectures and technologies utilized in the proposed method, which include convolutional neural network models such as VGG16 and ResNet50. Also, this section explains the term transfer learning and its usage in the experiment.

A. VGG16

VGG16 is a classification algorithm and type of CNN constructed by Karen Simonyan and Andrew Zisserman (University of Oxford) [5] in 2015. It can classify 1000 images of 1000 categories with 92.7% accuracy, so that, nowadays, it is one of the best algorithms of image classification problem.

VGG16 has thirteen convolutional layers, five Max Pooling layers, and three Dense layers which sum up to twenty-one layers but only sixteen of them have weight layers, which explains its name as VGG16. The input size to ConvNet is a fixed-size 224 × 224 RGB image as shown in Fig. 1. The architecture of the algorithm focused not on having a large number of hyper-parameters but on having convolution layers of 3x3 filter with stride 1 and always used the same padding and maxpool layer of 2x2 filter of stride 2. The creators of this model pushed the depth to 16 weight layers making it approximately — 134 million trainable parameters.

B. ResNet50

Unlike traditional sequential architecture as AlexNet and VGG16, ResNet relies to "building blocks" architecture, which is a collection of micro-architectures building a bigger one. The architecture was introduced in 2015, in paper "Deep Residual Learning for Image Recognition" written by Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun and based on residual learning [6]. It is a type of learning, where the connections between layers are a little different. The output of one earlier convolutional layer connects to the input of another future convolutional layer several layers later as. The functional block diagram of the architecture is shown in Fig. 2. ResNet50 consists of 50 layers deep: 48 convolution layers along with 1 maxpool and 1 average pool layer. The input size of an image is the same as VGG16 — 224-by-224.

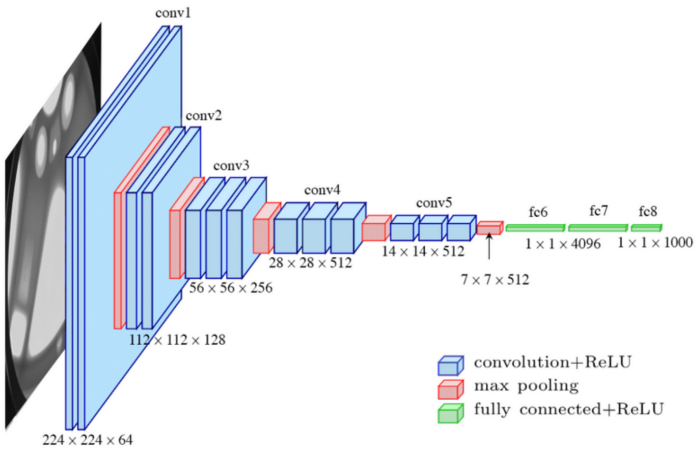

Fig. 1. The architecture of VGG16

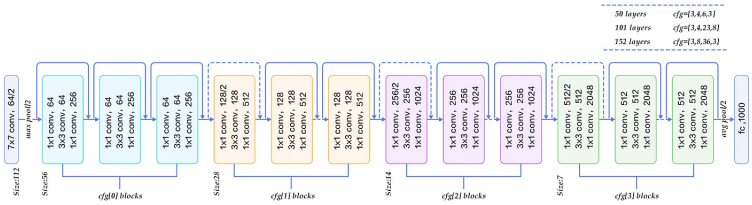

Fig. 2. The architecture of ResNet50

C. Transfer Learning

Transfer learning is the reuse of a pre-trained model as a starting point for a new problem. Nowadays, it is an important part of deep learning. It try to transfer the knowledge from the source domain to the target domain as shown in Fig. 3, by relaxing the assumption that the training data and the test data must be i.i.d [8]. So that, the pre-trained model's weights and learned features are transferred to the new model, but the final classification layer of the model is replaced to adapt to the new task. Transfer learning is effective when the pre-trained model has learned generic features that are relevant to the new task, especially when the new task has limited training data. This improves the efficiency and solve the problem of training on insufficient data.

D. Fine-tuning

Fine-tuning is a process where pre-trained model is used to train the data on a new dataset. It is a continuation of the transfer learning process. It allows to change not only final layers of the pre-trained model, but also, get access and unfroze some of the model's earlier layers. Fine-tuning is good when the task is similar to pre-trained model's orginal task because it easily adapts to a new task and fine-tune on new data effectively.

E. Shallow neural

A shallow neural network consists of only one or a small number of hidden layers positioned between the input and output layers [9]. The input layer receives the data, the hidden

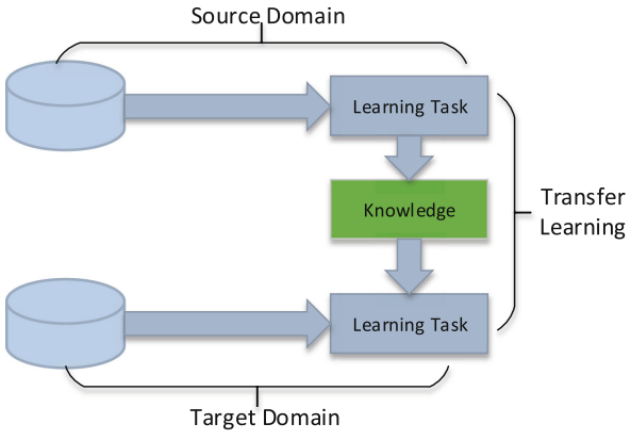

Fig. 3. Learning process of transfer learning

layer(s) perform computations on it, and the output layer generates the final output.

III. Proposed Method

A. General Explanation

1) Step 1 - Patch Extraction: As it was mentioned before the proposed method is a modified method of already existing one. Two stages were presented as shown in Fig. 1. In the first stage, particularly is Step 1, the image was divided into nine same-sized patches. But before that, image size was scaled according to the standard of CNN model. Four patches divide the image into equal squares: upper right, upper left, lower right, lower left. Then another four was taken by overlapping the 50% of neighbouring two patches: center-right, center-left, center-up, center-down. And the last one overlaps the 25% of the first four patches.

2) Step 2 - Classifier 1: In Step 2, all 9 patches generated in Step 1 goes through CNN model. The model trains on the dataset, so that it does not have pre-trained values. The size of the last fully-connected layer is determined by the number of artistic styles. And this Softmax layer determines the probability of each style, so that image could be painted in that style. After running through training parameters of each patch were taken in Step 3. These parameters formed the C_{i_j} output vector.

$$C_{i_j} = p_{ij_{k=1,\dots,L}} \tag{1}$$

where *i* is the index of the analyzed input image (*i = 1,...,M*), *j* is the patch number (*j = 1,...,N*), and *k* is the style index (*k = 1,...,L*).

3) Step 3 - Probability Vector Assembling: The probability vectors C_{ij} generated in Step 2 that belong to the same image (*i = 1,...,M*) are concatenated into a single vector of probabilities I_i as given in (2)

$$I_i = [p_{i,1,1}, p_{i,2,1}, \dots, p_{i,N,1}, \dots\dots, p_{i,1,L}, p_{i,2,L}, \dots, p_{i,N,L},] \tag{2}$$

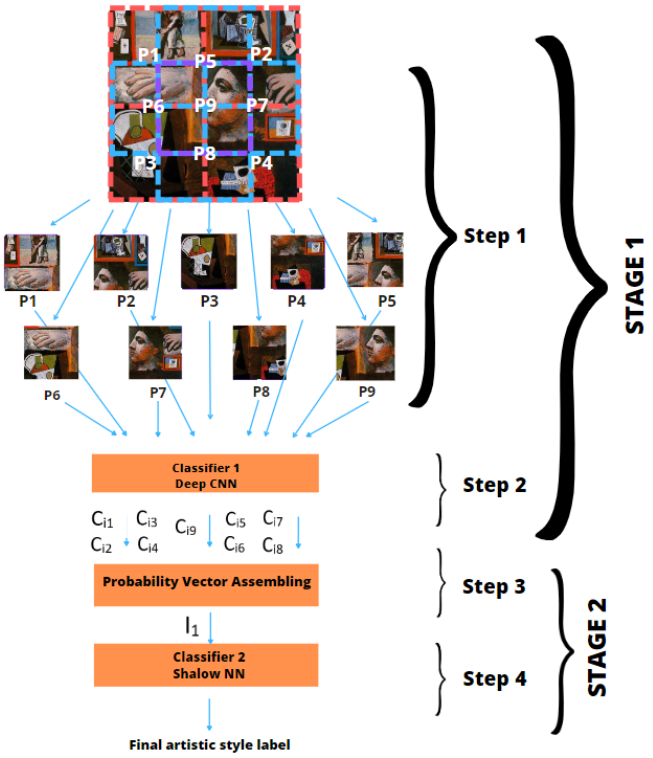

Fig. 4. The proposed two-stage classification framework using nine image patches; i is the analyzed image index.

After that each image's patch output vectors were concatenated into one image vector as in (2).

4) Step 4 - Shallow Neural Network: The final concatenated vectors then used as features for the second stage, specifically in Step 4. As features they run through the second classifier. The shallow neural network again train them. But this time on a probability vectors not the images. After training at first and then at second classifier the final artistic style labels and results shows overall accuracy and loss.

B. Programming

The Fig. 6 illustrates the flowchart of the the whole two stage classification method. First, we prepare the dataset and by preprocessing we divide it to train, validation and test sets. Then we load the model and it's already trained parameters. By utilizing transfer learning we fine tune the model to our train and validation sets. After training on them, we save the model. The next step is to get the vector parameters of trained features. Therefore, we predict the model on train set, then we save those values in a new list. After that we need to group the values by image patches, concatenate them and store in one list(i.g. one image was taken as a set of ten patches, each patch contains six (styles) probability vectors. After concatenating them new list will consists of lists(the number of lists equals to the number of images), and each list contains sixty probability vectors. Starting from a point, where we predict the model on the train set, simultaneously we execute identical steps on the test set. Subsequent to this, we generate a new list of

labels, so that each label corresponds to a particular class, and they are represented as integers from zero to five (according to our six styles) containing true values for each vector in a concatenated train and test lists. Next, we create a shallow neural network (SNN) and train it on concatenated train values. SNN has three layers as shown in Fig. 5. One fully connected layer, where every neuron is connected to every neuron in the previous layer. Its input size is 60 features. Another one fully connected layer that has 128 neurons and uses ReLU as the activation function. This layer takes the shape of the previous layer's output as its input. And the third Dense layer that has 6 neurons representing the output classes of the model. As an activation function was used Softmax.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 60)                3660

 dense_1 (Dense)             (None, 128)               7808

 dense_2 (Dense)             (None, 6)                 774

=================================================================
Total params: 12,242
Trainable params: 12,242
Non-trainable params: 0
```

Fig. 5. Shallow Neural Network

The last step is evaluating the model on concatenated test values with the list of our labels. As a result we get overall accuracy and loss of the model.

IV. Implementation and Experiment

A. Dataset

The dataset was taken from kaggle [10] - "WikiArt all artpieces". Dataset contained 176436 pictures and in 193 different styles. The number of pictures for each style varied from 1 to 16083. Therefore, the number of pictures and styles in pre-processing stage were decreased. Six popular styles were taken as in Fig. 6: Abstract Expressionism, Cubism, Minimalism, Realism, Surrealism and Symbolism. Some of the artistic styles have similar patterns in their paintings such as geometrical figures and shapes and color intensity as in Cubism, Minimalism, Symbolism and Surrealism. But there is also a distinguished style as Realism, which mostly have people in paintings. The total size of dataset that was used to train the CNN model are 2,130 pictures (dataset 1) and 4,260 pictures (dataset 2). Firstly, there were taken two different sized datasets to see the differences and changes in a model training. Secondly, the reason for taking that amount of pictures is that when each picture is divided into 9 patches, the total number of dataset increases and make 21,300 and 42,600 images including the initial picture itself. The amount of pictures in every styles are balanced. Dataset was divided into train, validation, and test sets by proportion 70/10/20. In dataset 1, train set has 15000, validation set - 2,100, test set - 4,200 image patches. Dataset 2 consists of train set - 30,000, validation set - 4,200, test set - 8,400. Each set

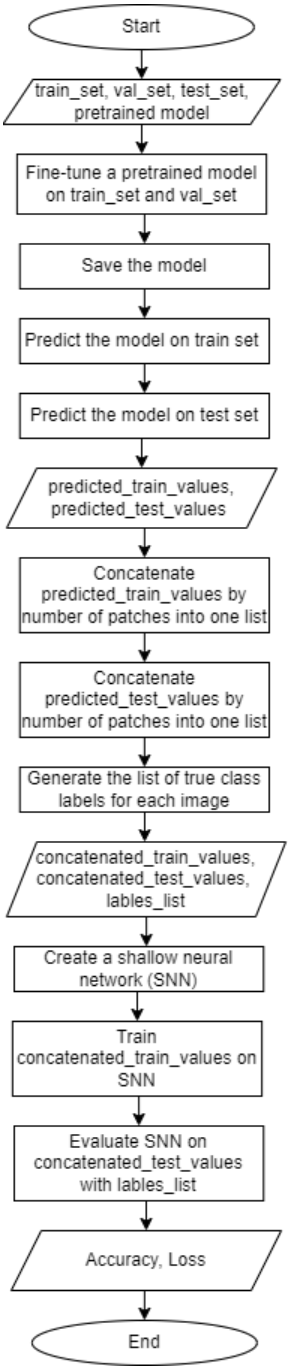

Fig. 6. The flowchart of the proposed method

contains six folders and each style has 355 and 710 image patches, respectively. Also, there is a third dataset (dataset 3), which contains 25,560 image pacthes. This dataset was made by dividing picture into not nine but five patches. So, in total one picture is taken as set of six patches. As another datasets this one was also divided into train (18,000), validation (2,520) and test (5,040) sets. Each style folder has 426 image patches

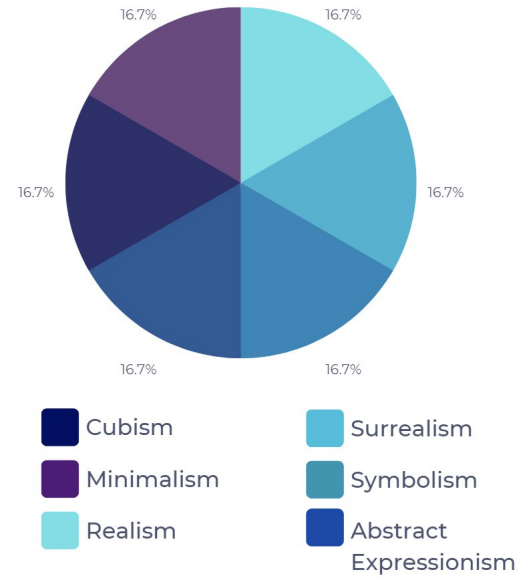

Fig. 7. The flowchart of the proposed method

B. Setup

For training the model it was taken two different CNNs: VGG16 [5] and ResNet [6] (Table 1). The advantages of the architectures:

- VGG16 uses very small receptive fields and fully-connected layers by that it showed high scores in the ImageNet Challenge.
- The ResNet architecture does not need to fire all neurons in every epoch. This greatly reduces the training time and improves accuracy. Once a feature is learnt, it does not try to learn it again but rather focuses on learning newer features. A smart approach that greatly improved model training performance.

TABLE I
COMPARISON OF CNN MODELS

Model	No of layers	Architecture	Input Size	Parameters (Millions)
VGG-16	16	linear	224x224	14.8
ResNet50	50	residual blocks	224x224	24.1

In experiment 80% of the data was utilized to train the CNN models, while the remaining 20% was used to assess the system's performance. The results of perfomance were measured by accuracy and loss.

The proposed two-stage method was used in experiment described in Section III. The probability vectors obtained during the first-stage classification of individual patches of a given input image were assembled into feature vectors, as in (2) and passed to the second stage classifier to determine the final label for the analyzed image. At the end, the results showed the overall accuracy.

C. Experiment 1: VGG16

In experiment 1, VGG16 architecture was used for training. Firstly, experiment was conducted on dataset 1. Dataset was inputted and first classifier VGG16 was loaded. Total parameters number was 14,865,222, but only 150,534 of were trained. The reason for that is only last Softmax layer were used for training and the previous layers were just frozen. After that we fine tuned the model on our dataset 1 and then retrieved 15,000 train and 4,200 test parameters. By concatenating them, 1,500 train and 420 test probability vectors were taken. As in stage 4 in Fig. 4, train vectors were trained on shallow neural network. Lately, trained model was evaluated on concatenated test set a d results were taken. The same steps were used in on dataset 2. So that we received 30,000 train and 8,400 parameters from classifier 1. After concatenating each 10 image patch parameters, 1500 train and 840 test probability vectors were taken. Then by training them on classifier 2 - SNN and evaluating them, we received second dataset's accuracy and loss results. Lately, trained model was evaluated on concateneted test set a d results were taken. The same steps were used on dataset 2. So that we received 30,000 train and 8,400 parameters from classifier 1. After concatenating each 10 image patch parameters, 1500 train and 840 test probability vectors were taken. Then by training them on classifier 2 - SNN and evaluating them, we received second dataset's accuracy and loss results.

D. Experiment 2: ResNet50

Experiment 2 was conducted the same way as experiment 1. But instead of VGG16, ResNet architecture was used for training. As it was in the previous experiment, Firstly, experiment was conducted on dataset 1.

E. Experiment 3: Dataset 3

The third experiment was conducted on both models: VGG16 and ResNet50. But this experiment focused on comparison of taking not only ten patches of one image but also 6 patches as it was mentioned in original work. Dataset 3 were trained on both models and then tested on SNN. So that we can compare the results of six and ten image patches.

V. RESULTS

With implementing the two-staged method and conducting the experiment the following results were taken (Table 2).

A. Experiment 1 - VGG16

After fine tuning the model the first classifier's training accuracy were around 77% and loss 24% on dataset 1 and 79.4% of accuracy and 27% loss on dataset 2. The results are almost the same. But after training them on the second classifier accuracy showed around 97% results on both datasets. But after testing the SNN model the final accuracy were different on datasets. Dataset 1 showed 58.09%, while on dataset 2 result was 63.57% as shown in Table II. As expected increasing the number of images can lead to better results. Dataset 2 is twice bigger than the dataset 1, so accuracy growth

was expected. But the change in 5% is still not much as it seems.

B. Experiment 2 - ResNet50

The training accuracy of the first classifier - ResNet50 model showed 37.5% on dataset 1. In comparison with experiment 1, it is twice less. Similar results were taken on dataset 2: 39.1%. On SNN model the results increased but not as much as in experiment 1. It showed around 59.9% on training for both datasets. But the final testing results accuracy dramatically decreased. 37.85% on dataset 1 and 49.52% on dataset 2 as shown in Table II. These are low results, comparing to experiment 1. We assume the reason for that is that the ResNet50 has lower parameter numbers and that it does not learn again already learned parameters.

C. Experiment 3 - Dataset 3

In experiment 3, as it was mentioned previously, we took dataset 3 and used proposed method on it. The results in comparision with dataset 2 are showed in Table IV. Dataset 3 on VGG16 classifier showed 61.21% and on ResNet50 47.73% accuracy. While dataset 2, 63.57% and 49.52% respectively. These accuracy results show us the improvements of the originally proposed method. The growth in accuracy on VGG16 model is 2.36%, while on ResNet50 is 1.79%. Even if the results are small, it is still a big improvement because the changes in that kind of semantic art style recognition, it is hard to improve the results dramatically.

TABLE II
COMPARISON OF ACCURACY (%)

	VGG-16	ResNet50
2130 images	58.09	37.85
4260 images	63.57	49.52

TABLE III
COMPARISON OF LOSS

	VGG-16	ResNet50
2130 images	2.91	1.41
4260 images	1.98	1.75

TABLE IV
COMPARISON OF ACCURACY (%)

	VGG-16	ResNet50
6 patches	61.21	47.73
10 patches	63.57	49.52

VI. CONCLUSION

By making modifications as increasing the number of image patches from six to nine as an input for automatic fine art style classification, the two stage deep learning approach was improved. The proposed approach applied two independently trained stages of classification. While the first stage applied a deep CNN trained directly on image data, the second stage used a shallow neural network trained on the class probability vectors generated by the first stage classifier. The modification on proposed method increased the accuracy results on VGG16 by from 61.21% to 63.57% by 2.36% and on ResNet50 from 47.73% to 49.52% by 1.79% on dataset containing 4260 images of six art styles.

We encountered some difficulties during the project. Fisrt case, we imported our dataset with a lot of pictures of different styles, during recognition by styles, we distributed first by styles then into train(70%), test(10%), validation(20%). which caused the model to not work because the hierarchy was wrong. Then we changed this hierarchy: we first divided it into train(70%), test(10%), validation(20%) and then distributed it by styles. Thus the picture referred to the name of the folder, and then the name of the style. Second case, it was difficult to put output of the first classifier to the input of the second classifier. The further research may improve this paper by conducting experiment of increased dataset, First is increasing the number of styles. Secondly, enlargment the dataset by number of images in it.

REFERENCES

[1] C. Sandoval, E. Pirogova and M. Lech, 1955, “Two-Stage Deep Learning Approach to the Classification of Fine-Art Paintings”, vol. 7, pp. 41770-41781.

[2] L. Fichner-Rathus, Understanding Art, 9th ed. Belmont, CA, USA: Wadsworth, 2010, p. 560.

[3] A. Krizhevsky, l. Sutskevera and E. Hinton, ”ImageNet Classification with Deep Convolutional Neural Networks”, n.d.

[4] Y. Bar, N. Levy and L. Wolf, ”Classification of Artistic Styles Using Binarized Features Derived from a Deep Neural Network”,, vol 8925, January, 2015.

[5] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” CoRR, vol. abs/1409.1556, pp. 1–14, Sep. 2014. [online].

[6] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778. doi: 10.1109/CVPR.2016.9

[7] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Las Vegas, NV, USA, Jun. 2016, pp. 2818–2826. doi: 10.1109/CVPR.2016.308.

[8] Aggarwal, C.C. (2018). Machine Learning with Shallow Neural Networks. In: Neural Networks and Deep Learning. Springer, Cham.

[9] Lopes, S. (2022), WikiArt all artpieces

[10] Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., Liu, C. (2018). A Survey on Deep Transfer Learning. In: Kůrková, V., Manolopoulos, Y., Hammer, B., Iliadis, L., Maglogiannis, I. (eds) Artificial Neural Networks and Machine Learning – ICANN 2018. ICANN 2018. Lecture Notes in Computer Science(), vol 11141. Springer, Cham.