Image Classification on Imbalanced Artwork Datasets

Josafat-Mattias Burmeister*, Konstantin Dobler*, and Nataniel Müller*

Hasso Plattner Institute for Digital Engineering
Josafat-Mattias.Burmeister@student.hpi.de
Konstantin.Dobler@student.hpi.de
Nataniel.Mueller@student.hpi.de

Digitizing art collections is a major challenge for many museums and art galleries. To facilitate and accelerate cataloging photos of artworks, we tackle two classification tasks from the art domain using deep learning: type classification and genre classification of artworks. To train our models, we use the popular transfer learning approach. Since our training dataset is highly imbalanced, our work focuses on coping with imbalanced training data. Our results show that the transfer learning approach can produce very good results even for small and highly imbalanced training datasets. We observed that acquiring or generating additional training data as well as certain data augmentation methods can slightly improve training results. Over- and undersampling techniques, on the other hand, do not seem to be necessary and did not provide a substantial benefit.
To optimize performance in both classification tasks, we experiment with multiple training methods and model architectures. In this way, we obtain good results in both tasks: In the type classification task, we achieve an accuracy of over 99% and an F_1-score above 97% for both the minority and majority class. In the genre classification task, we achieve an accuracy of over 96% and F_1-scores ranging from 88% to 99% for the respective classes.

1 Introduction

In recent years, many museums and art galleries have made great efforts to digitize their collections. For example, Amsterdam's Rijksmuseum [36, 44], London's National Gallery [33], Oslo's National Museum [8, 43] and the J. Paul Getty Museum [14] have made significant parts of their art collections available online. Recently, the COVID-19 pandemic has prompted further institutions to digitize their collections [24, 4]. Digitization involves capturing high-resolution images of the artworks [2, 35] and linking them to related information in digital repositories [31]. These repositories can not only facilitate researchers' access to information [42], but can also be used to provide digital content to museum visitors [1]. In addition, digital collections play an important role in marketing and in digital selling of prints and merchandise [1, 44].

*Authors contributed equally.

However, the creation of digital collections is a major challenge for many institutions. Many museums and galleries own large collections but have limited human and financial resources for digitization [1, 24]. In particular, cataloging photos and updating existing records is expensive and laborious [1, 31]. During cataloging, photos of artworks are linked to metadata such as author, style, genre, and type of artwork. Automated acquisition of these metadata using computer vision techniques could greatly facilitate and accelerate the digitization of museum collections [29, 6, 32]. A wide range of research has already addressed this issue. Some works rely on the extraction of image features and classify them using shallow machine learning models such as SVMs or kNN classifiers [48, 38, 12]. The image features used can be either low-level features such as color histograms and edge maps [48], or feature maps extracted from convolutional neural networks [38, 6]. In contrast, another part of the existing work uses end-to-end deep learning models [41, 23, 37]. Training deep neural networks to classify artworks is challenging because the available datasets are comparatively small [37] and in some cases unbalanced [47]. Labeling of additional training data is often impractical as it requires expert knowledge from the art domain [38, 7]. Therefore, the transfer learning approach became popular in the art domain [7, 37, 23, 41]. In transfer learning, the deep learning models are first pre-trained on large photo datasets such as ImageNet with a different classification task. Subsequently, some layers of the models are re-trained on the actual training set from the art domain to solve a classification task for artworks [37].

This work applies the transfer learning approach to two classification tasks from the art domain: In the first classification task, artworks are to be classified in terms of their type as a painting or a drawing. To our knowledge, similar classification tasks have been studied only by Sabatelli et al. [37] and Mensink et al. [29] so far. The second task is a genre classification of artworks. Most existing work on genre classification uses only the popular Wikiart dataset [38, 41], although additional datasets with genre annotations are available [13, 16, 27, 34, 9]. In this paper, we make use of several other datasets for training.

For both classification tasks, we systematically evaluate different architectures and training methods. We investigate how different techniques for data augmentation and different approaches for coping with imbalanced data affect model performance. In addition, we study whether model performance can be improved by ensemble learning. Through model optimization, we achieve human-like performance in type classification, with F_1-scores above 97 % for both classes on the test set. In genre classification, we also achieve high performance, with F_1-scores above 88 % for all classes on the test set.

2 Related Work

Since the majority of available fine-art datasets contain metadata on style, artist, genre, technique and material, research efforts have been predominantly focused on style, genre, and artist classification [7]. These classification problems have been addressed through two major methodological categories namely classical approaches and deep learning-based techniques [39]. In classical approaches, image features are extracted and classified using shallow machine learning models. Feature extraction approaches are divided into feature engineering and feature learning methods [32].

Feature engineering approaches. In feature engineering approaches, domain-specific knowledge is used for transforming "low-level" feature sets such as brush strokes and color into meaningful image features [38]: Florea et al. [12] use local and global features in combination with shallow machine learning models like SVM, Random Forest Models and k-Nearest Neighbor to classify artworks in terms of the artistic movement. Other works use texture feature extractors that take into account global color features and composition features to classify artworks based on their genre [48].

Feature learning approaches. With the advancements of CNN-based feature learning approaches, Cetinic et al. investigate the use of features derived from pre-trained CNN layers [6]. Results indicate higher accuracies for "high-level" CNN-based feature sets given the problem of genre classification compared to "low-level" feature sets derived by Scale-Invariant Feature Transform (SIFT) [26] and Histogram of Oriented Gradients (HOG) [10].

Deep learning-based approaches. Lecoutre et al. demonstrate the performance of a residual neural network (ResNet50) and a pre-trained AlexNet for genre classification on the Wikiart paintings dataset, achieving an overall accuracy of more than 62% over 25 classes [23]. Emphasizing the importance of brush stroke in fine-art classification, Huang et al. implement a two-channel deep residual network consisting of a RGB channel and a brush stroke information channel. They achieve a test accuracy of 68.96 % using a pre-trained ResNet50 [18]. Sabatelli et al. compare the performance of transfer learning approaches that solely re-train the decision layer of pre-trained CNNs with approaches that also retrain the convolutional layers [37]. While retraining the convolutional layers is computationally more demanding, it also yields better results.
Tan et al. compare different fine-tuning methods on the Wikiart paintings dataset for style, genre and artist classification. They show that pre-trained CNNs with an additional softmax layer (genre accuracy: 74.14 %) outperform a pre-trained CNN with a 1000 dimensional feature extraction layer compressed by PCA and a SVM trained on top [41]. Zhao et al. tackle the same genre classification problem on the Wikiart paintings dataset with pre-training on ImageNet and random initialization in the last fully connected layer and achieve an accuracy of 78.03% [47].

Cetinica et al. [7] evaluate five different fine-tuning scenarios on a range of datasets including the Wikiart paintings on a CaffeNet architecture [19]: They observe that retraining all except the first two convolutional layers yields the best results with a test accuracy of 77.7 %.

Recently, Mohammadi et al. proposed a hierarchical classification approach, based on clustering the Wikiart paintings dataset styles into 7 super-style parent classes P each containing image style children C [30]. Then a hierarchical ensemble of P+1 parallel CNNs predict parent as well as the child class, improving the average F_1-score of a DenseNet121 compared to a hierarchical DenseNet121 by more than 3%. Focusing on the highly imbalanced class distribution of the Wikiart paintings dataset, Joshi et al.[20] train a semi-supervised Ensemble of Auto-Encoding Transformations (EnAET) model [45]: Instead of pre-training the model, autoencoding transformations are used to train the classifiers in a four-block wide ResNet-28-2. When comparing to ResNet50 models with/without data augmentation and fine-tuned over all layers this approach yields a test accuracy of 82.61% compared to the ResNet50 baseline of 50.1% [20].

3 Datasets and Classification Tasks

In this paper, we aim to solve two distinct classification tasks for artworks. Both tasks are to be solved using deep learning models that receive images of the artworks as input. The first task is to determine the type of an artwork. We distinguish two types of artworks, drawings and paintings. Examples from both classes are shown in Figure 1. In the following, we refer to this binary classification task as "type classification". The second task is a multi-class classification task, where artworks are to be classified in terms of their genre. The term "genre" is used in different meanings in existing works. In this paper, we follow the definition from Cetinic et al. [6] and distinguish between the five classical genres of art: genre painting, history painting, landscape painting, portrait, and still life. Example images for each genre are shown in Figure 2. In the following, we refer to this task as "genre classification".

3.1 Training Set

The main training datasets for both classification tasks consist of images provided by the Getty Research Institute ("Getty dataset"). All images shown in Figure 1 and Figure 2 belong to this Getty dataset. The label distribution of the Getty dataset is very unbalanced: Paintings occur much more frequently than drawings (Table 1). The most represented genre are landscape paintings with 1156 samples, the least represented genre are portraits with only 35 samples (Table 2).

To enlarge the training datasets and to address their imbalance, we used data from additional sources in some of our experiments. For the type classification, a subset

(a) Drawings

(b) Paintings

Figure 1: Example images from the classes of the type classification task.

Table 1: Datasets that were used for the type classification task (not all datasets were used in all experiments).

Dataset	Number of suitable training images		
	Drawings	Paintings	Total
Bing[1]	662	0	662
Brill[2]	185	20	205
Getty	746	4425	5171
Kaggle[3,4]	1,231	2,270	3,501
Metropolitan[5]	1,000	0	1,000
Rijksmuseum[6]	14,223	3,593	17,816
Wikiart[7]	3,943	0	3,943
All datasets	21,990	10,308	32,298

[1] https://www.microsoft.com/en-us/bing/apis/bing-image-search-api

[2] https://labs.brill.com/ictestset

[3] https://www.kaggle.com/thedownhill/art-images-drawings-painting-sculpture-engraving

[4] https://www.kaggle.com/ikarus777/best-artworks-of-all-time

[5] https://metmuseum.github.io

[6] https://doi.org/10.21942/uva.5660617

[7] https://github.com/cs-chan/ArtGAN/tree/master/data

(a) Genre painting **(b)** History painting **(c)** Landscape

(d) Portrait **(e)** Still life

Figure 2: Example images from the classes of the genre classification task.

Table 2: Datasets that were used for the genre classification task (not all datasets were used in all experiments).

Dataset	Number of suitable training images					
	Genre	History	Land-scape	Portrait	Still life	Total
Art500k[1]	14,752	8,394	18,632	19,593	3,467	64,838
Europeana[2]	11	0	4,368	6,213	1,068	11,660
Getty	763	898	1556	35	421	3673
SemArt[3]	1,813	8,931	2,779	3,650	1,029	18,202
WGA[4]	2,897	1,4507	4,474	5,184	1,435	28,497
All datasets	20,236	45,010	42,287	35,984	7,648	151,165

[1] `https://deepart.ust.hk/ART500K/art500k.html`
[2] `https://pro.europeana.eu/page/search`
[3] `http://noagarciad.com/SemArt/`
[4] `https://www.wga.hu/index1.html`

of the Brill Iconclass AI Test Set ("Brill dataset") [34], two datasets from Kaggle ("Kaggle datasets"), a dataset from the Rijksmuseum Amsterdam ("Rijksmuseum dataset") [29], and a subset of the Wikiart paintings dataset ("Wikiart dataset") [41] were used. Table 1 provides an overview of these datasets. Because the datasets differ in quality, in some of our experiments only a subset of the datasets was used. While the Kaggle datasets and the Rijksmuseum dataset included type labels that were suitable for our type classification task, the other datasets required preprocessing of the labels: Images labeled as "sketches" in the Wikiart dataset were assigned to our drawings class. Images from the Brill dataset with the Iconclass codes "48(+354) / art (+ drawing)" or "48C52 / drawing" were assigned to our drawings class and images with the iconclass code "48(+351) / art (+ painting)" to our paintings class. Some images from the Brill dataset were discarded in a manual filtering step because they differed strongly from the rest of the training data. In addition to the previously mentioned datasets, we also downloaded training data from the Collection API of New York's Metropolitan Museum of Art ("Metropolitan dataset") [40] and from the Bing Image Search API ("Bing dataset"). In the case of the Metropolitan Museum API, we retrieved all images from the "Drawings and Prints" department whose "objectName" property was "drawing". For the Bing Image Search API, we used the search terms "anatomical drawing" and "court sketch" to retrieve images of drawings. To ensure high data quality, the query results were filtered manually in both cases.
For the genre classification task, in addition to the Getty dataset, we used a subset of the Art500k dataset ("Art500k dataset") [27], the SemArt dataset ("SemArt dataset") [13], and data from the Web Gallery of Art ("WGA dataset") [16]. In the Art500k, SemArt, and WGA datasets, there is no "history paintings" class, but a "religious paintings" class. Images from this class were classed as history paintings for our genre classification task. Additional training data for the genre classification task were obtained from the Europeana Search API ("Europeana dataset") [9]. An overview of all datasets applicable for the genre classification task is provided in Table 2. Since the listed datasets differ in quality, not all available training data were used in all of our experiments.

3.2 Validation Set and Test Set

For both tasks, a split of the Getty dataset was used for model validation ("Getty validation set"). The validation set for the type classification consists of 1,438 images. Similar to the Getty training set for this task, it is also very unbalanced: 1,271 of the images represent paintings, while only 212 images show drawings. The validation set for the genre classification consists of 1,049 images and is also unbalanced: It includes 445 landscape paintings, 276 history paintings, 208 genre paintings, 109 still lifes and 11 portraits.
To evaluate the models, another split of the Getty dataset was used, which was not used to train or validate the models ("Getty test set"). The test set for the type classification task consists of 750 images. 636 of these are paintings and 114 are drawings. The test set for the genre classification task consists of 156 images. It

includes 47 landscape paintings, 37 history paintings, 19 genre paintings, 50 still lifes, and 3 portraits. For the genre classification task, we also used the SemArt dataset described in Subsection 3.1 to evaluate some of our models. This dataset was not used for training the models in these cases.

4 Methods

We have two distinct problem statements: a binary classification whether an image is a painting or a drawing ("type classification") and a multi-class classification of an image's genre ("genre classification"). Our approaches to these two share many similarities; however, we will also highlight the differences.

4.1 Evaluation metrics for imbalanced datasets

For the Getty dataset, we can trivially construct a classifier with 85% validation accuracy in the type classification challenge by classifying each image as painting. When dealing with imbalanced datasets in general, accuracy is not a good performance metric [3]. Therefore, we use two other metrics that are commonly recommended for such scenarios: the F_1-score and Matthew's Correlation Coefficient. Both metrics account for the frequency distribution of the classes.

A hypothetical classification of 24 images, with TP=18, TN=1, FP=3, and FN=2 would yield an F_1-score of 88%. This would indicate a good classifier, however closer examination reveals the following facts: Only one in four drawings is classified correctly; also, two out of three predicted drawings are actually paintings. One of the weaknesses of the F_1-score is that it is not symmetric; the positive class is given more "weight" and this choice therefore affects the result. Also, the number of true negatives (TN) has no impact on the F_1-score as it does not influence precision or recall. If we were to flip positive and negative classes in our example, the F_1-score would drop to 29%. Matthew's Correlation Coefficient (MCC) is a symmetric performance measure that is also well-suited for imbalanced datasets. It measures the correlation between predicted and actual values (obviously, a high correlation is desirable). It is given by the formula:

$$MCC = \frac{TP \times TN + FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{1}$$

Because the F_1-score was one of the primary metrics by which we were evaluated in the challenges, we still use it as our primary metric. We also use the MCC as a "sanity check"; when the MCC and F_1-score are too far apart, it indicates that our model is not performing well across all classes.

4.2 Type classification

We use the popular DenseNet and ResNet architectures [15, 17]. Motivated by similar research [23, 37] and initial experiments, we make heavy use of transfer learning. In particular, we use the ResNet18, ResNet34, ResNet50 and DenseNet201 architectures with weights pre-trained on ImageNet [11], which are provided by Torchvision [28]. Because of the imbalances in our primary dataset (the Getty dataset), we pay special attention the the F_1-score computed with "drawings" as the positive class.

Data preprocessing. Our dataset contains images that often exceed a dimension of 1000×1000 pixel. Because of hardware limitations, we are forced to re-scale our inputs to a smaller size. Additionally, the images are not square. Therefore, we also apply a quadratic crop before feeding the images to our networks. Although researchers often re-scale their inputs to 224×224, following a convention from AlexNet [22], we also experiment with input dimensions up to 400×400. We expect that larger images contain more information and thus improve the performance of our models.
In order to further speed up the training process, we re-scaled the entire training set so that the smaller dimension (width or height) is equal to 400. This way, we can still apply various cropping techniques but significantly reduce disk I/O, which was a bottleneck in our initial experiments. In particular, we apply either a centered or a random quadratic crop. As we observed that some paintings and drawings are framed or do not fill the entire input image, we also implemented a "random borderless crop": before applying the random crop, we truncate 20 pixels from every side of the input image. In most cases, this was sufficient to remove frames from a picture.
We also used a multitude of different augmentation techniques on our training images. We experimented with random horizontal flipping, perspective transformations, color jittering, grayscale transformations, Gaussian blurs and random rotations, shearing or re-scaling with varying probabilities.

We normalized all images with the channel means and standard deviations from ImageNet. It should be noted that this is suboptimal and could be improved by calculating the means and standard deviations of our actual training set; however, we did achieve satisfactory results.

Model architecture and training details. As mentioned above, we conducted experiments with ResNet50 and DenseNet201. We chose these as the deepest representatives of their respective classes that we could feasibly train on our hardware. However, we also ran experiments with smaller versions of ResNet, such as ResNet18 and ResNet34, which have the benefit of reduced training times. We use transfer learning: the model weights are pre-loaded with weights pre-trained on ImageNet. As we want to adapt to a new problem space, we replace the fully-connected decision layers. Whereas the original architectures use only one fully-connected layer

to produce the one thousand outputs of ImageNet, we also experimented with using 4 consecutive fully-connected layers with the ReLU activation function. The motivation is two-fold: firstly, our problem space requires only one output instead ImageNet's thousand and we theorized that this additional reduction warrants additional layers. Secondly, we believe our problem space to be more complex and additional layers might be able to better capture this complexity. Our best model for the type classification task uses four such fully-connected layers.
There are two main variants of transfer learning: feature extraction and fine-tuning. During feature extraction, only the decision layer is trained on the new dataset, while the rest of the weights are frozen. Therefore, the pre-trained part of the network extracts useful features that the new decision layer then receives as inputs. During fine-tuning, all weights of the network are re-trained to adapt the entire network to the new problem space. Of course, one can also re-train only selected layers such as the last convolutional layer during fine-tuning.

During our experiments, we try feature extraction as well as differing degrees of fine-tuning. In all cases, we do not freeze the weights of batch normalization layers. These layers capture statistics of the underlying dataset, which is ImageNet in the cases of our pre-trained weights. As we use a different dataset, we want these statistics to be updated according to our new dataset. We found that fine-tuning leads to superior results compared to feature extraction. Since our dataset consists of paintings and drawings, which are not part of ImageNet, we expected to achieve better results when adapting larger parts of our model to the new dataset.

We use the AdaDelta and Adam optimizers [21, 46] and a binary cross-entropy loss. We run experiments with cosine learning rate scheduling [25], step-wise learning rate decay, and constant learning rates. We employ a version of *early-stopping*: after every epoch, we evaluate our model on the validation set and store the result as well as the model weights. At the end of the training, we can then choose the model weights that yielded the best results.

Combating data imbalances. A significant challenge was the imbalance of our primary dataset. In the following, we detail several different strategies that we implemented to address this problem.

If we train our models only on the Getty dataset, they will see many more paintings than drawings. This can be counteracted by sampling drawings at a higher rate than paintings, so that the model is presented an even distribution between the two classes. Since we have few original drawings, this should be coupled with strong augmentation techniques. Otherwise the model might overfit to the drawings that it is repeatedly shown. There are two "flavours" of this sampling technique: oversampling and undersampling. Oversampling works as described by sampling the minority class at a higher rate, while undersampling works by sampling the majority class at a lower rate to make the two classes balanced. While oversampling has the drawback of potential overfitting, undersampling can cause loss of information

as majority-class samples are randomly excluded from the training set each epoch.

Another solution is to use additional datasets that contain drawings to achieve balance. As can be seen in Section 3, we experimented with many additional datasets. Because they also did not have an even distribution of paintings and drawings, we combined the additional datasets with oversampling or undersampling. In some cases, depending on the additional datasets that were used, paintings became the minority class and had to be oversampled (or drawings undersampled).
Another approach we have taken is to generate artificial training data. We perform a rudimentary style transformation of paintings to drawing-like images. Paintings are first converted to greyscale, then a gaussian filter with a σ of 10 is applied. The inverted image and the gaussian filter are then overlayed. After the style transformation, a random resize crop to 224 x 224 pixels, a random horizontal flip and a normalization is applied. We evaluate transformation probabilities of $\frac{1}{3}$ and $\frac{1}{2}$ that regulate the proportion of transformed paintings.

The methods described above operate at the *data level*: they modify the data so that the model is not "aware" of the imbalance. We also implemented methods at the *algorithm level*; specifically, we use two modifications of the loss function. One is a weighted loss, where the influence of wrongly classified minority-class samples on the loss can be increased by a weight factor. The imbalance in the dataset is then offset by the fact that the minority class has more impact on the loss and therefore the training process.
Another modification is to use a different loss that incorporates a balancing mechanism: the F_1-score. This has the added benefit of a closer alignment between the loss that is used to train the model and the primary metric that we want to optimize. Unfortunately, the F_1-score is not guaranteed to be differentiable as there is the possibility of a division by zero. This problem can be overcome by adding a small $\epsilon = 1e\text{-}10$ to the denominator wherever this possibility exists. As we want to maximise the F_1-score, we minimize the F_1-score subtracted from one as our loss.

Other improvements. We employ *test-time augmentation*. Using PyTorch's FiveCrop and TenCrop modules,[1] we feed our model five versions of the same image at test-time: one crop from every corner of the image as well as the center crop. In the case of TenCrop, the model is also fed a horizontally flipped version of every crop. The final prediction is then set to the most common prediction over all crops. We also experiment with an ensemble of three different ResNet models. As the ResNet50 model has more parameters than the smaller ResNets, it is more prone to overfitting, which is why we also included a ResNet34 and ResNet18.

[1] https://pytorch.org/vision/stable/transforms.html#torchvision.transforms.[Five | Ten]Crop

4.3 Genre classification

In the genre classification challenge, we implemented two different approaches: an ensemble-based approach consisting of five one-versus-the-rest classifiers for the five genres and one single-model approach.
The data preprocessing and techniques to combat data imbalances are analogous to the type classification described in Subsection 4.2. However, we use an additional sampling technique: a combination of over- and undersampling. The open-source ImbalancedDatasetSampler[2] aims to achieve a balance between over- and undersampling to alleviate their respective drawbacks.

Ensemble model. To classify the five different genres, we train one model for each genre that classifies whether an image belongs to that genre or not ("one-versus-the-rest"). These models then build an ensemble. The final predicted genre is decided by the network that yielded a positive result with the highest confidence. For the individual models of the ensemble we used the ResNet50 architecture. We train each ResNet50 model with different combinations of augmentation methods and learning rate scheduling. We did not do a systematic search for the best combinations but rather made choices based on intuition. Our first ensemble consisted of the best-performing classifiers for each genre. However, we found that this choice was not optimal. Through manual experimentation, we found an ensemble where the individual classifiers were sub-optimal but together, they outperformed the previous ensemble.

Single model. In contrast to the ensemble approach, we also trained a single model capable of classifying all genres. We experiment with the same model architectures mentioned in Subsection 4.2. Because there are five different genres, our models' final decision layer now has five outputs instead of just one, which are fed into a softmax layer. Instead of binary cross-entropy loss, categorical cross-entropy loss is used.

5 Results and Discussion

In the following, we describe our results for both classification tasks. For the type classification task, we explored multiple training methods. In the genre classification task, we built on these results and used the best training approaches from the type classification task.

[2]https://github.com/ufoym/imbalanced-dataset-sampler

5.1 Type Classification

For the type classification task, we first describe the main findings we obtained from the exploration of different training methods. Subsequently, we describe the model with the best overall performance and compare it to some of our other models.

5.1.1 Impact of Data Augmentation

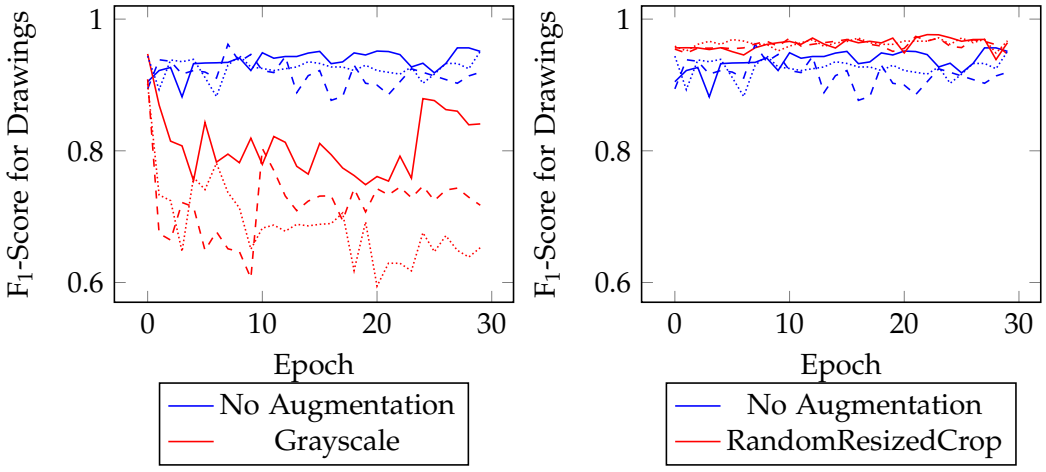

Figure 3: Performance of ResNet50 models trained with different data augmentation methods on the unbalanced Getty dataset. In both plots, we show the F_1-score for the drawings class on the Getty validation set.

Since the Getty dataset is comparatively small, we experimented with various augmentation methods to prevent model overfitting. To identify the most suitable augmentation methods, we trained ResNet50 models on the Getty dataset with one augmentation method each and compared them to a baseline model trained without data augmentation. To account for different model initializations, we ran three trainings with 30 epochs for each augmentation method. We determined the best model (best F_1-score for the drawings class on the Getty validation set) from each training and report the averaged F_1-scores of the best models in Table 3. Details on the training setting are also provided in Table 3. As shown in Figure 3, converting the training images to grayscale images significantly degrades the model performance on the validation set compared to the baseline model. Random resizing and cropping of the training images, on the other hand, slightly improves the performance on the validation set. For all other augmentation methods we tested, the performance of the models on the Getty validation set is very similar to that of models trained without data augmentation.
The fact that converting training images to grayscale images significantly degrades the training results suggests that the models strongly account for color information

in discriminating between drawings and paintings. This is surprising given that only a subset of the training set consists of color images. However, among the colored training images, paintings are indeed usually colorful, while drawings use a limited color palette.
One possible explanation for the performance increase achieved by random resizing and cropping is that it prevents overfitting to certain image features. However, it is unclear why random cropping does not have a similar effect on model performance. Our custom "border cropping" technique that removes image frames led to a slight decrease in performance. It could be that the image frames, which we initially thought of as clutter or noise, actually contain information that our model can learn from.

5.1.2 Coping With Unbalanced Training Data

As described in Subsection 4.2, we implemented several approaches to address the imbalance in the Getty training set. Below, we describe how these approaches impact model performance.

Oversampling and undersampling. To examine how oversampling of the minority class (drawings) or undersampling of the majority class (paintings) affects model performance, we trained three ResNet50 models for 30 epochs with each sampling method. To determine the baseline performance, we trained three Resnet50 models on the unbalanced Getty dataset. We determined the best model (best F_1-score for the drawings class on the Getty validation set) from each training and report the averaged F_1-scores of the best models in Table 4.
We expected that undersampling the majority class would degrade the model performance because information is lost by randomly discarding training images. On the other hand, we expected that oversampling would improve the results by counteracting a model bias in favor of the majority class. However, as our results in Table 4 show, this did not prove true. All three ResNet50 models achieve very similar performance on the Getty test set. Similar observations were made when training models with other architectures, e.g. ResNet18. This demonstrates that transfer learning yields satisfactory results even on highly imbalanced datasets and that over- or undersampling techniques are not necessarily needed.

Additional training data. We expected that models trained on larger training sets would perform better and overfit less. To verify this assumption, we trained ResNet50 models on the Getty dataset and one additional dataset each. To exclude differences caused by different distributions of the datasets, we balanced the training sets by random undersampling. Table 5 shows for each dataset combination the metrics of the model that achieved the highest F_1-score for the drawings class on the Getty validation set. Despite our assumption, not all datasets improve performance compared to the model trained only on the Getty dataset. Possibly this is because some datasets differ too much from the Getty dataset in terms of style and era of the artworks. In particular, the images in the Bing and Brill datasets differ significantly from those in the Getty dataset. In contrast, the Rijksmuseum dataset, which is stylistically most similar to the Getty dataset and represents the

Table 3: Performance of ResNet50 models trained on the Getty dataset with different data augmentation methods. The augmentation methods are sorted by the F_1-score for the drawings class on the Getty evaluation set in descending order.

Augmentation Method	Validation Set		Test Set	
	F_1-Score for Drawings	F_1-Score for Paintings	F_1-Score for Drawings	F_1-Score for Paintings
RandomResized-Crop	0,974 ± 0,002	0,996 ± 0,000	0,971 ± 0,005	0,995 ± 0,001
No Augmentation	0,956 ± 0,004	0,993 ± 0,001	0,967 ± 0,009	0,994 ± 0,002
ColorJitter	0,959 ± 0,004	0,993 ± 0,001	0,965 ± 0,008	0,994 ± 0,001
Border Cropping	0,946 ± 0,008	0,991 ± 0,001	0,965 ± 0,002	0,994 ± 0,000
RandomPerspective	0,961 ± 0,002	0,994 ± 0,000	0,964 ± 0,000	0,994 ± 0,000
RandomHorizontal-Flip	0,959 ± 0,005	0,991 ± 0,003	0,963 ± 0,006	0,993 ± 0,001
RandomRotation	0,958 ± 0,004	0,993 ± 0,001	0,962 ± 0,008	0,993 ± 0,001
RandomAffine	0,948 ± 0,004	0,992 ± 0,001	0,961 ± 0,006	0,993 ± 0,001
RandomCrop	0,955 ± 0,005	0,993 ± 0,001	0,954 ± 0,014	0,992 ± 0,002
Grayscale	0,919 ± 0,020	0,986 ± 0,004	0,920 ± 0,022	0,985 ± 0,005

Models were pre-trained on ImageNet; batch norm, conv. 4, conv. 5 and fc. layers were fine-tuned for 30 epochs on the unbalanced Getty dataset. Training images were resized and cropped to 224 x 224 pixels. A batch size of 100 was used. An Adam optimizer and a cosine annealing learning rate scheduler were used (initial learning rate: 10^{-4}, η_{min} = 0). For all augmentation techniques except border cropping, the implementations from the torchvision package were used.[1] For border cropping, a custom implementation was used as described in Subsection 4.2. For the augmentation methods from the torchvision package, the default parameters were used unless otherwise specified below:

ColorJitter: brightness=0.5, contrast=0.5, saturation=0.5, hue=0.1

RandomAffine: degrees=(0, 40), translate=(0.0, 0.4), scale=(0.6, 1.4), shear=0.2, resample=BICUBIC

RandomCrop: size=224, images were resized to 224 pixels (longer edge) in advance

RandomResizedCrop: size=224, images were resized to 300 pixels (longer edge) in advance

RandomRotation: degrees=360

[1] https://pytorch.org/vision/stable/transforms.html

Table 4: Performance of ResNet50 models trained on the Getty dataset using different sampling methods.

Dataset	Validation Set		Test Set	
	F_1-Score for Drawings	F_1-Score for Paintings	F_1-Score for Drawings	F_1-Score for Paintings
Getty, unbalanced	0,956 ± 0,004	0,993 ± 0,001	0,967 ± 0,009	0,994 ± 0,002
Getty, random undersampling	0,960 ± 0,004	0,994 ± 0,001	0,966 ± 0,008	0,994 ± 0,001
Getty, random oversampling	0,955 ± 0,006	0,993 ± 0,001	0,965 ± 0,005	0,994 ± 0,001

Models were pre-trained on ImageNet; batch norm, conv. 4, conv. 5 and fc. layers were fine-tuned for 30 epochs. Training images were resized and cropped to 224 x 224 pixels. A batch size of 100 was used. An Adam optimizer and a cosine annealing learning rate scheduler were used (initial learning rate: 10^{-4}, $\eta_{min} = 0$).

largest dataset, yields the model with the best performance. This indicates that additional, high quality datasets can improve the models. However, the transfer learning approach allows to obtain satisfactory results even with small datasets.

Artificial training data. Besides acquiring additional training data, we also experimented with algorithmically converting paintings into drawing-like images. The results of these experiments are listed in Table 6. For each experiment, we report the highest micro-averaged F_1-scor achieved on the Getty validation set. Converting a portion of the paintings to drawing-like images slightly improved results over the baseline model. This indicates that the models consider color and edge information when discriminating between drawings and paintings, as our style transformation mainly changes colors and edges of the images. This is consistent with our observation from the experiments with data augmentation, which showed that color information is very important for the type classification task.

Adapted loss functions. As described in Subsection 4.2, we also tested two custom loss functions. In the first loss function, the cross-entropy loss of the minority class was weighted higher by a factor. Intuitively, this factor should be chosen according to the relation of drawings to paintings in the training set. This did not prove correct in our initial experiments and we had to manually fine-tune this factor to give drawings even more weight in order to achieve satisfactory results. Because of this difficulty and the introduction of yet another hyper-parameter in form of the weight factor, we discarded this strategy in our further experiments. Using the F_1-score as loss function also did not improve the results. Therefore, for simplicity, we used only the binary cross-entropy loss in our other experiments.

Table 5: Performance of ResNet50 models trained on the Getty dataset and one additional dataset each. The models are sorted by the F_1-score for the drawings class on the Getty test set in descending order.

Dataset	Validation Set		Test Set	
	F_1-Score for Drawings	F_1-Score for Paintings	F_1-Score for Drawings	F_1-Score for Paintings
Getty + Rijksmuseum	0.9644	0.9941	0.9735	0.9953
Getty + Wikiart	0.9573	0.9929	0.96	0.9929
Getty + Metropolitan	0.9567	0.993	0.96	0.9929
Getty	0.954	0.9926	0.96	0.9929
Getty + Bing	0.9571	0.9929	0.9469	0.9906
Getty + Brill	0.9592	0.9933	0.9432	0.9898
Getty + Kaggle	0.9471	0.9914	0.9417	0.9898

The models were pre-trained on ImageNet; batch norm, conv. 4, conv. 5 and fc. layers were fine-tuned for 30 epochs. Training images were resized and cropped to 224 x 224 pixels. The training sets were balanced by random undersampling and a batch size of 100 was used. An Adam optimizer and a cosine annealing learning rate scheduler were used (initial learning rate: 10^{-4}, $\eta_{min} = 0$).

5.1.3 Model With the Best Overall Performance

The overall best performance in the type classification task was achieved with a DenseNet201 pre-trained on the ImageNet dataset. We fine-tuned all layers of the model using the Getty dataset, the Kaggle datasets, the Metropolitan dataset and the Wikiart dataset. Images were resized to 400×400. We apply random horizontal flipping and random affine augmentations[3]. During training, the drawings class was randomly oversampled. The model optimization was done using the binary cross-entropy loss, a constant learning rate of 0.005 and the AdaDelta optimizer. When generating predictions, we used FiveCrop for test-time augmentation, as described in Subsection 4.2. With this configuration, we achieve an F_1-score of 0.9825 for the drawings class on the Getty validation set. On the Getty test set, we achieve an F_1-score of 0.9739 for the drawings class and an F_1-score of 0.9953 for the paintings class (Table 7). Overall, this is our best result, but we have obtained similar results with other model architectures and configurations. For example, the ResNet50 listed in Table 7 that was trained on the Getty and the Rijskmusuem dataset, yields almost the same performance. However, the ensemble model listed in Table 7 did not improve the results. Considering the comparatively low complexity of the type classification task, satisfactory results can also be obtained with flatter models, e.g. ResNet18 models. Due to the transfer learning approach, good results are usually achieved after only a few training epochs, allowing to train models for type classification even with limited computational resources.

[3] https://pytorch.org/vision/stable/transforms.html#torchvision.transforms.RandomAffine

Table 6: Performance of ResNet18 models trained on the Getty dataset, with a varying proportion of paintings algorithmically converted to drawing-like images.

Dataset	Micro F_1-Score
Getty	0.9256
Getty, 1/3 of paintings converted to drawings	0.9357
Getty, 1/2 of paintings converted to drawings	0.9365

The models were pre-trained on ImageNet; batch norm, conv. 4, conv. 5 and fc. layers were fine-tuned for 30 epochs. Training images were resized and cropped to 224 x 224 pixels and a batch size of 32 was used. An Adam optimizer and a constant learning rate of 10^{-3} were used.

Table 7: Performance of selected models in the type classification task. The models are sorted by the F_1-Score for the drawings class on the Getty test set in descending order.

Model	Validation Set		Test Set	
	F_1-Score for Drawings	F_1-Score for Paintings	F_1-Score for Drawings	F_1-Score for Paintings
DenseNet201	0.9749	0.9959	0.9739	0.9953
ResNet50	0.9644	0.9941	0.9735	0.9953
Ensemble of Resnet18, Resnet34 and Resnet50	0.969	0.9949	0.96	0.9929
ResNet34	0.9592	0.9933	0.9553	0.9922
ResNet18	0.9596	0.9933	0.9391	0.9890

The DenseNet201 was trained on the Getty dataset, the Kaggle datasets, the Metropolitan dataset and the Wikiart dataset. The ResNet50 was trained on the Getty dataset and the Rijksmuseum dataset. The ResNet34 and the ResNet18 were trained on the Brill dataset, the Getty dataset, the Kaggle datasets and the Rijksmuseum dataset. In the ensemble model, the predictions of the ResNet18, the ResNet34 and the ResNet50 were combined by a majority vote.

5.2 Genre Classification

Building upon the results of the binary type classification task, we approach the multi-class genre classification with single model setups of different architectures, as well as ensemble-based approaches.

5.2.1 Single Model Approaches

For the single model approach, we used two different architectures, ResNet50 and DenseNet201. Table 8 compares the results obtained with both architectures. There, we report the highest micro-averaged F_1-score achieved on the Getty validation set for each model.

To determine the baseline performance for the DenseNet201 architecture, we fine-tuned a model on the Getty and Art500K datasets. In this baseline configuration, all layers were fine-tuned and the training set was balanced by random oversampling. In Table 8, this baseline model is compared to a model that was trained with cosine annealing learning rate scheduling [25]. We observe that the model without learning rate scheduling performs slightly better and achieves a 0.3% higher micro F_1-score. We observed only small differences between the FiveCrop and TenCrop test-time image augmentations. FiveCrop performed better in our final model but we do not believe that this observation is generalizable. The overall best model of the DenseNet201 architecture achieves a micro F_1-score of 0.924. For this model, we resized and cropped the images to 400 x 400 pixels and applied random horizontal flipping and random affine transformations for data augmentation.

For the ResNet50 architecture, we use a model as baseline with only the softmax classifier fine-tuned. Compared to the baseline model, a model trained with the Pytorch "reduce learning rate on plateau" scheduler[4] achieves a 0.3% higher micro F_1-score. Using the ImbalancedDatasetSampler[5] we improve the baseline micro F_1-score by 3.4%. For type classification, we observed that unfreezing the last convolutional layers and the classifier results in further improvements. We therefore evaluate the impact of retraining layers in combination with the previously explained balancing technique for the genre classification. We note that retraining three layers outperforms retraining five layers and results in an overall best run of the ResNet50 with a micro F_1-score of 0.931.

5.2.2 Ensemble model approach

Besides single models, we also experimented with ensemble models in the genre classification task. As described in Subsection 4.3, our ensemble models consist of five one-versus-the-rest ResNet50 classifiers where each classifier is trained to detect one genre class. For each of the five classifiers, we evaluate the impact of data augmentation and learning rate scheduling. For all genres, we use ResNet50 models pre-trained on the ImageNet dataset. For the portrait class, we addition-

[4] https://pytorch.org/docs/stable/optim.html#torch.optim.lr_scheduler.ReduceLROnPlateau

[5] https://github.com/ufoym/imbalanced-dataset-sampler

Table 8: Results of our experiments with single models of DenseNet201 and ResNet50 architecture in the genre classification task.

Experiment	Configuration	Micro F_1-Score
DenseNet201		
Baseline	Baseline configuration[1]	0.924
Learning rate	Cosine annealing scheduling	0.921
ResNet50		
Baseline	Baseline configuration[2]	0.838
Learning rate	Reduce on plateau scheduling	0.841
Balancing	Imbalanced sampler	0.872
Retraining	3 layers retrained[3]	0.928
	5 layers retrained[4]	0.881

[1]The DenseNet201 models were pre-trained on ImageNet; all layers were fine-tuned for 30 epochs using the Art500k dataset and the Getty dataset. The training set was balanced by random oversampling and a batch size of 32 was used. A constant learning rate of 10^{-3} was used in the baseline configuration.

[2]The ResNet50 models were pre-trained on ImageNet; the classifier was fine-tuned for 30 epochs using the Getty dataset. A batch size of 32 was used. A constant learning rate of 10^{-3} was used in the baseline configuration. Both retraining configurations used the ImbalancedDatasetSampler for balancing.

[3] Retraining of conv. 4, conv. 5 and fc.

[4] Retraining of conv. 2, conv. 3, conv. 4, conv. 5 and fc.

Table 9: Performance of ResNet50 models trained to detect one genre each (one-versus-the-rest) using different data augmentation techniques and learning rate schedulers.

	Class					
Experiment	Portrait, ImageNet	Portrait, VGGFace2	Genre	History	Still life	Landscape
Stepwise LR decay	0.474	0.545	0.728	0.899	0.53	0.9
Cosine Ann. Scheduler	0.439	0.390	0.705	0.9	0.545	0.918
Stepwise LR decay with data augmentation	0.428	-	0.572	0.841	0.472	0.762
Cosine ann. scheduler with data augmentation	0.342	0.5	0.592	0.702	0.395	0.734

The models were pre-trained on ImageNet, for the portrait class an additional model pre-trained on VGGFace2 was evaluated. Batch norm, conv. 4, conv. 5 and fc. layers were fine-tuned for 30 epochs on the Art500k, Europeana, Getty and WGA datasets. Training images were resized and cropped to 224 x 224 pixels. The training sets were balanced by random undersampling and a batch size of 115 was used. An Adam optimizer was used. For stepwise learning rate decay, the initial learning rate was set to 10^{-3} and the step size to 4. For cosine annealing learning rate scheduling, the initial learning rate was set to 10^{-4} and η_{min} to 0. The transformations RandomGrayscale, RandomPerspective and RandomHorizontalFlip from the torchvision package were used for data augmentation [1].

[1] https://pytorch.org/vision/stable/transforms.html

Table 10: Performance of two ensemble models for genre classification on the Getty validation set and on the SemArt dataset. Ensemble 1 is a combination of the best individual models from the experiments presented in Table 9. Ensemble 2 was created by exploring various single model combinations.

Genre	Ensemble Model 1		Ensemble Model 2	
	F_1-Score on Getty	F_1-Score on SemArt	F_1-Score on Getty	F_1-Score on SemArt
Genre	0.757	0.5101	0.7646	0.5237
History	0.763	0.8542	0.7467	0.8631
Landscape	0.9154	0.8574	0.9145	0.8658
Portrait	0.8	0.6949	0.8	0.7736
Still life	0.9507	0.7434	0.9507	0.7711

Both Ensemble models consist of one individual ResNet50 classifier per genre category pretrained on ImageNet. Batch norm, conv. 4, conv. 5 and fc. layers of each ResNet50 were fine-tuned for 30 epochs on the Art500k, Europeana, Getty and WGA datasets. For details on the training setting, see Table 9. In both ensemble models, the prediction of the classifier with the highest confidence was used as final prediction.

ally evaluate an InceptionResNet pre-trained on the VGGFace2 dataset [5] for face recognition. Table 9 shows the performance of the best single models for these different experimental settings. For each model, we report the highest F_1-score that was achieved on the Getty validation set. For landscape paintings we observe a baseline F_1-score of 0.9 which is the highest baseline score over all classes. Data augmentation reduces the F_1-score by 14%. This decrease caused by the use of augmentation is observed throughout all single models of the ensemble with varying margins. Fine-tuning with cosine annealing learning rate scheduling improved the F_1-scores of the landscape, history and still life classifiers up to 1% compared to the stepwise learning rate decay. For the classification of the genre paintings class the use of stepwise learning rate decay outperforms the cosine scheduling by 2.3%. A similar effect is noticeable for the InceptionResNet classifier for the portrait class where the the F_1-score drops by 15% when using the cosine annealing learning rate scheduling. Comparing the ResNet50 and the InceptionResNet for classifying the portrait class, we notice that they both perform better without image augmentation but the InceptionResNet shows slightly higher confidence.

When selecting the individual models for the ensemble, we compared a combination of the best individual models (Ensemble 1) with a trial-and-error ensemble (Ensemble 2) whose composition was guided by intuition. Table 10 shows that both ensemble models perform similarly on the Getty validation set. When validated on the SemArt dataset, Ensemble 2 performs slightly better. In particular, Ensemble 2 achieves a higher F_1-score for the history paintings class on the Semart dataset. We

therefore conclude that Ensemble 2 shows increased robustness compared to the best-of-model.

6 Conclusion

In this work, we successfully applied deep learning models to two image classification tasks from the art domain. In the type classification of artworks, our models achieve human-like accuracy. As expected, the genre classification of artworks turned out to be more a complex problem. Building on our results, future work should aim to further improve model performance for this classification task.
Overall, our results demonstrate that the transfer learning approach can produce very good results even on small, highly unbalanced training datasets. By using additional training data and employing random resizing for data augmentation, we were able to further improve the performance of our models. In contrast, most data augmentation methods, over- and undersampling techniques, and adjustments to the loss functions did not yield any benefits. Ensemble learning also did not improve the results compared to the single-model approach. Since most of our experiments rely on a comparatively small number of training runs, future work should strive for additional statistical evidence. In particular, the impact of data augmentation and over- or undersampling techniques likely depends on the training datasets and should therefore be further investigated.

References

[1] S. Baker. "Finding A Way To Make Digitizing Art Collections Profitable". In: *Forbes* (June 1, 2019).

[2] M. Barni, A. Pelagotti, and A. Piva. "Image processing for the analysis and conservation of Paintings: Opportunities and challenges". In: *IEEE Signal Processing Magazine* 22.5 (Sept. 2005), pages 141–144. ISSN: 1558-0792. DOI: 10.1109/MSP.2005.1511835.

[3] P. Branco, L. Torgo, and R. Ribeiro. "A Survey of Predictive Modelling under Imbalanced Distributions". In: *arXiv* 1505.01658 (May 13, 2015).

[4] V. Burke, D. Jørgensen, and F. A. Jørgensen. "Museums at Home: Digital Initiatives in Response to COVID-19". In: *Norsk museumstidsskrift* 6.2 (Dec. 10, 2020), pages 117–123. ISSN: 2464-2525. DOI: 10.18261/issn.2464-2525-2020-02-05.

[5] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman. *VGGFace2: A dataset for recognising faces across pose and age*. May 13, 2018.

[6] E. Cetinic and S. Grgic. "Genre classification of paintings". In: *Proceedings of the International Symposium on Electronics in Marine* (Zadar, Croatia). Edited

by M. Muštra, D. Tralić, and B. Zovko-Cihlar. Zadar, Croatia: ELMAR, 2016, pages 201–204. ISBN: 978-953-184-221-1. DOI: 10.1109/ELMAR.2016.7731786.

[7] E. Cetinica, T. Lipica, and S. Grgic. "Fine-tuning Convolutional Neural Networks for fine art classification". In: *Expert Systems With Applications* 114 (Dec. 30, 2018), pages 107–118. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2018.07.026.

[8] *Collection*. The National Museum of Art, Architecture and Design Oslo. URL: https://www.nasjonalmuseet.no/en/collection (visited on 2021-03-16).

[9] C. Concordia, S. Gradmann, and S. Siebinga. "Not just another portal, not just another digital library: A portrait of Europeana as an application program interface". In: *IFLA Journal* 36.1 (2010), pages 61–69. ISSN: 1745-2651. DOI: 10.1177/0340035209360764.

[10] N. Dalal and B. Triggs. "Histograms of oriented gradients for human detection". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (San Diego, USA). Volume 1. IEEE, 2005, pages 886–893. ISBN: 0-7695-2372-2. DOI: 10.1109/CVPR.2005.177.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. "ImageNet: A large-scale hierarchical image database". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (San Francisco, USA). IEEE, 2009, pages 248–255. ISBN: 978-1-4244-3992-8. DOI: 10.1109/CVPR.2009.5206848.

[12] C. Florea, R. Condorovici, C. Vertan, R. Boia, L. Florea, and R. Vrânceanu. "Pandora: Description of a Painting Database for Art Movement Recognition with Baselines and Perspectives". In: *Proceedings of the 24th European Signal Processing Conference* (Budapest, Hungary). IEEE, 2016, pages 918–922. ISBN: 978-0-9928-6266-4. DOI: 10.1109/EUSIPCO.2016.7760382.

[13] N. Garcia and G. Vogiatzis. "How to Read Paintings: Semantic Art Understanding with Multi-modal Retrieval". In: *Computer Vision – ECCV 2018 Workshops. Part II* (Munich, Germany). Edited by L. Leal-Taixé and S. Roth. Lecture Notes in Computer Science. Springer, Cham, Jan. 29, 2019, pages 676–691. ISBN: 978-3-030-11012-3. DOI: 10.1007/978-3-030-11012-3_52.

[14] *Getty Search Gateway*. The J. Paul Getty Trust. URL: https://search.getty.edu/gateway (visited on 2021-03-16).

[15] K. He, X. Zhang, S. Ren, and J. Sun. "Deep Residual Learning for Image Recognition". In: *arXiv* 1512.03385 (Dec. 10, 2015).

[16] J. M. Hollands. "Web Gallery of Art". In: *Reference Reviews* 15.6 (2001), pages 32–32. ISSN: 0950-4125. DOI: 10.1108/rr.2001.15.6.32.335.

[17] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. "Densely Connected Convolutional Networks". In: *arXiv* 1608.06993 (Jan. 18, 2018).

[18] X. Huang, S.-h. Zhong, and Z. Xiao. "Fine-art painting classification via two-channel deep residual network". In: *Advances in Multimedia Information Processing – PCM 2017* (Harbin, China). Edited by B. Zeng, Q. Huang, A. El-Saddik, H. Li, S. Jiang, and X. Fan. Lecture Notes in Computer Science. Springer, Cham, May 10, 2018, pages 79–88. ISBN: 978-3-319-77383-4. DOI: 10.1007/978-3-319-77383-4_8.

[19] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. "Caffe: Convolutional Architecture for Fast Feature Embedding". In: *Proceedings of the 22nd ACM International Conference on Multimedia* (Orlando, USA). New York, USA: ACM, 2014, pages 675–678. ISBN: 9781450330633. DOI: 10.1145/2647868.2654889.

[20] A. Joshi, A. Agrawal, and S. Nair. "Art Style Classification with Self-Trained Ensemble of AutoEncoding Transformations". In: *arXiv* 2012.03377 (Dec. 6, 2020).

[21] D. P. Kingma and J. Ba. "Adam: A Method for Stochastic Optimization". In: *arXiv* 1412.6980 (Jan. 17, 2017).

[22] 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* (Lake Tahoe, USA). Edited by P. L. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger. Volume 1. Red Hook, USA: Curran Associates Inc., 2012, pages 1097–1105.

[23] A. Lecoutre, B. Negrevergne, and F. Yger. "Recognizing Art Style Automatically in painting with deep learning". In: *Proceedings of the Ninth Asian Conference on Machine Learning*. Edited by M.-L. Zhang and Y.-K. Noh. Proceedings of Machine Learning Research. PMLR, Nov. 2017, pages 327–342.

[24] E. Leventaki. "Art Professionals and the Frenzy of Digitisation". In: *ICOM Voices* (July 8, 2020).

[25] I. Loshchilov and F. Hutter. "SGDR: Stochastic Gradient Descent with Warm Restarts". In: *arXiv* 1608.03983 (May 3, 2017).

[26] D. G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". In: *International Journal of Computer Vision* 60.2 (Nov. 2004), pages 91–110. ISSN: 1573-1405. DOI: 10.1023/B:VISI.0000029664.99615.94.

[27] H. Mao, M. Cheung, and J. She. "DeepArt: Learning Joint Representations of Visual Arts". In: *Proceedings of the 25th ACM international conference on Multimedia* (Mountain View, USA). New York, USA: ACM, Oct. 2017, pages 1183–1191. ISBN: 9781450349062. DOI: 10.1145/3123266.3123405.

[28] S. Marcel and Y. Rodriguez. "Torchvision the Machine-Vision Package of Torch". In: *Proceedings of the 18th ACM International Conference on Multimedia* (Firenze, Italy). New York, USA: Association for Computing Machinery, 2010, pages 1485–1488. ISBN: 978-1-60558-933-6. DOI: 10.1145/1873951.1874254.

[29] T. Mensink and J. van Gemert. "The Rijksmuseum Challenge: Museum-Centered Visual Recognition". In: *Proceedings of the International Conference on Multimedia Retrieval* (Glasgow, UK). New York, USA: ACM, 2014, pages 451–454. ISBN: 9781450327824. DOI: 10.1145/2578726.2578791.

[30] M. R. Mohammadi and F. Rustaee. "Hierarchical classification of fine-art paintings using deep neural networks". In: *Iran Journal of Computer Science* 4.1 (Sept. 30, 2020), pages 59–66. ISSN: 2520-8446. DOI: 10.1007/s42044-020-00072-0.

[31] T. Navarrete. "Digitization in Museums". In: *Teaching in Cultural Economics*. Edward Elgar, 2020, pages 204–213. ISBN: 978 1 78897 073 0.

[32] E. Oomen. "Classification of painting style with transfer learning". Master's thesis. Tilburg: Tilburg University, July 30, 2018.

[33] *Paintings*. The National Gallery London. 2021. URL: https://www.nationalgallery.org.uk/paintings (visited on 2021-03-16).

[34] E. Posthumus. *Brill Iconclass AI Test Set*. Feb. 21, 2020. URL: https://labs.brill.com/ictestset (visited on 2021-03-17).

[35] T. Rieger. *Technical Guidelines for Digitizing Cultural Heritage Materials: Creation of Raster Image Files*. Federal Agencies Digitization Guidelines Initiative, Aug. 2016.

[36] *Rijksstudio*. Rijksmuseum Amsterdam. URL: https://www.rijksmuseum.nl/en/rijksstudio (visited on 2021-03-16).

[37] M. Sabatelli, M. Kestemont, W. Daelemans, and P. Geurts. "Deep Transfer Learning for Art Classification Problems". In: *Computer Vision – ECCV 2018 Workshops. Part II* (Munich, Germany). Edited by L. Leal-Taixé and S. Roth. Lecture Notes in Computer Science. Springer, Cham, Jan. 29, 2019, pages 631–646. ISBN: 978-3-030-11012-3. DOI: 10.1007/978-3-030-11012-3_48.

[38] B. Saleh and A. Elgammal. "Large-scale classification of fine-art paintings: Learning the right metric on the right feature". In: *International Journal for Digital Art History* 2 (Oct. 2016), pages 70–93. ISSN: 2363-5401. DOI: 10.11588/dah.2016.2.23376.

[39] C. Sandoval, E. Pirogova, and M. Lech. "Two-stage deep learning approach to the classification of fine-art paintings". In: *IEEE Access* 7 (Mar. 28, 2019), pages 41770–41781. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2907986.

[40] L. Tallon. *Scaling the Mission: The Met Collection API*. The Metropolitan Museum of Art. Oct. 25, 2018. URL: https://www.metmuseum.org/blogs/now-at-the-met/2018/met-collection-api (visited on 2021-03-20).

[41] W. R. Tan, C. S. Chan, H. E. Aguirre, and K. Tanaka. "Ceci n'est pas une pipe: A Deep Convolutional Network for Fine-art Paintings Classification". In: *Proceedings of the IEEE International Conference on Image Processing* (Phoenix, USA). IEEE, 2016, pages 3703–3707. ISBN: 978-1-4673-9961-6. DOI: 10.1109/ICIP.2016.7533051.

[42] M. Terras. "Opening Access to collections: the making and using of open digitised cultural content". In: *Online Information Review* 39.5 (Sept. 2015), pages 733–752. ISSN: 1468-4527. DOI: 10.1108/OIR-06-2015-0193.

[43] *The National Museum in 2018*. The National Museum of Art, Architecture and Design Oslo. 2018. URL: https://www.nasjonalmuseet.no/contentassets/98adac84980c4555ae99de8a5ed00e80/rsmelding2018_eng.pdf (visited on 2021-03-17).

[44] L. Volkers. "Image First: Opening Up the Rijksmuseum With Rijksstudio". In: *The Digital in Cultural Spaces* (Singapore). Edited by T. Karthigesu, T. C. Hua, and C.-A. L. M. Gek. Singapore: Culture Academy Singapore, 2017, pages 15–22. ISBN: 978-981-11-5764-6.

[45] X. Wang, D. Kihara, J. Luo, and G.-J. Qi. "Enaet: Self-trained ensemble autoencoding transformations for semi-supervised learning". In: *arXiv* 1911.09265 (Feb. 1, 2021).

[46] M. D. Zeiler. "ADADELTA: An Adaptive Learning Rate Method". In: *arXiv* 1212.5701 (Dec. 22, 2012).

[47] W. Zhao, D. Zhou, X. Qiu, and W. Jiang. "Compare the performance of the models in art classification". In: *PLOS One* 16.3 (Mar. 12, 2021), pages 1–16. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0248414.

[48] J. Zujovic, L. Gandy, S. Friedman, B. Pardo, and T. N. Pappas. "Classifying Paintings by Artistic Genre: An Analysis of Features & Classifiers". In: *Proceedings of the IEEE International Workshop on Multimedia Signal Processing* (Rio de Janeiro). IEEE, Nov. 2009, pages 1–5. ISBN: 978-1-4244-4463-2. DOI: 10.1109/MMSP.2009.5293271.