UNIVERSITAT POLITÈCNICA DE CATALUNYA
UNIVERSITAT DE BARCELONA
UNIVERSITAT ROVIRA I VIRGILI

FACULTAT D'INFORMÀTICA DE BARCELONA

MASTER IN ARTIFICIAL INTELLIGENCE

Using unaltered, variable sized and high-resolution images for training fully-convolutional networks

Day of defense: July 1, 2019

Author:

Olga Fourkioti

Supervisor:

Dario Garcia Gasulla

June 24, 2019

Preface

The conventional deep learning approaches make use of a fixed input image size, which is usually a result of cropping or resizing. However, in the real word objects tend to exhibit a considerable variability in scale, illumination, viewpoint. Particularly, using high resolution images is necessary in a broad range of applications where attention to detail is required, such as medical imaging, astronomy and video surveillance. Despite their usefulness and the constantly increasing computational power in contemporary computing systems, training a convolutions neural network with unaltered high-resolution images has not been sufficiently explored. In this thesis, we investigate using unaltered, variable sized and high-resolution images for training fully-convolutional networks.

List of Figures

List of Tables

Contents

1 Introduction

In recent years, the accessibility to large scale training data and the appearance of deep Convolutional Neural Networks (CNNs) has enabled a rapid revolutionary change in the computer vision community. CNNs are at the core of most state-of-the-art computer vision solutions offering performance gains in a broad range of applications such as object detection (Ren et al. 2017, Cai et al. 2016) , image classification (Karpathy et al. 2014), image segmentation (Ronneberger 2017; Shelhamer, Long, and Darrell 2017 and many other recognition tasks (Lawrence et al. 1997, Simonyan and Zisserman 2015). So far, in most deep-learning based approaches CNN architectures require a fixed input image size (Simonyan and Zisserman 2015,Krizhevsky, Sutskever, and Hinton 2017). However, this paradigm does not constitute a convincing representation of the reality. In a realistic setting, objects tend to exhibit a considerable variability in scales, illumination, viewpoints, etc. In this thesis, our interest is focused on investigating training of CNNs using variable-sized high resolution images. The motivation behind this idea is based on the following key observations that pose novel challenges:

- memory constraints render training CNNs with high resolution images on large scale prohibitively consuming. Coming up with a way to allocate and manage memory appropriately to overcome these severe memory limitations is increasingly important, particularly since the increase in computational power has made memory management the only limiting factor
- the conventional pre-processing step before feeding the data to a CNN when applied to images of arbitrary size is to fit the input image either using cropping, resizing or padding. However, the cropped image may not contain the entire object leading to significant loss of information, while resizing can result in geometric distortion. In this context we hypothesize that by using full image representations, the expressivity of the representation can be increased and the performance of the model enhanced.
- the conventional approach of a sequence of dense layers that are positioned at the end of the CNN's computation sequence is not applicable in this case unless

there are changes introduced in the architecture of the model so that images of arbitrary size can be appropriately processed.

1.0.1 Motivation

One of the most prominent scientific fields where working with high resolution images is desirable is that of medical imaging. Medical images provide anatomical information of the structure of the human body and they also uncover physiological information about the function of human organs or tissues. Nowadays, medical imaging constitutes a very useful tool a physician utilizes for diagnostic purposes. High resolution is of high importance in such a task where even small details can unveil significant information determinant to the diagnosis of a disease. This justifies the reason why there have been many attempts to leverage low-resolution imaging-detectors to generate high resolution imagery (S. Goyal et al. 2018, Nandi et al. 2019).

Besides medical imaging, high resolution can be very useful in astronomy where attention to detail is paramount. In recent years, the advances in astrophotography which is the photography of celestial events and astronomical objects has revolutionized the field of astronomical research because of the detailed astronomical representations it can provide. One of the key characteristics of astrophotographic images is their high resolution as they are able to capture details of the sky invincible to the human eye such as stars, nebulae, and galaxies.

Images with high resolution are also desirable in video surveillance applications. High quality footage provides higher image clarity that is needed to accurately reflect what a scene looks like and identify the objects and people in the background.

Therefore, using high resolution images is very important in a broad range of applications, where attention to detail and high quality footage is required. With the rise of deep learning methods that have enjoyed great success in an abundance of computer vision applications, the need of training CNNs with high resolution images has become imperative.

1.0.2 Thesis Layout

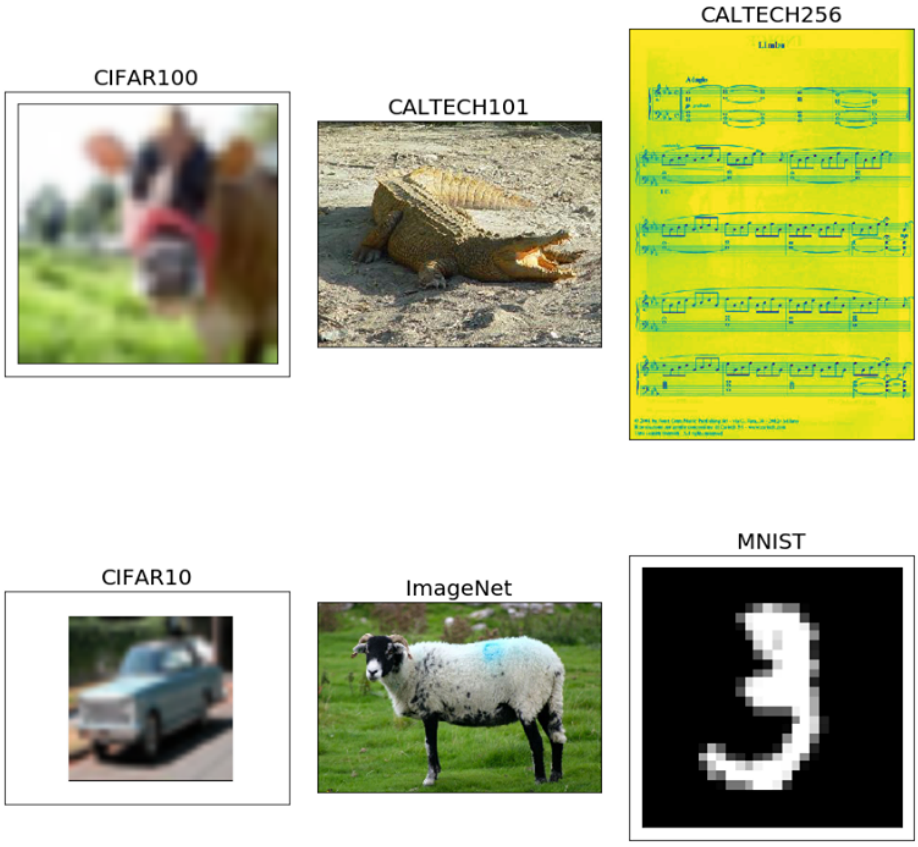
The rest of the thesis is organized as follows. Section Related Work gives an overview of previous related work. Section Baseline Experiments describes our baseline experiments. Section Methodology introduces our proposed methodology. Section Results describes our experimental setup and results. Finally, conclusions and directions for further research are summarized in Section Conclusions.

2 Dataset

Several datasets have emerged as standards for the vision community, the most common of which are described in this paragraph. Some basic properties of the datasets introduced are listed in Table 1. MNIST dataset and its variations usually utilized for benchmarking purposes comprise of 60,000 28x28 grayscale images distributed equally among 10 classes. The application scenario in this case is the recognition of hand-written digits. CIFAR RGB image datasets contain thousands of 32 x 32 color images belonging in various object classes. CALTECH RGB datasets consist of color images of variable size that as in CIFAR data set also correspond to a variety of object categories. Finally, ImageNet (Deng et al. 2009) is a large-scale variable-sized image database organized according to the WordNet hierarchy. It aims to populate the majority of the 80,000 synsets of WordNet (mainly nouns) with an average of 500-1,000 clean and full resolution images, resulting in a database of more that 14 million images. However, the most popular releases of this dataset, such as LSCVRC 2012 are limited to 1,000 different classes. In Figure 1, we can see some sample figures from each dataset.

The common characteristic shared among those datasets is that they are not exclusively constructed to facilitate studies on training high resolution images. Even ImageNet that offers a wide and diverse coverage of the image world with images of different scales, sizes ,view points has an average image resolution of 400 x 350. The difficulty remains therefore, in generating a sufficiently accurately annotated data set of images that can satisfy the specifications of our task. It is also notable to mention that the most common preprocessing strategy used with these datasets is to crop or resize the input images (Krizhevsky, Sutskever, and Hinton 2017, Çalik and Demirci 2018).

Figure 1. Sample figures from common computer vision datasets



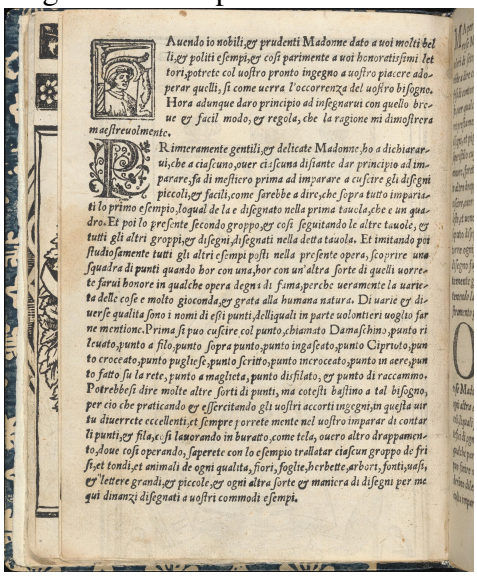
The Metropolitan Museum of New York (MET) has recently released over 200,000 images of public domain art-work, that includes different types of artworks such as paintings, drawing, furniture, sculptures, photographs, illustrations, etc. The sizes of the images cover a broad spectrum of sizes and thus can be used for the purposes of our task. In Figure 2, an example of two images from the MET data set with different image and aspect ratio is provided.

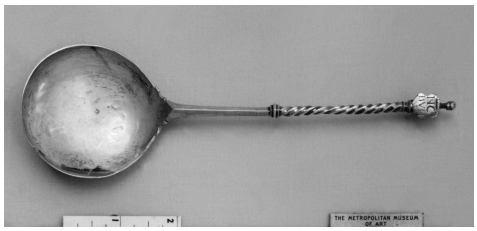
Besides that, MET data set can be also useful for addressing art classification tasks. Re-

Table 1. Common computer vision datasets

Computer Vision Data sets for Classification Tasks		
Name	Description/Number of classes/Instances per class/Aspect ratio	Train/Test Images
CALTECH 101	101 classes of objects with 40–800 images per class with dimensions roughly 300x200 pixels	9,146 images
CALTECH 256	256 classes of objects with 80–827 images per class of variable sizes	30,607 images
CIFAR-100	100 classes of objects with fixed dimension (32x32) containing 600 images each	Train: 50,000 images, Test: 10,000 images
CIFAR-10	10 classes of objects with fixed dimension (32x32) containing 6000 images each	Train: 50,000 images, Test: 10,000 images
IMAGENET	image database of variable sizes organized according to the WordNethierarchy	>14M images
MNIST	10 classes of hand-written digits with dimension 28x28 pixels	Train: 60,000 images, Test: 10,000 images

Figure 2. Example from MET data set of two images with different sizes and aspect ratios





cently, the impressive growth of internet technologies led to a significant increase of online fine art collections. The availability of those data sets enabled the scientific community to explore, retrieve and archive artworks. Specifically, the task of fine art painting classification by artist, genre or style has been addressed several times and the MET data set offers a plethora of information that can be of interest in art related projects. Specifically, the collection is accompanied by some particular meta data in the form of annotations done by human

experts. These annotations integrate a wide range of additional information about each artifact exhibited in the museum, such as the artist's name, the culture the artifact belongs to, the medium used to construct it, the begin date of its construction as well as the date that it was released to the public, etc. The meta data available comprise in total 43 attributes, the most important of which are listed in together with the percent of missing values, as well as the number of classes per attribute in Table 2.

A small description of the attributes presented in Table 2 follows. The name of the object and a small description of the artwork exhibited can be found in the fields *Object Name* and *Title*. *Object Name* refers to a category of objects which share the same attributes and it is not unique for every artwork. The Classification field has the same properties as the *Object Name*, but refers to broader categories of objects. Object type refers to an evem broader category of objects. The field Culture is more general and includes information about the culture that the artifact can be categorized into, while the *Medium* and *Dimensions* fields refer to the medium used to manufacture the artifact and the dimensions of it, respectively. The Department is the department in the museum in which the artifact is located which is also indicative of the culture it belongs to (Egyptian Art, the department of Islamic Art etc).

Out of the 43 fields, 6 fields are related to the historical period of the artifact, such as the field period which refers to the chronological period in which an artifact belongs to. The field *Dynasty* is associated explicitly with Egyptian art and refers to the pharaonic dynasties which coincide with the artifact's creation. Similar with the field *Dynasty* but not regional specific is the field *Reign* which offers additional information of the artifact's chronology and location based on the rulers that governed during the period and near the region where the artifact was created/discovered. The day of creation of an artwork, the date that it was completed as well as the time interval between the creation and its completion are described in the fields *Object End Date*, *Object Start date* and *Date*.

There are also some fields comprising information regarding the artist of an artwork. The *Artist Display Name*, *Artist Display Role*, *Artist Nationality*, *Artist Begin Date* and *Artist End Date* make reference to the name, the profession, the nationality, the birth date and death date of an artist, respectively. In some cases, there is also a brief biography of the artist's life, focusing mostly on the date and region where he or she was born and died.

Table 2. Attributes of MET dataset

Attribute	Number of classes	Percentage of missing values%	Number of Instances
Artist Alpha Sort	22,551	56.25	93,628
Artist Begin Date	8,504	61.61	82,144
Artist Display Bio	18,480	60.28	84,989
Artist Display Name	22,555	56.24	93,640
Artist End Date	8,685	61.39	82,615
Artist Nationality	1,157	66.70	71,251
Artist Role	1,976	56.62	92,832
Artist Prefix	2,258	87.32	27,116
Artist Suffix	691	98.24	3,747
City	1446	92.61	15,812
Classification	881	9.09	194,557
County	735	97.77	4,761
Country	635	78.90	45,143
Credit Line	18,614	0.022	213,963
Culture	4,661	41,89	124,347
Dimensions	142,268	7.62	197,695
Dynasty	286	95.08	10,523
Excavation	282	96.11	8,306
Geography Type	93	82.33	37,810
Locale	612	96.16	8,201
Locus	988	97.71	4,884
Medium	37,128	0.48	212,974
Title	93,406	12.06	188,198
Period	1,554	75.85	51,678
Portofolio	1,295	97.11	6,182
River	56	99.80	422
Reign	299	97.45	5,445
Region	377	91.14	18,954
State	93	99.10	1,914
Subregion	310	93.97	128,89
Object Begin Date	1,887	0.00	214,011
Object Date	18,906	5.09	203,103
Object End Date	1,860	0.00	214,011
State	93	99.10	

Table 3. Snuff bottle

Snuff bottle	Properties
	• Period: Qing dynasty • Date:19th century • Culture:China • Medium: Painted enamel • Dimensions: H 7.9 cm • Classification: Snuff Bottles • Department: Asian Art • Object Type: Vessels • Geographic Location: Asia • Date / Era: A.D. 1800–1900

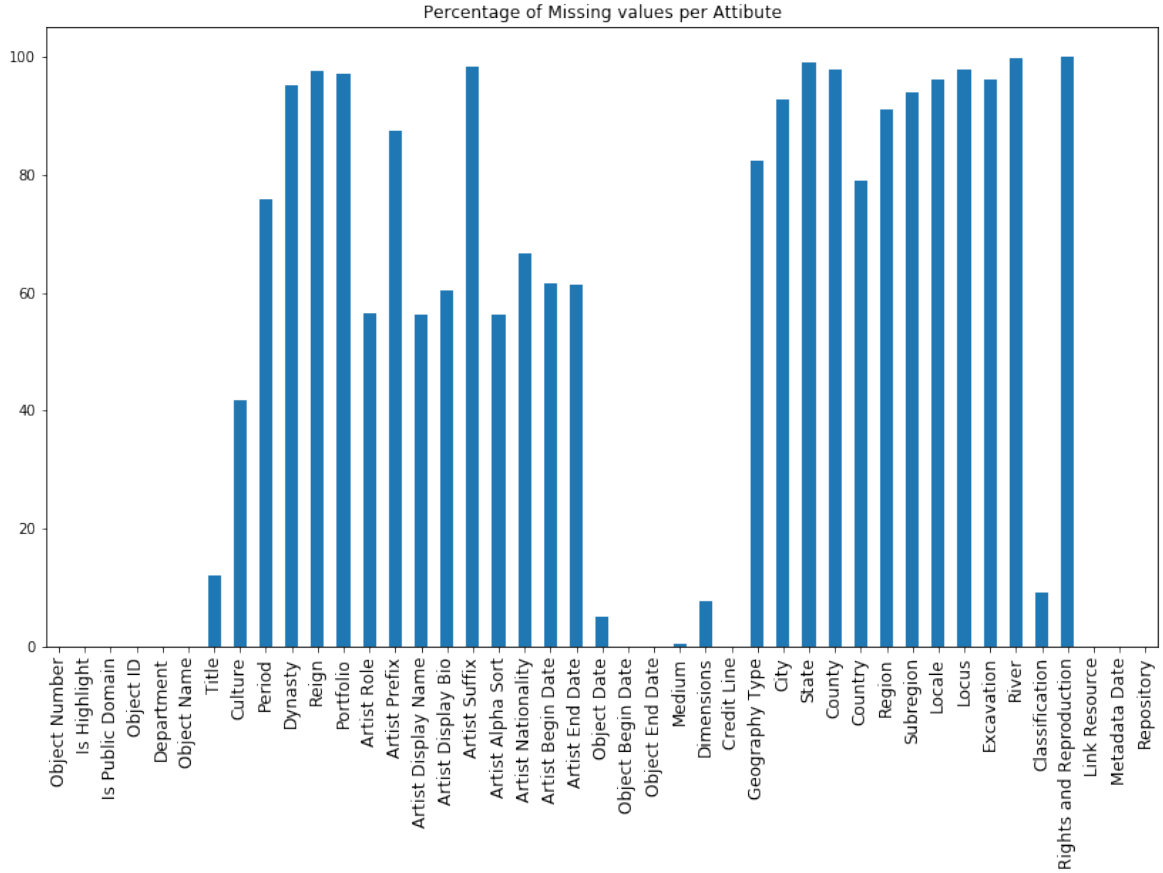
The full name of the artist and the name of the artist's collection in which the artwork was exhibited correspond to the fields Artist Aplha Sort and *Portofolio*. Even though quite rare, the field *Artist Suffix* offers additional information about the nationality of the artist or the names of the other artists in case of a collaboration.

The geographical information about the location where an object was discovered, made or excavated are also included in the meta data. *Geography type* attribute determines the country of origin of the artifact or the country in which it is attributed to. The geographical fields range from the more general ones like the *Country* or *Region* of origin of the object to more specific ones like the *City*, *County* or *Subregion*. The *locale* and the *Locus* give even more specific geographical information for ancient objects that were discovered mostly during excavations. But these attributes are present in a handful of artworks, mostly in Egyptian Art. The company responsible for an excavation and the year that it took place are also included in the *Excavation field*. For a better understanding of the attributes describing an object in the MET collection, a representative example of an artwork and its available metadata can be seen in Table 3, where a snuff bottle is depicted.

The vast majority of the attributes described cannot formulate a concrete classification task due to the intense presence of missing values. Figure 3 indicates that more than half of

the meta data attributes consist of more than 65% of missing values. Thus, they do not provide us with sufficient data to conduct a classification task. Our interest is mostly focused on attributes that their amount of missing values and the number of images per class still provides us with a sufficient number of images to perform a classification task using deep CNNs.

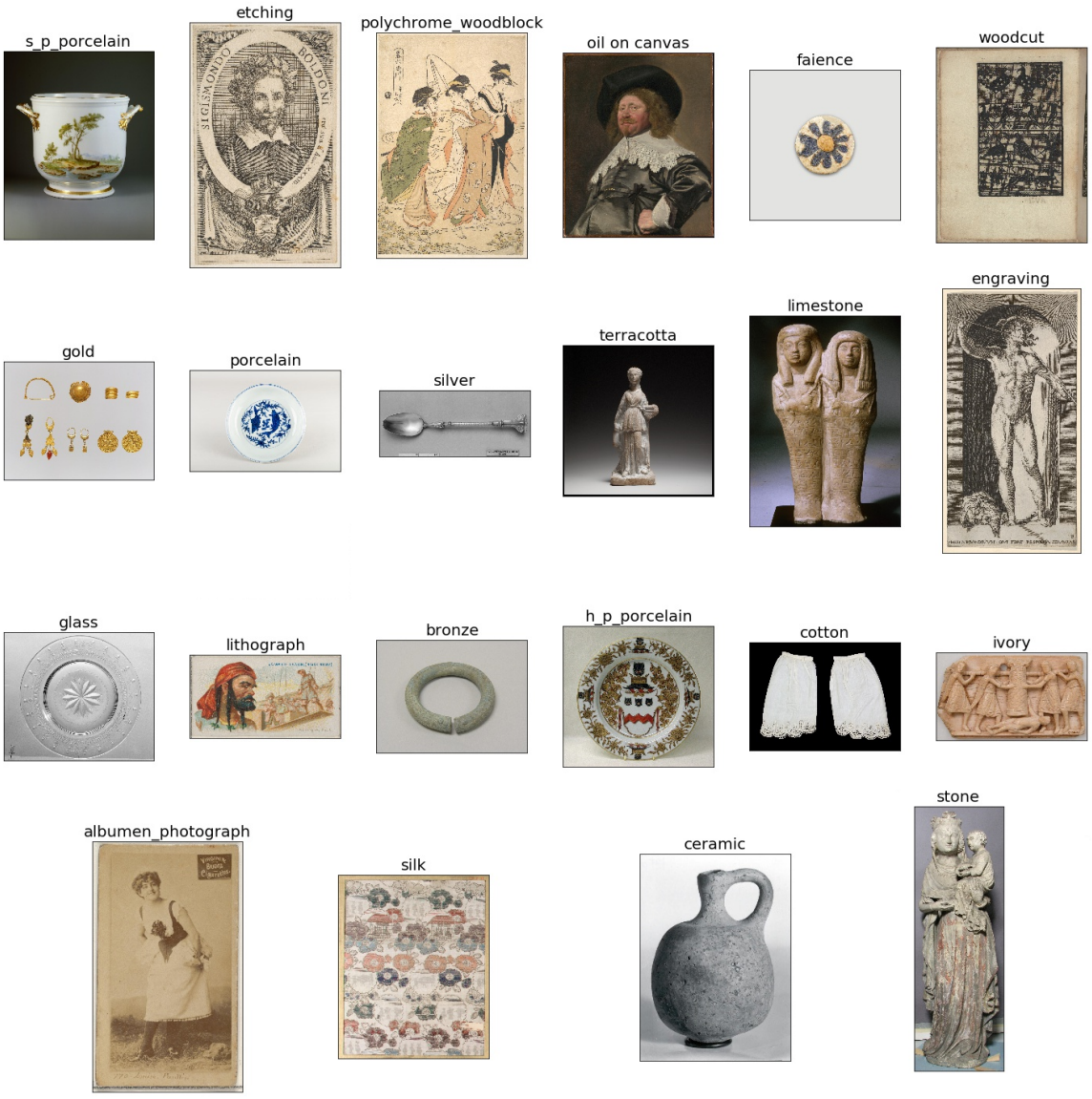
Figure 3. Missing values per attribute in the MET data set



Based on the number and distribution of images in the data set, as well as the number of instances per classes used in previous image classification works (Cetinic and Grgic 2016; Tan et al. 2016), one classification task was defined to be be performed on the MET data set, that of classifying the medium of an artifact. The goal of this classification task is the recognition of the different materials from which the artworks are constructed, such as silver, gold, etc.

By taking into account the number of instances per class present in medium attribute, we

Figure 4. Sample figures of the 23 classes present in the data set

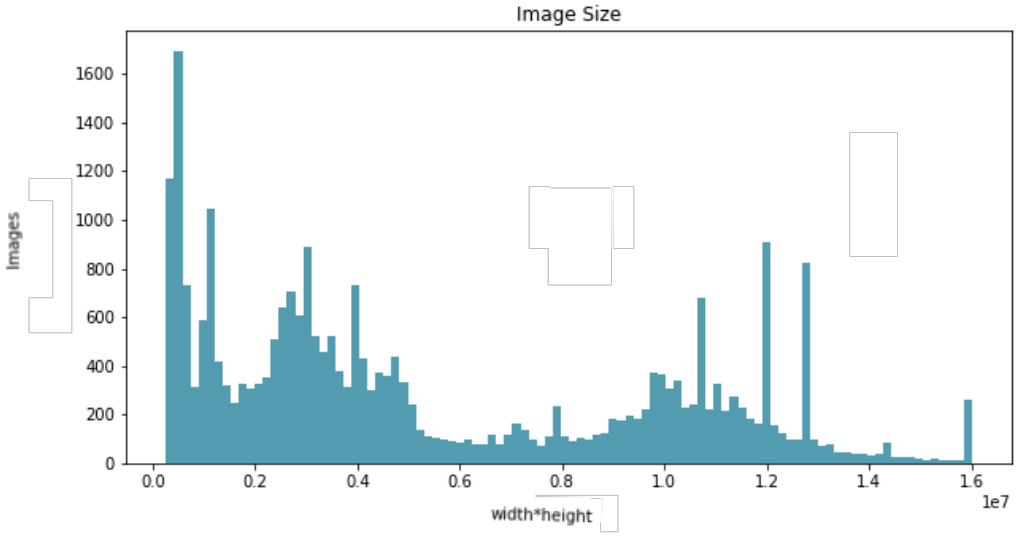


defined the subset of classes for the task. We used a subset of 23 classes. Figure samples from each class can be seen in Figure 4. In order to reduce the complexity of the problem and also to overcome problems in the training introduced by class imbalance each class was represented with precisely 1,200 images. An additional constraint was imposed on the size of the images included in the data set. Images with width or height less that 500 pixels were considered too small and were excluded from the collection. Also, an upper threshold to the image size was also applied and images larger 4Kx4K were removed. After applying those

two criteria, we ended up with a collection of 27,600 images distributed evenly among 23 classes. 23,000 images constitute the training set,2,300 images the validation set and the remaining 2,300 the test set.

The two key properties of this subset that make it applicable for this task are the variability of image size and aspect ratio whose distributions are illustrated in Figure 5 and 7. In Figure 6, we can see the scatter plot of the dimensions of the images in the data set. The densely populated rectangle covering a rather large area where the majority of the data points lie is easily recognizable, highlighting the fairly large variability of the image sixes of the collection. The distribution of the image sizes provides a more rigorous way to visualize the distribution of the size of the images because we can observe the whole range of sizes that extend from 250,000 to 16,000,000 pixels. We can also easily observe some dimentions that are over-represented in comparison to the rest. A representative example is that of images with height or width 4K, which create the boundaries of the upper square in the Figure 6. An interior square of datapoints is also visible in that plot and its boundaries are defined by the 2K x 2K images. The high variability can be observed in the distribution of the aspect ratios of the images. In 7, it is easy to see that the aspect ratios are not centered tightly around a mean value but they exhibit a considerable variability ranging from 0.5 to 2.

Figure 5. Size distribution of MET subset



(1).png

Figure 6. Scatterplot of the dimensions of the MET subset

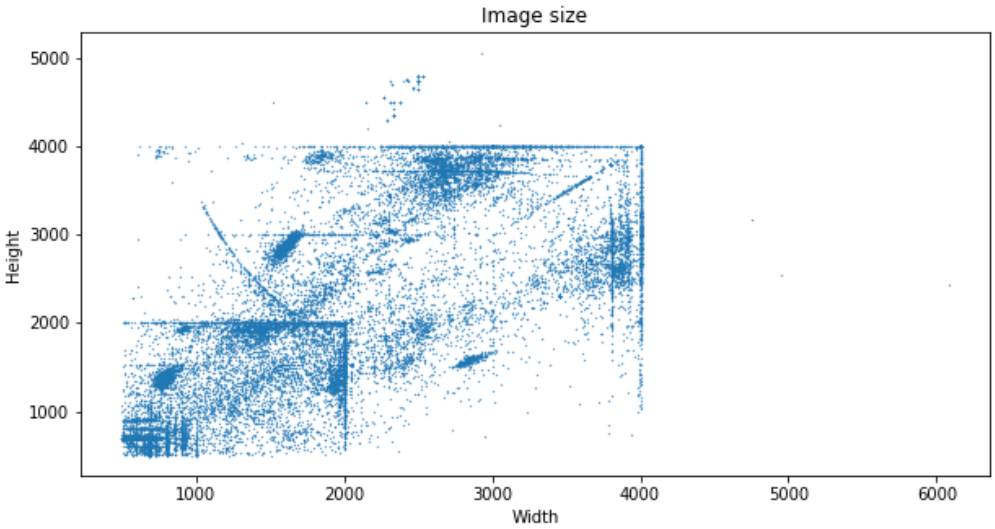
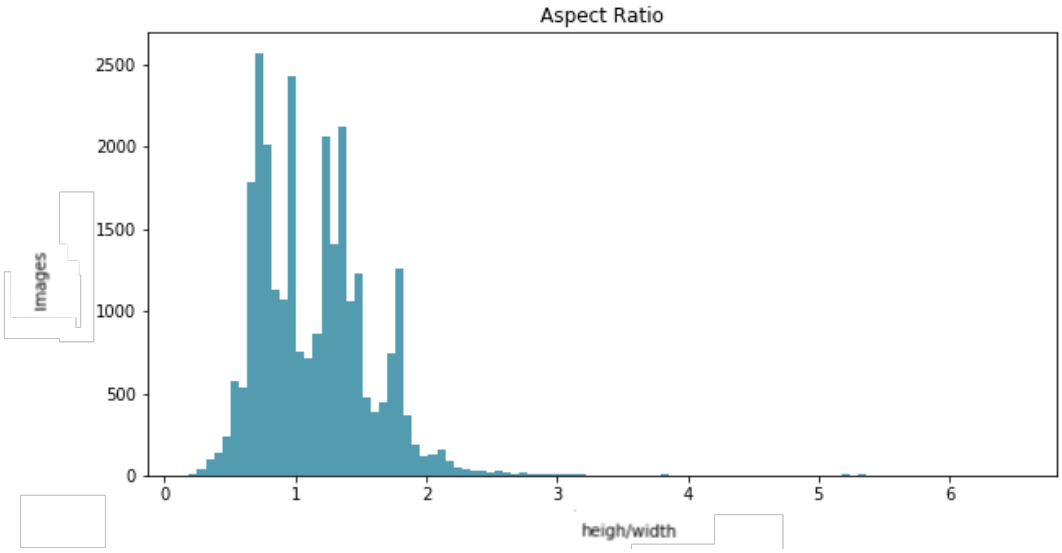


Figure 7. Aspect ratio distribution of MET subset



3 Related Work

Training variable-sized high resolution images raises two different points of interest: the fist one pertaining to the architectural changes of the model to accept arbitrary sized images and the second one to the efficient memory management of the input data. In the sections to follow we cover state-of-the-art methodologies on dealing with both those complexities.

3.1 Variable Sized images

As it was outlined in Introduction, CNN architectures are usually composed of cascaded stages of alternating stacked convolutional and pooling layers followed by fully connected layers at the end of the network. The convolutional layers act upon input feature maps in a sliding window manner to generate convolved feature maps and the pooling layers aggregate this information by applying either an average or max pooling operation. The feature maps of the last convolutional layer are then vectorized and fed into the fully connected layers. The sequence of convolutional and pooling layers can handle images of arbitrary sizes since convolution filtes can be sized a variable number of times dependent on the input. What imposes the requirement of fixed input size are the fully connected layers that demand a fixed-sized vectorized input. There have been some approaches to tackle the problem of varying length feature vector representations.

Lin, Chen, and Yan 2014 instead of adopting the traditional fully connected layers introduced global average pooling. Global average pooling operation is equivalent to the operation of calculating the mean not of a given neighborhood window as in the case of average or max pooling but of the entire feature map. In their work, global average pooling was used to entirely replace the fully connected layers of the network by generating one feature map for each corresponding category of the classification task. The authors claim that the global pyramid pooling (GOP) layer can be considered as a structural regularizer that natively prevents overfitting because of the lack of parameters to optimize in that layer. Furthermore, it ensures the coherence of the overall network because it does not bridge the convolutional structure of the network making it more easily interpretable as it reinforces the correspon-

dence between feature maps and categories.

Global average pooling can be considered as a special case of spatial pyramid pooling (SPP) that was introduced later on by He et al. 2015. Apart from behaving as a regularizer, SPP generates fixed length vector representations by applying successive pooling operations in arbitrary sub-regions of the convolved feature-maps. Experiments conducted in a variety of datasets demonstrated that ability of the SPP-net to elevate the recognition accuracy of some CNN architectures such as the ZF-5, Convnet*-5 and Overfeat-5/7. The authors also were the first to test their method on a single network with input images of variable sizes by implementing multi-size training. In multi-size training, two different scales were successively used to train (one scale for the first epoch, different scale for the next one) the network and during the inference phase one scale was used. Results showed that multi-size training improves accuracy when compared to its single-size training counterpart. Even though the authors experimented with multi-size training, the experiments were limited in just two different scales (180,224) and not to cover a wider range of sizes.

A successful application of SPP layer was presented by Han et al. 2017 who utilized the SPP layer for high spatial resolution remote sensing images. The architecture proposed was a variant of ALexNet architecture modified to incorporate the scale-spatial pooling as well as a side supervision processing framework. Additional experimentation was held to investigate the effect of the SPP layer. This was accomplished by keeping all the parameters generated by the pre-trained AlexNet unchanged, while the number of SPP layer was varied over the range of one to four. Results in all the data sets concluded that the best obtained classification performance was achieved when the spatial pyramid layer is equal to 4. However, all the experiments made use of fixed sized input representations.

In line with the work on SPP layer, Gupta et al. 2018 introduced a dynamic support vector machine (SVM) kernel to address the issues of variable image size in convolutional neural networks. The . The DSPMK kernel is computed using a similarity score between two different images. n particular, two images of different size are given as an input to a convolutional layer in such a way that we get a set of deep activation maps after a series of convolutional and max-pooling layers. These deep activation maps are spatially divided into sub-blocks to form a spatial pyramid. In each level of the pyramid, the activation maps are partitioned to

blocks and the values of the cells of each block are summed or max pooled to form a fixed length representation. The vectors are normalised using either L1 or L2 normalisation and are used to compute a matching score at each level. The DSPMK kernel is computed as a weighted sum of the matching scores at the different levels of the spatial pyramid. Combining the power of varying size CNN-based set of deep feature maps with dynamic kernel yielded state-of-the-art results for several visual recognition tasks such as scene classification on standard datasets like MIT67 and SUN397, but the authors avoided to compare their method to its CNN counterpart: a CNN with a SPP layer.

In the same direction, multi-scale order-less pooling Gong et al. 2014 (MOP-CNN) extracts the CNN's last layer activations when processing different local patches of the image at different scales, performs order-less VLAD pooling of these activations at each level separately, and concatenates the result. This approach achieved state-of-the-art results on scene classification data sets and competitive results on other classification and retrieval tasks.

3.2 Efficient Memory Management

Apart from handling variable sized images, training high resolution images requires substantial computational and memory throughput and introduces non-trivial challenges on how the memory can be efficiently allocated. In that direction, there are a lot of works addressing memory optimization issues.

Chen et al. 2016 propose a systematic algorithm that decreases significantly the memory consumption of a deep neural network model. This is done primarily by memory sharing. Some nodes that store intermediate results can be dropped and recomputed from an extra forward computation when needed. Additionally, the algorithm traverses the graph in topological order and applies in place operations between nodes that are not characterized by overlapping lifespan. The algorithm designed requires sub linear memory cost memory to train a n layer network with a $\mathcal{O}(\sqrt{n})$ memory cost, including only the computational cost of an extra forward pass per mini-batch. Experiments on a 1,000-layer deep residual network demonstrated a reduction in memory from 48G to 7G on ImageNet data set.

Rhu et al. 2016 propose a run time memory manager that virtualizes memory usage of deep

neural networks across both GPUs and CPUs. Despite the fact that at any given point in time a GPU accommodates only a small subset of the intermediate feature maps related to a single layer computation, most ML frameworks adopt a network-wide allocation policy requiring the intermediate feature maps of the layers to be stored in the GPU. The proposed memory manager is responsible for offloading the intermediate feature maps to CPU when they are not immediately required and prefetching them to GPU when hey are directly or indirectly -because of data dependency relations- involved in a computation. By exploiting this memory-release and reuse patterns, the virtualized DNN reduced significantly memory usage in a series of know architectures. The most outstanding example was that of vgg-16 which was trained using VDNN with a batch size of 256 that requires 28GB of memory on a single NVIDIA Titan X GPU card containing 12 GB of memory.

4 Baseline Experiments

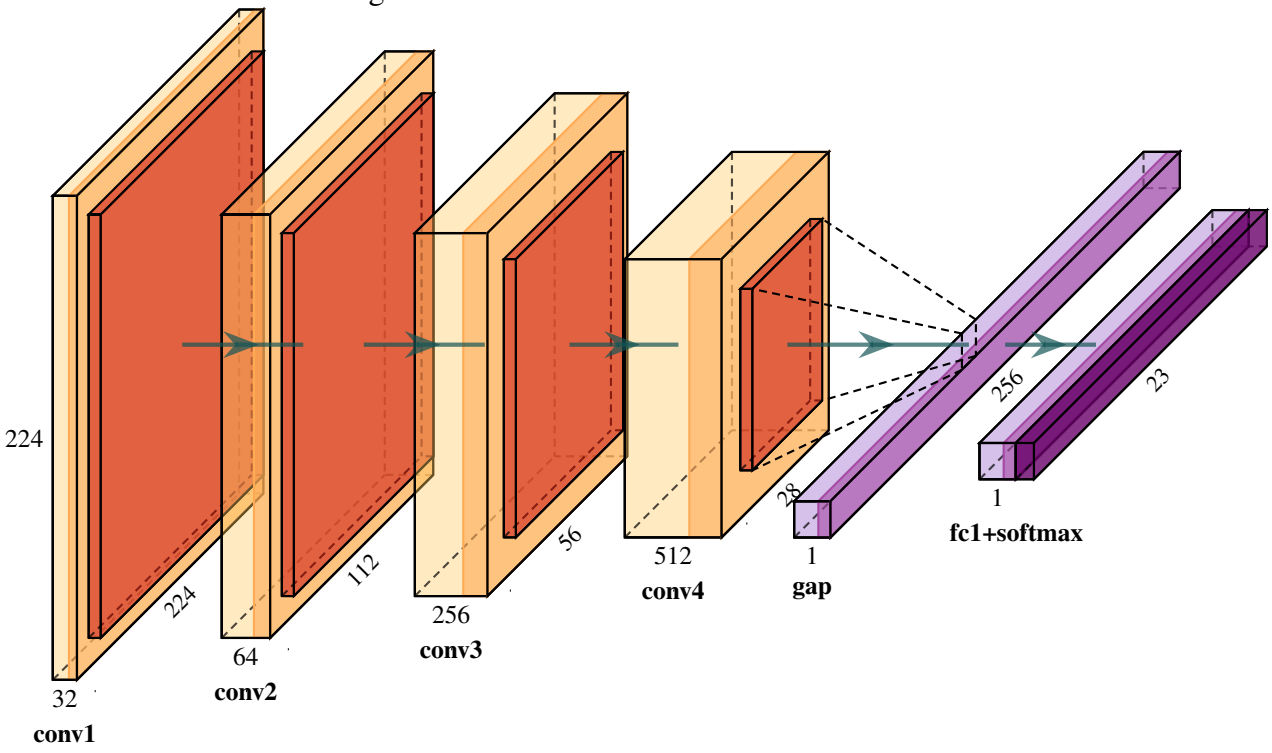
In order to assess the performance of our model and draw conclusions about whether the full image representations and B2B methodology can facilitate the generalization ability of the model, we define as our baseline experiment the conventional approach of feeding to a CNN resized cropped images of size 224x224.

4.1 Architecture

In order to enable the comparison between the baseline and our experiments exactly the same architecture is used in both scenarios. Because of the need to handle the arbitrary sized images present in our second experiment, a fully convolutional architecture was implemented. Additionally, the high resolution of the images and their consequent memory limitations does not allow us to use a very sophisticated CNN architecture. Thus, our goal is to build a simple model with a small number of layers and filters that can fulfill the memory requirements of the high resolution images. After calculating the amount of memory needed to train the most memory consuming images in our variable sized data set we came up with the following simple architecture:

The CNN is composed of a sequence of eight alternating convolutional and max pooling layers. The max pooling layers operate on all the convolved feature maps within a given neighborhood window of 4 pixels with a stride of 2. Additionally, the RELU non-linearity is applied to all the convolutional layers. In order to enable the comparison of the baseline architecture that accepts a fixed input image with one that accepts arbitrary input sizes the traditional fully connected layer at the end of the layer was substituted by a global average pooling layer. As pointed out previously, the global average pooling layer is used to remove the fixed size input constraint and to provide fixed length vector representations independent of the received input shapes. The global average pooling is followed by a fully connected layer whose output is fed to a softmax layer. Finally, the softmax layer produces a probability distribution over the 23 class labels of the data set. A schematic representation of the model in use is presented at Figure 8.

Figure 8. Baseline model architecture



Going into more detail about the parameters of the layers of the CNN, the first convolutional layer filters an input image of size 224x224x3 with 32 kernels of size 5x5x3 pixels. Same padding is used, meaning that the output shapes of the convolutional layers are zero-padded in such a way that they do not differ in size with the input. The second convolutional layer takes as input the output of the first max-pooling layer and filters it with 64 kernels of size 5x5x112. The third and fourth convolutional layers filter the received feature map of size 5x5x56 and 5x5x28 with 128 and 256 kernels, respectively. The global average pooling layer returns the average of the entire feature map of the previous layer resulting to a vector of 256 values. Finally, the last fully connected layer has 23 neurons that correspond to the medium categories. We trained the models using stochastic gradient descent with a batch size of 128 examples and a learning rate of 0.001.

5 Methodology

In this section, the methods used in our experiments are explained. We conducted in total three experiments in a multi-gpu setting. In our fist experiment, the bucket to bucket methodology (B2B) which is a key component of all our experiments was tested. Our second experiment included the B2B method combined with mixed precision training (MXP). In our third and last experiment, besides MXP and B2B, the spatial pyramid pooling SPP layer was integrated in our CNN.

5.1 Bucket to bucket methodology- B2B

The first experiment we performed is the implementation of the B2B methodology in a muti-gpu setting. In the next two sections, the B2B methodology and the modifications that needed to take place so that multi-gpu training would be made possible are explained.

Despite the indisputable efficacy of CNNs and their state-of-the-art performance in many recognition tasks, the high resolution images of our data set pose an additional computational constraint since training a network with large-scale high resolution images can be prohibitively expensive and imposes additional limitations both on the architectural choices of the network and to the size of the batches used:

- the DRAM capacity limits of the GPU(s) in the system eventually limits the size of the CNN that can be trained encouraging smaller number of layers, smaller number of filters per layer.
- training a network with relatively large batches is not applicable in that case because the memory needed to store the high resolution images in large batches exceeds by far the amount of available memory in current GPUs.

To tackle these problems, the conventional pre-processing step before feeding the data to a CNN when applied to images of arbitrary size is to fit the input image either using cropping, resizing or padding. However, the options of resizing and padding are not regarded as considerable choices because of our desire to keep the lossy fixed resized or cropped images aloof.

As a consequence, the only option that remains to preserve the original size representation is padding.

By taking into account the high variability in size and aspect ratios that the images exhibit the most naive approach to follow would be to train the images one by one. But this would lead to excessively large training times. Another alternative would be to create random batches and pad all the images to the maximum dimensions of the largest image in each batch. In the worst case scenario, one where the largest and smallest image are allocated within the same random batch the number of padded pixels for the smallest image would be X, which is unacceptable considering the total number of actual pixels of that image Y. So, this approach would again lead to an inefficient memory allocation and result in training times we are not willing to tolerate because of the tremendously small batch sizes. Besides that, the padding of the images introduced in this scenario would be very large increasing the amount of noise in the images and, thus, hampering the ability of the network to generalize. An illustrative example of the disastrous effect such a policy could entail is presented in Figure 14, where a relatively small image of our dataset is padded to the dimensions of the largest image in the batch. In this example, a batch size of 2 is used just for visualization purposes. A large amount of padding in that case would result in noisy representations of the input images and to poor generalization performance.

Another undesirable effect of such a policy is also the rapid increase in the computational cost originating from the inefficient memory management of the batches. Let's recall the worst case scenario where the smallest image in the data set resides in the same random batch as the largest one. The padded batch created would have the dimensions of the largest image, forcing it to occupy more memory resources than it would have if the smallest image was batched with other images in the dataset of similar size. A representative example of a good padding policy is illustrated in Figure 15, where it is noticeable that the amount of padding applied to the example images has significantly decreased. Thus, to diminish the negative effects of a random padding, the images should be grouped in batches in such a way that the padding area in each batch is minimized. Afterwards, a dynamic batch size could be applied in an effort to optimize the memory allocation, by creating larger batches for smaller images and smaller ones for larger ones. That way the full representations of the images can

be preserved without extreme padding and severe memory consumption.

Figure 14. Sample figure of bad padding scenario

Therefore, to work around this memory capacity bottleneck as well as the possible performance degradation due to noise, a bucketing methodology was implemented. The key rationale behind this, as explained previously, is that the padding cost in each batch should be minimized. So, we have to select the images in the batches in such a way that the padding cost defined in equation 5.1 would be minimal.

Figure 15. Sample figure of good padding scenario

$$\min \sum_{i=1}^{M} \sum_{j=1}^{N_i} \left(\max width_{ij} * \max height_{ij} \right) - \left(width_{ij} * height_{ij} \right) \tag{5.1}$$

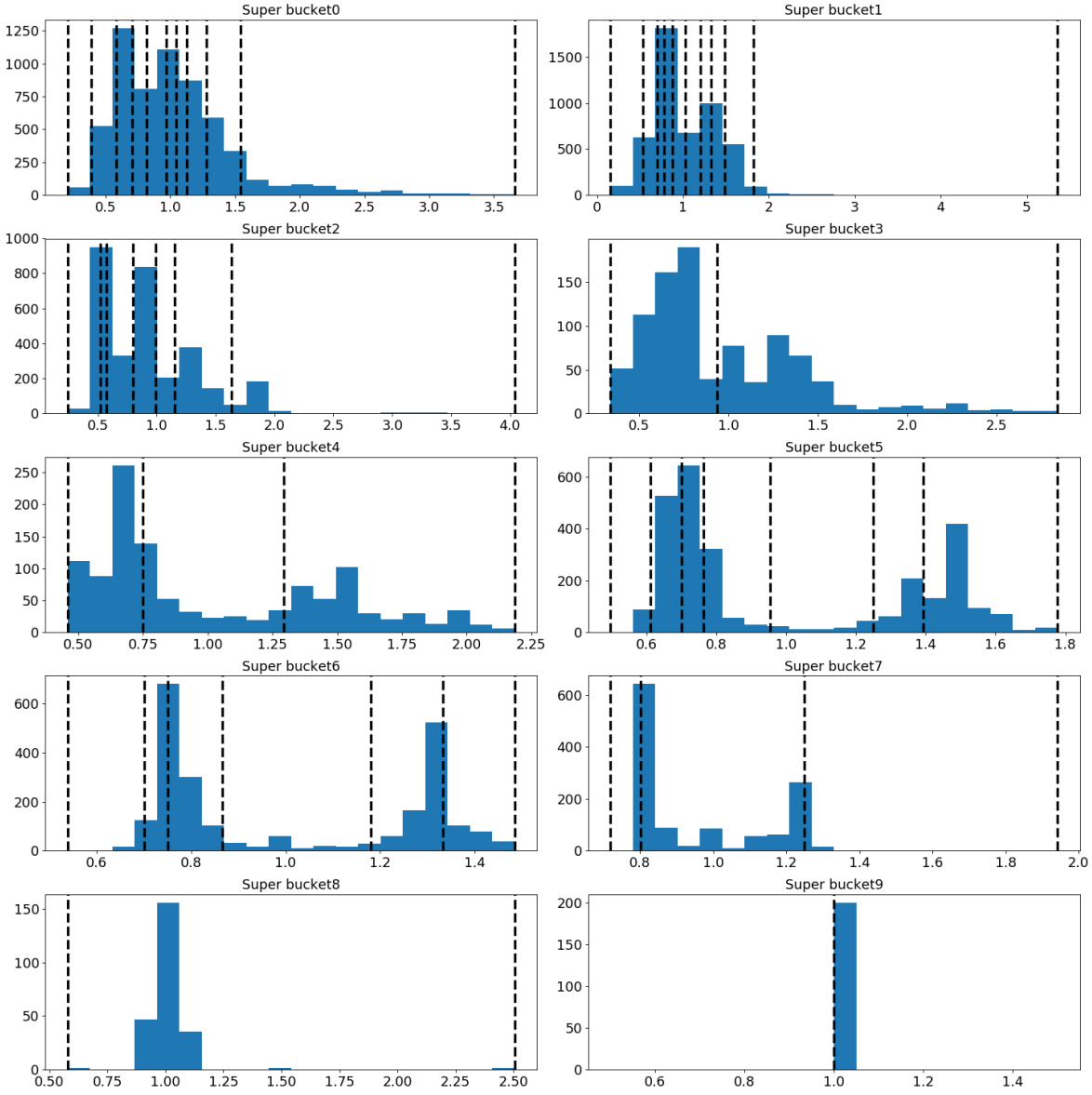
In order to obtain a more complete understanding of the bucketing task, a reference should be made to the knapsack problem. In the knapsack problem, we are given a knapsack of capacity B and a set of items N. Each of the items have size s_i and profit p_i. Given a subset of elements $A \subset N$, the functions $P(A) = \sum_{i \subset A} s_i$ and $S(A) = \sum_{i \subset A} w_i$ correspond to the total

profit and size of the subset of items respectively. The goal is to select a subset of items in such a way that the profit in the subset P(A) would be maximized and the sum of the weights in A would be less or equal to the knapsack capacity B. This optimization problem is NP-hard, as there is no known polynomial algorithm which can tell, given a solution, whether it is optimal. One of the recommended solutions makes use of a greedy algorithm that sorts items in increasing order of v_i/w_i ratio and adds successive elements in the sack until the maximum capacity is reached. If the last item's weight exceeds the capacity of the sack, a fraction of it that fits the sack is added.

To reduce the search space needed to be explored, the images are first divided into M buckets of similar image size using equally spaced intervals. Ten size buckets are created ranging from 250,000, which is the bottom limit of the size of the images to 16,000,000 pixels. Inside the super-buckets, the target goal is to create subsets of images (mini-buckets) in such a way that the minimization criterion of 5.1 is satisfied. Motivated by the greedy knapsack algorithm, in each of the super-buckets the images are sorted in increasing order according to their aspect ratio. Our experimentation included also other sorting criteria, such as the height, the width, the size of the images. It has been proven that the aspect ratio yields better results. After being sorted, the initial mini-bucket boundaries are initialized by dividing the images in equally sized chunks. The initial total padding cost of the mini-buckets is calculated. If the total cost does not increase, contiguous elements are moved between adjacent mini-buckets and the bucket-boundaries are updated. This process is repeated until the total cost is not decreased further and the resulting boundaries define the limits of each mini-bucket. In Figure16, the aspect ratio distribution of every super-bucket in the training set is presented. The vertical lines correspond to the limits of the mini-buckets.

The initial step of splitting the images in equally spaced buckets (super buckets), entails the risk of ending up with buckets which are characterized by an over-representation or under-representation of specific classes. From the distribution of the classes on Figure 5 in each sub-bucket it can be seen that such an effect is not counteracted in the case of some labels. We can observe that the vast majority of images that correspond to the *albumen photografen* are present only in the first super bucket. Also, the under-representation of some labels is clear in the last super bucket that contains 4KX4K images.

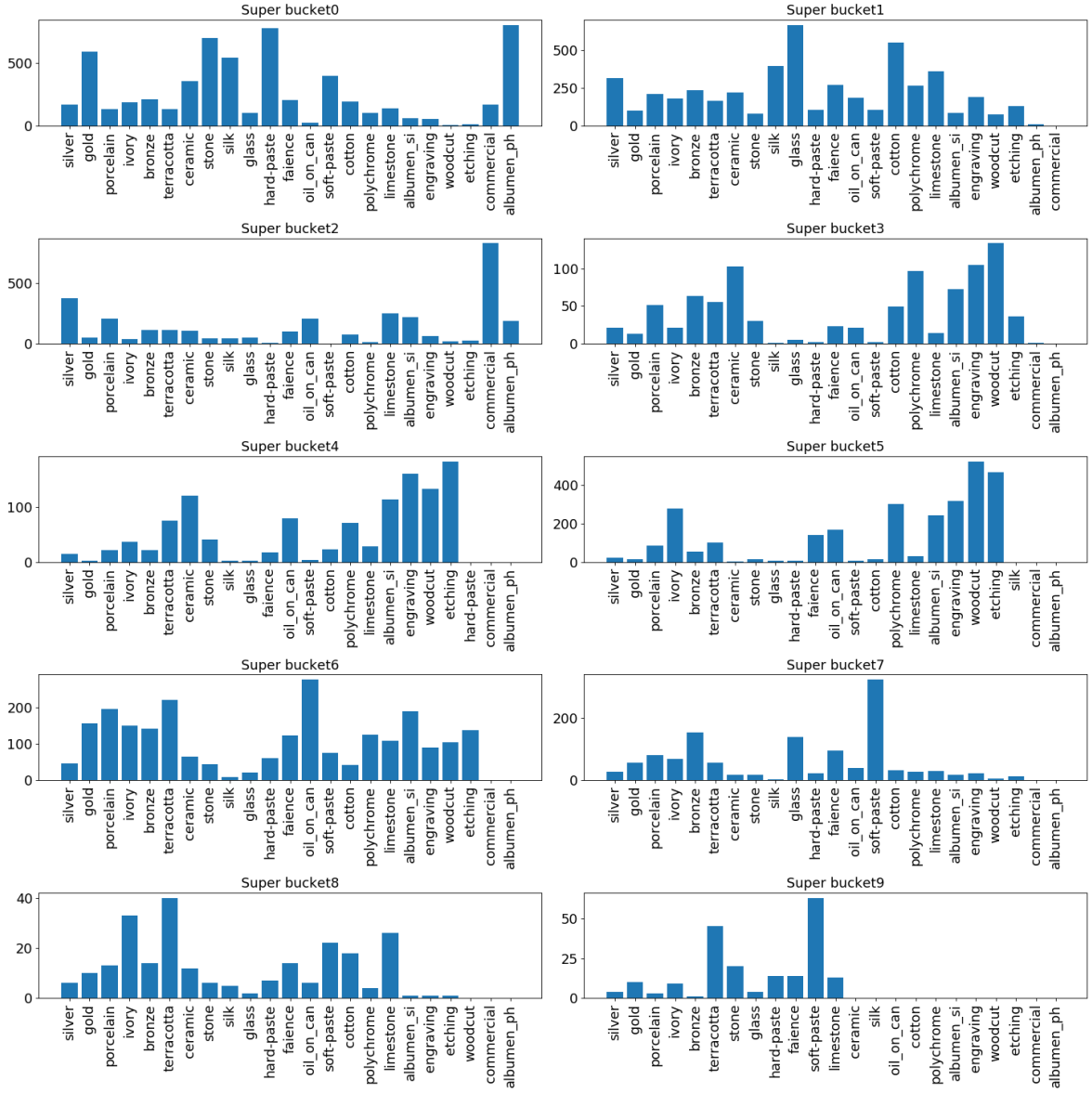
Figure 16. Mini-bucket boundaries inside every super-bucket



5.2 Training with multiple GPUs

The implementation of the B2B strategy results in 88 mini-buckets in total, the batch size of which ranges from 16 for the smaller sized images to 2 for the larger ones. Because of the low memory capacity of the GPUs that enforces these small batch sizes in the mini-buckets, multi-gpu training is considered essential for our experiment. Parallelizing the model across multiple GPUs expedites the decrease of the training time as it processes simultaneously as

Figure 17. Distribution of classes in every super-bucket



many batches as the number of available GPUs. Because every GPU can accommodate one batch at a time, batches are divided among multiple GPUs, and processed in parallel. The machine on which our experiments were performed has four NVIDIA V100 GPUs with a memory capacity of 16Gb each.

Additionally, as highlighted in section 1, the images are gathered in subgroups given their size and aspect ratio and afterwards a different batch size is utilized for each sub-bucket in

an effort to exploit efficiently the memory resources. Therefore, the learning rate has to be accordingly adjusted for every different batch size. That's why, we make use of the hyper parameter-free linear scaling rule which is an empirical rule that associates the learning rate with the batch size. The essence of this rule can be summarized as follows: *When the mini batch size is multiplied by k, the learning rate is also multiplied by k.* In contrast to our experiment, this linear scaling rule when initially proposed by P. Goyal et al. 2017 was tested in a large batch size regime in an attempt to maintain the generalization accuracy when using large batch sizes and match the small mini-batch accuracy. In our experiments, the linear scaling rule is deployed in a small batch size regime to enable the multi-gpu training using a dynamic batch size.

The learning rate for our experiments is defined as $\eta = \frac{kn}{128}$ where n denotes the reference learning rate that corresponds to a 128 batch size and k the new batch size. In multi-gpu training, for a setting of N available GPUs the new batch size can be defined as the sum of the individual batch sizes residing in each GPU:$k = \sum_{i=1}^{M} k_i$ Because of the different batch sizes the gradients to do not contribute equally to the final mean. Therefore, the importance of the batch sizes should also be reflected when it comes to the computation of the gradients after they are accumulated in the CPU. Instead of computing the arithmetic mean of the gradients, the weighted mean of the gradients is calculated to incorporate the information of the different weights: $\frac{\sum_{i=1}^{M} w_i * k_i}{\sum_{i=1}^{M} k_i}$, where w_i denotes the gradients and k_i the batch size in each GPU.

5.3 Mixed precision training- MXP

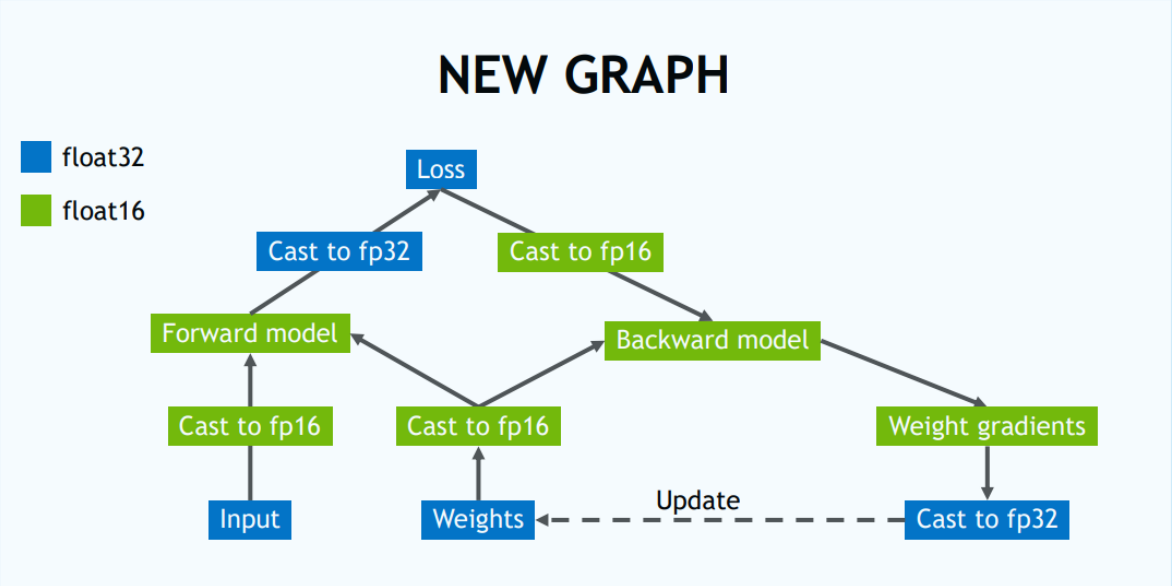
The small batch sizes as well as the computationally expensive mathematical operations during training because of the large matrices render B2B strategy, even in a multi-gpu setting significantly slow. This is the reason why a second experiment was conducted incorporating mixed precision training (MXP). An overview of the key elements of MXP strategy is provided in this section.

MXP proposed by Micikevicius et al. 2018 is an efficient way to alleviate the rigid physical limitations introduced by the GPU. MXP aims at reducing memory bandwidth and train-

ing time by using a combination of single and half numerical precisions. The danger that emerges when training a model with half precision is that it can result in significant accuracy loss because of the narrowed dynamic representation of FP16 format. However, FP32 performance can be matched when deploying the following key strategies: maintaining a master copy of the weights and scaling the loss. This is mainly attributed to the following reasons:

- The weight gradients can become very small and not re presentable in the FP16 range. This is equivalent to significant loss of information since the gradient values cannot not be preserved and thus, are converted to zero. Even in the case of FP16 representable weight updates, the ratio of the weight and the weight gradient can be very small. So any additional arithmetic operation conducted to update the weights would shift them to zero. Maintaining a master copy of the weights allowing the weights to be computed in FP32 format can eliminate the aforementioned undesirable effects.
- When converting the gradients to FP16 format, the vast majority of them falls below the minimum representable range and consequently are set to zero. An easy and effective way of shifting the gradients so that they can occupy the FP32 representable range and preserve their values is to scale the loss computed in the forward pass, prior to the optimization step.

Figure 18. Mixed precision training process



In our experiment, we performed MXP in combination to the B2B method introduced in the previous section. A schematic representation of the modifications that took place to enable MXP is presented in Figure 18. It is notable that the vast majority of memory intensive operations such as the generation of the feature maps and their gradients are executed using FP16 format. The weights and their updates incur an order of magnitude lower memory usage than the activations and the feature maps. So retaining the weights in FP32 format even though it increases the overall memory consumption compared to half precision training, it does not encumber substantially the system memory. Actually, we managed to halve the memory consumption since it allowed the increase of all the batch sizes by a factor of 2. Regarding the scaling rule, after trying some values in the range 8-32k it turned out that multiplying the loss by the scaling factor of 128 matches the FP32 performance.

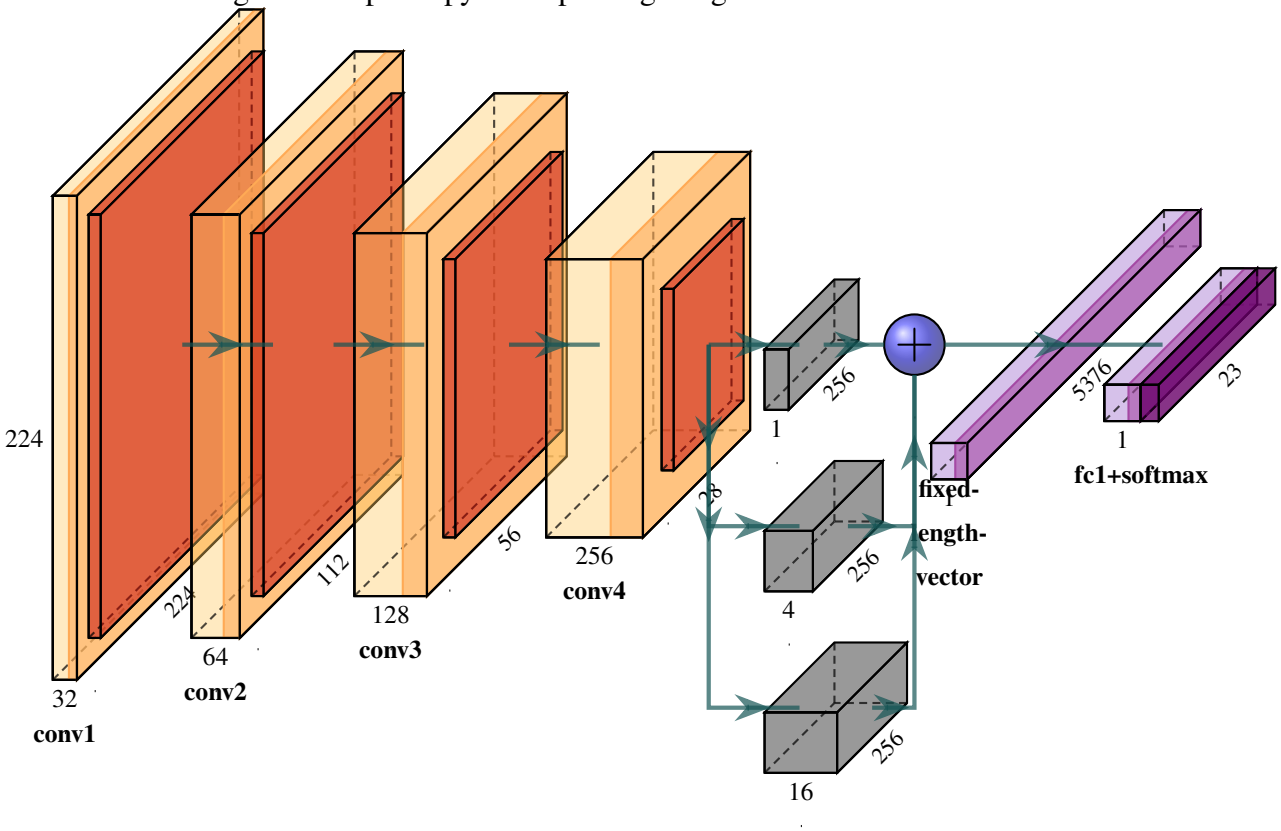
5.4 Spatial Pyramid Pooling- SPP

Our last experiment includes an architectural change that is not memory costly. The GOP layer is removed and replaced by a SPP layer. The basic idea behind this modification is that SPP will enable the model to capture resolution at different scales and it can enhance the overall performance of the model.

Spatial pyramid pooling can generate a fixed length pooling representation given a sequence of convolved feature maps regardless of the image size. This is done by a series of pooling operations in a pyramid level way. Given a specific level l, the pooling layers operate on the feature maps in a sliding window manner in a neighborhood window of size $\left\lceil \frac{w}{l} \right\rceil$ x $\left\lceil \frac{h}{l} \right\rceil$ and stride $\left\lfloor \frac{w}{l} \right\rfloor$ x $\left\lfloor \frac{h}{l} \right\rfloor$, where w,h correspond to the width and height of the receives as input to the spp feature maps. That way independent of the image size in a specific pyramid level l every feature map obtained by the last convolutional layer is divided in lxl sub regions.

In our approach we considered three levels. At level-0 the entire feature map is max-pooled resulting in a 256-dimensional (equal to the number of filters in the last convolutional layer) vector. At level 1, the feature maps are partitioned in 4 spatial bins resulting in 4x256 dimensional vector. Finally, at level-2 the same feature maps are divided 16x16 spatial regions resulting in 16x256 dimensional vector. The vectors of all the pyramid layer are concate-

Figure 19. Spatial pyramid pooling integrated to the model



nated into a single 5376 fixed-dimensional vector. A diagram of the architecture we used is provided in Figure 19. Incorporating spp in our model can augment its learning capability because of three primary reasons:

- it entails complementary spatial information
- it produces multi-level spatial bins, that have been shown to be robust to object deformation and
- it extracts features in multiple scales

Is is also worth mentioning that in previous studies SPP has been tested in a multi-size training setting, meaning that two predefined sizes of images (180x180, 224x224) were trained by two-fixed size networks that shared parameters. In our experiment, we have used one model with input images that covered a broad range of different sizes from 250,000 to 16,000,000 pixels.

6 Results

In this section the results of our three experiments and the baseline experiment are presented. Besides the learning curves that assess the performance of our models, the confusion matrices together with some sample figures from the data set are displayed.

6.1 Baseline Results

In the this section the training and validation curves obtained from the baseline experiments are provided. The best accuracy achieved is 63%. Looking at the loss curves, it is notable that the model is insufficient to learn the parameters of the model without considerable overfitting. But, two of the most prevalent approaches used to combat overfitting turned out not to be appropriate for our experiments. Because of the B2B methodology that makes use of high resolution images the option of data augmentation was no considered a reasonable choice because of the additional processing time that would be required to apply data transformations like image translations and horizontal reflections. We also experimented with Dropout that did not succeed in preventing overfitting. We hypothesize that this can be attributed to the relatively small capacity of the network. Dropout can be regarded as a regularizer that stochastically sets a number of input or hidden neurons of the network to zero during training. Thus, dropout can be considered as an ensemble approximation that facilitates training a large number of different architectures that share the same weights in parallel. However, if the model capacity is already low like ours that is relatively small and shallow, pruning it further hurts performance.

The confusion matrix illustrated in Figure 43 provides us also with some insight about the discriminatory power of the model with respect to each of the labels. *Oil on canvas*, *woodcut* and *albumen-photografen* are among the most easily distinguishable materials. Apart from the known materials of *silver*, *terracotta*, *glass*, etc that are easily identified there are also others that do not carry the same discriminative potential even with a human eye such as the pairs *limestone-stone*, *hard-soft-porcelain-soft-paste-porcelain*, *engraving-etching* and *ivory-terracotta* rendering the task of predicting the medium accurately not that easy to solve.

Figure 29. Stone

Figure 30. Limestone

More evidence of this are provided in Figures 29, 30, 31, 32, 33, 34, 35 and 36 where some example images of the mislabeled aforementioned pairs are provided. In the left side of the figures the correct label is displayed, while in the right side is depicted the label each artifact is incorrectly assigned to. In Figure 29 and 30, a stone material is falsely classified as limestone. In general, stone and limestone are two elements that share the same textural characteristics and it is quite difficult to be differentiated. The same applies for *etching* and *engraving* that are both printmaking methods used to produce images in hard surfaces, usually metallic ones. Their dissimilarity lies in the fact that in the case of etching the image is produced with the use of acid, whereby in the case of engraving it is produced by a sharp and pointed tool that cuts line into a metal surface. The output images of both procedures are really hard to be differentiated by a non experienced human eye. Recognizing between *hard-paste-porcelain* and *soft-paste porcelain* is equally difficult. Besides their difference in the ability to resist fracture, the two materials look very much alike as can be demonstrated in Figures 36 and 35. The CNN is also very poor at differentiating in general the *stone* material, that is confused with many other material, the most prominent of which are with decreasing order of frequency *limestone*, *cotton* and *glass*.

To further support the argument of the familiarity between some classes, the distances of the confusion matrix were used as features in an agglomerative clustering algorithm in order to help us visualize the close affinity of the previously mentioned pairs of classes. In Figure 40

Figure 31. Silver

Figure 32. Glass

Figure 33. Engraving

Figure 34. Etching

Figure 35. Hard-paste porcelain

Figure 36. Soft-paste porcelain

the distance between the classes is illustrated in a dendogram. Probably, some classes with small distance could be amalgamated into one super-class for future experiments, such as *hard-paste-porcelain*, *soft-paste-porcelain* and *porcelain* and *engraving* and *etching*.

Figure 37. Accuracy of the baseline experiment

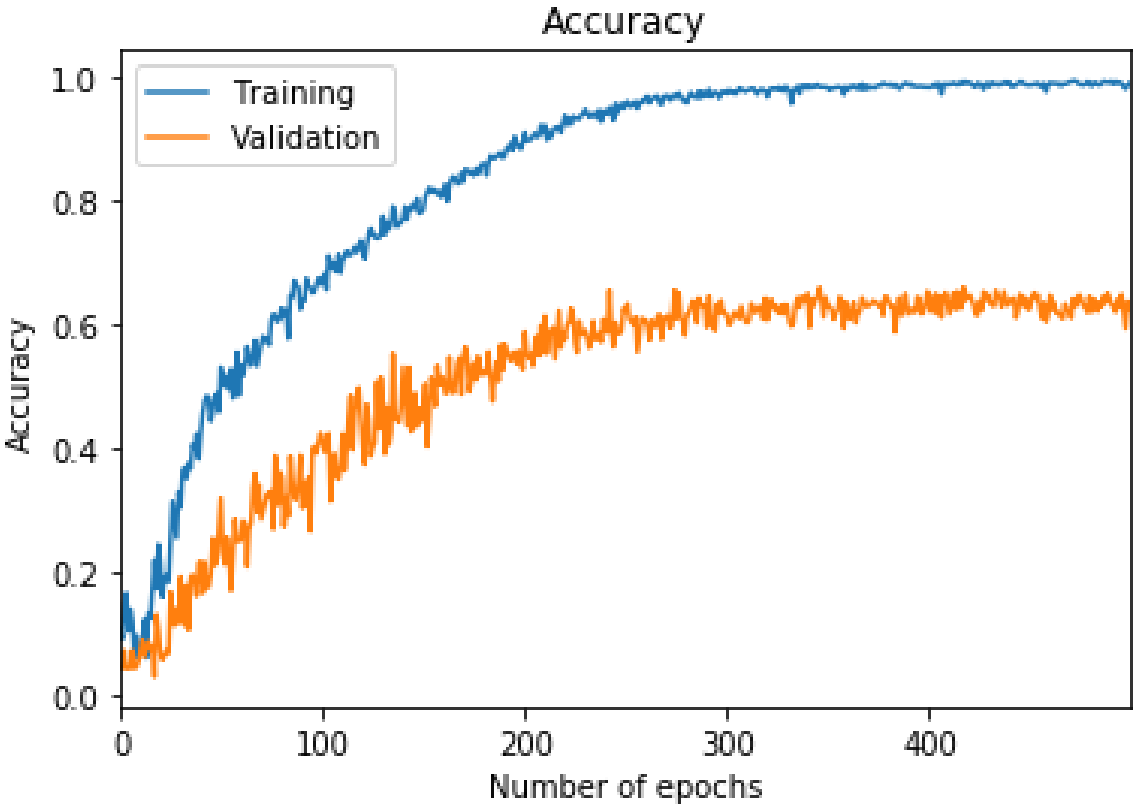


Figure 38. Loss of the baseline experiment

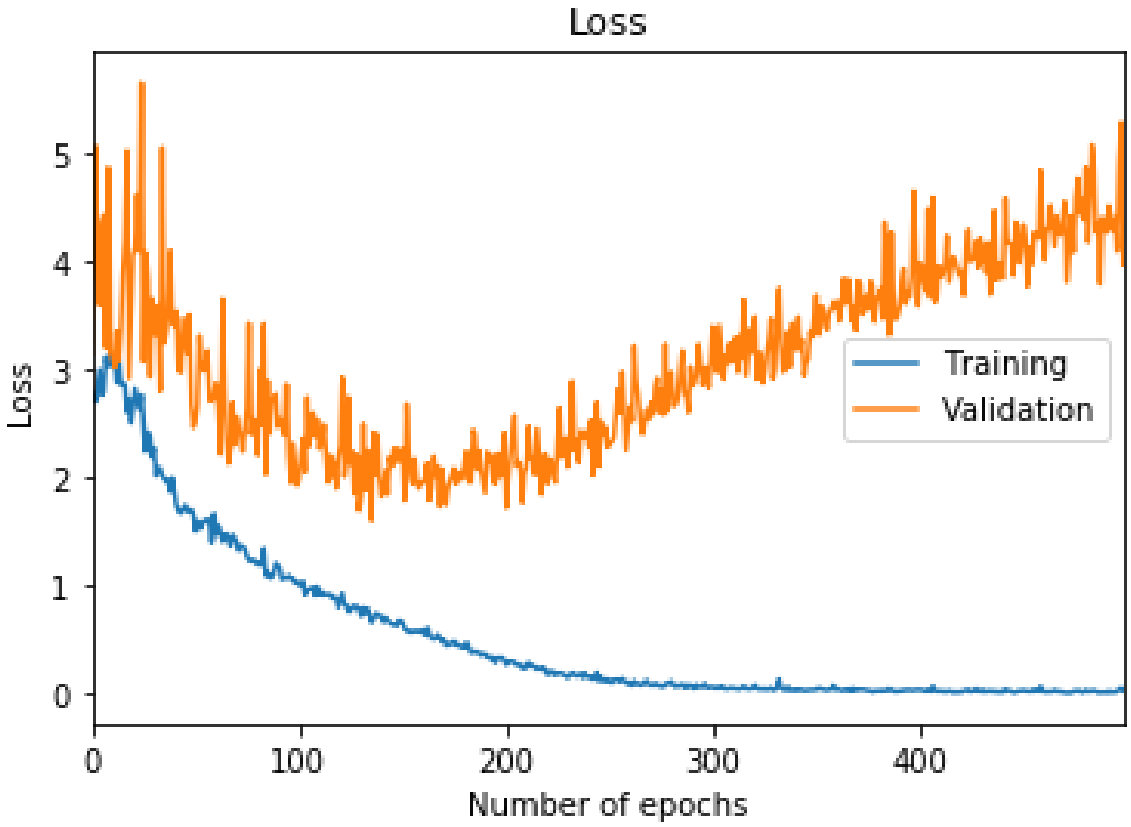


Figure 39. Confusion matrix of the baseline experiment

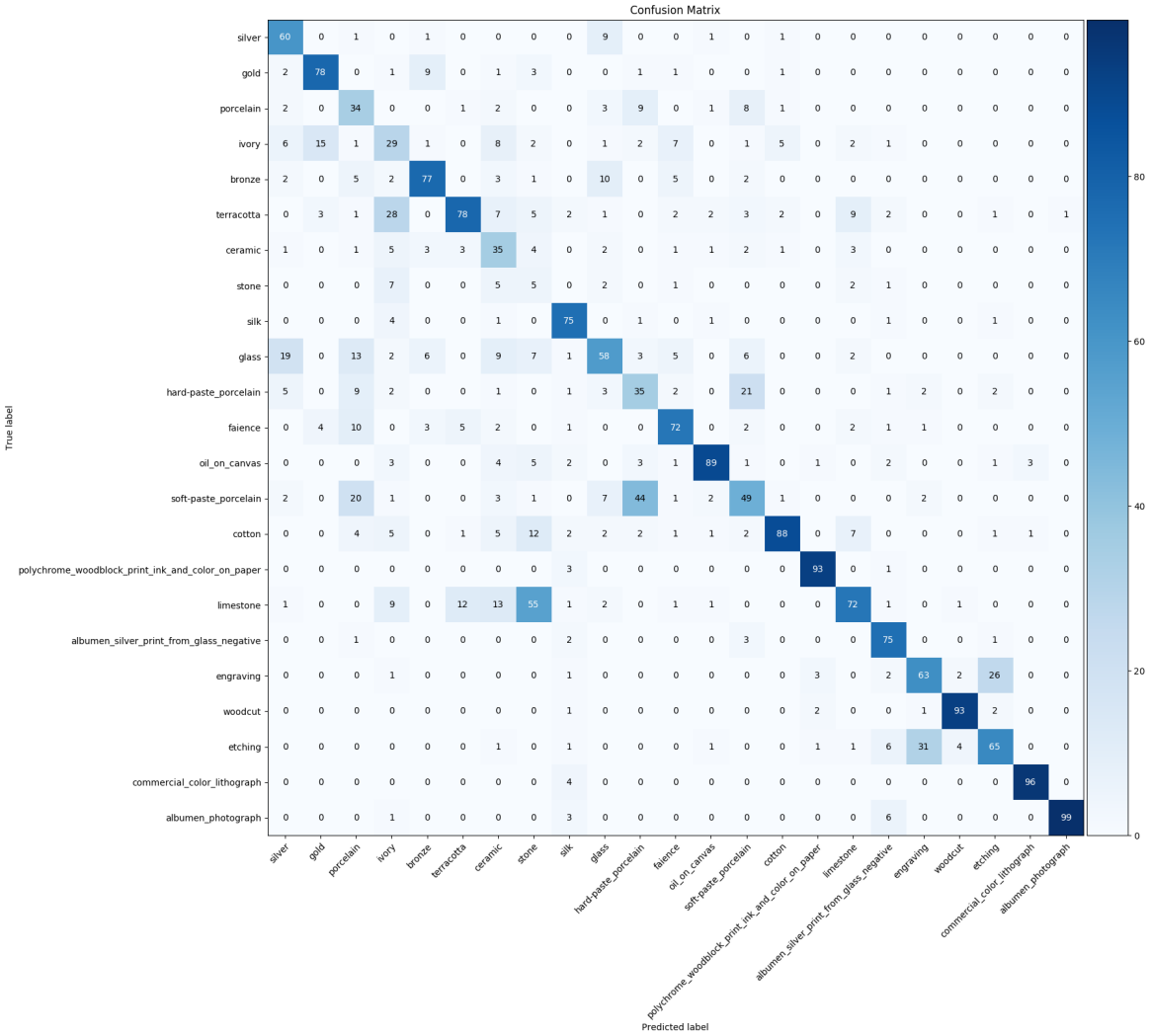
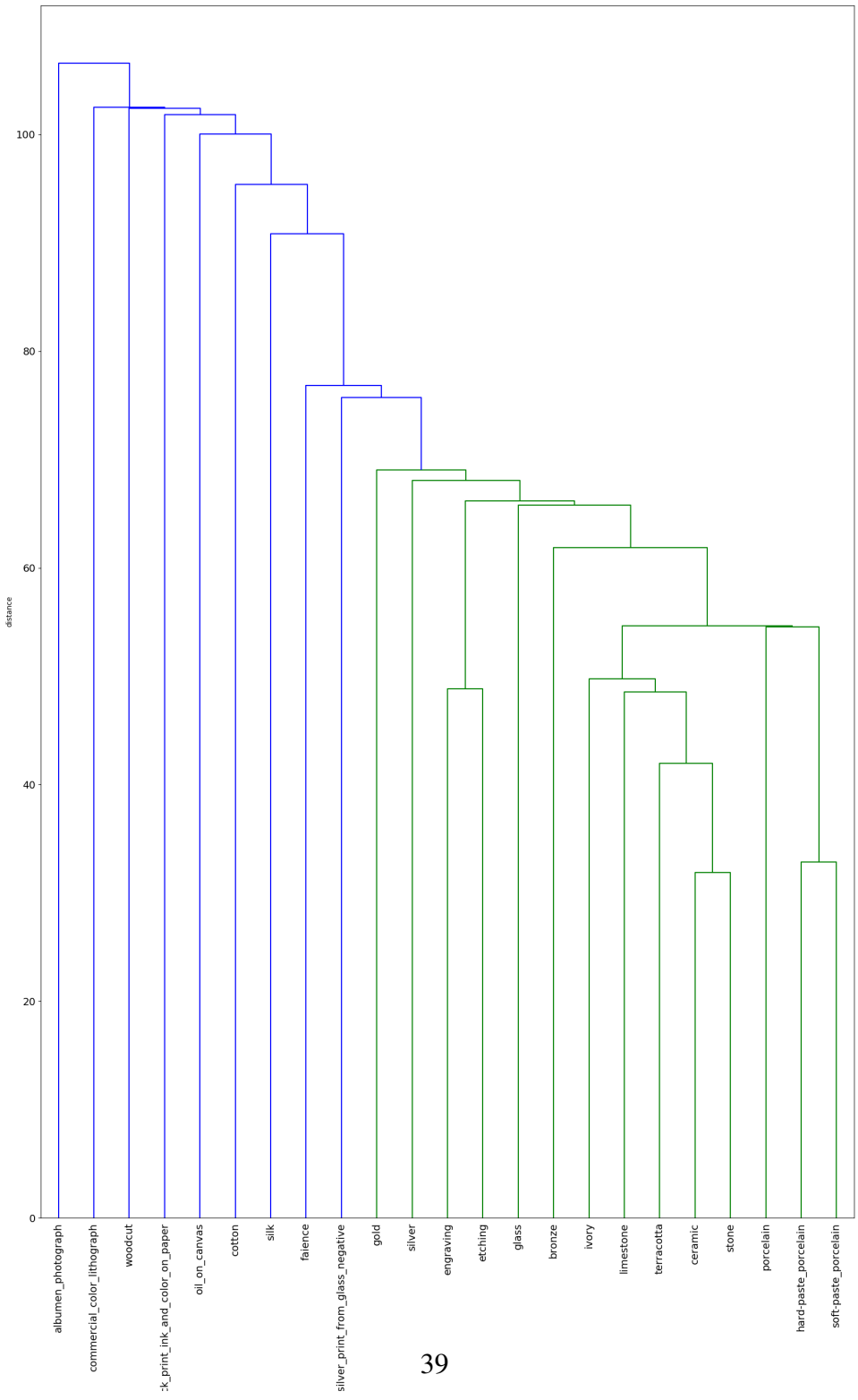


Figure 40. Dendogram of the classes



6.2 Bucket to bucket methodology

In this section the learning curves of the B2B methodology are displayed. The highest accuracy that the network obtains is 59% as can be seen in Figure 41. Looking at the validation curves in Figure 42, we can see that the validation loss is still decreasing, but again there is significant overfit. An interesting observation worth mentioning is the time of convergence compared to the baseline experiment. The time required for the model to converge is 70 epochs compared to the 200 of the baseline experiment. We hypothesize that this can be partially attributed to the small batches used in the B2B experiment. Several studies (Keskar et al. 2017; P. Goyal et al. 2017) have concluded that using larger batches can result in a degradation in the quality of the model because they tend to converge to sharp minimizers of the training and testing functions. In contrast, small-batch methods consistently converge to flat minimizers, which lead to better generalization and faster and more stable training. Another more intuitive explanation suggests that because of the noise introduced by the small batch sizes, each of which can be considered as a representation of the data set the model is not so susceptible to overfitting.

The confusion matrix in Figure 43 offers us some hints pertained to the behavior of the model. The variance of the diagonal of the confusion matrix is increased compared to the baseline experiments and extreme behaviors are reinforced, meaning that the model exhibits very good generalization capabilities for some materials and very poor ones for some others. The model generalizes very poorly and is not able to learn the underlying patterns of the *porcelain*, *stone* and *ceramic* materials. In particular, for those materials the correctly classified objects are below zero.

Figure 41. Accuracy of the B2B experiment

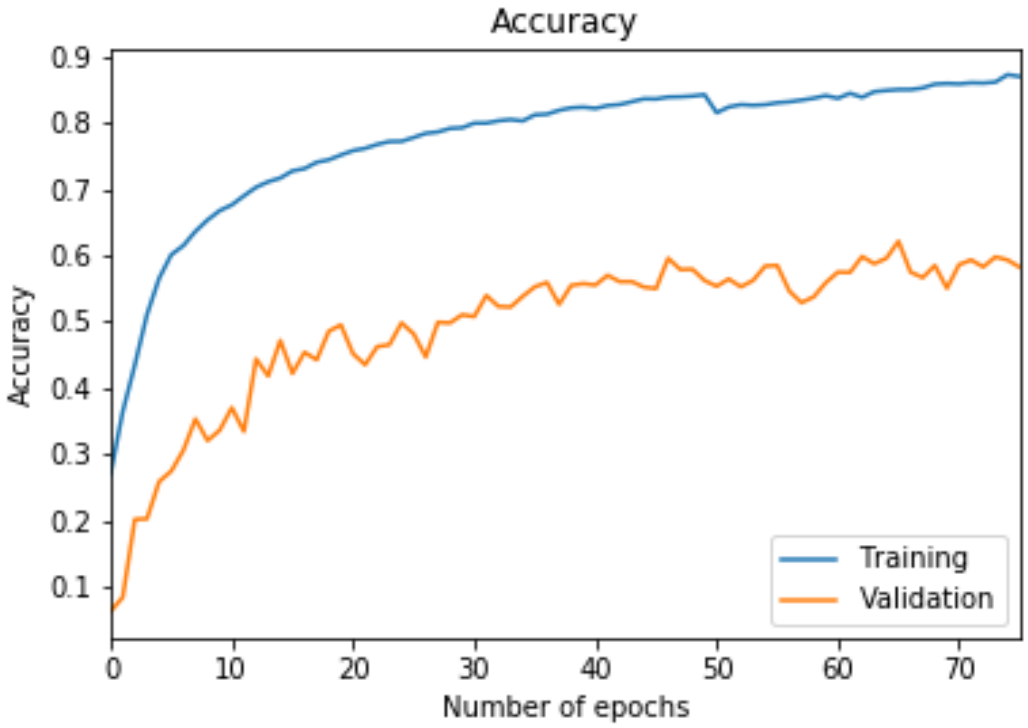


Figure 42. Loss of the B2B experiment

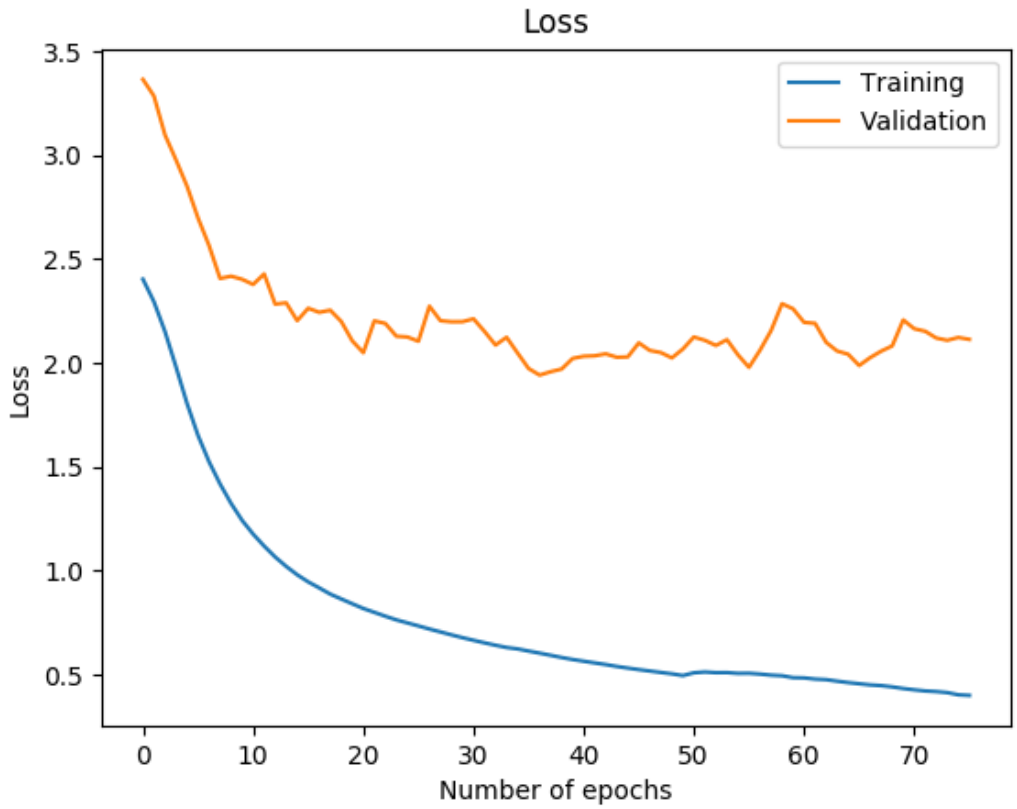
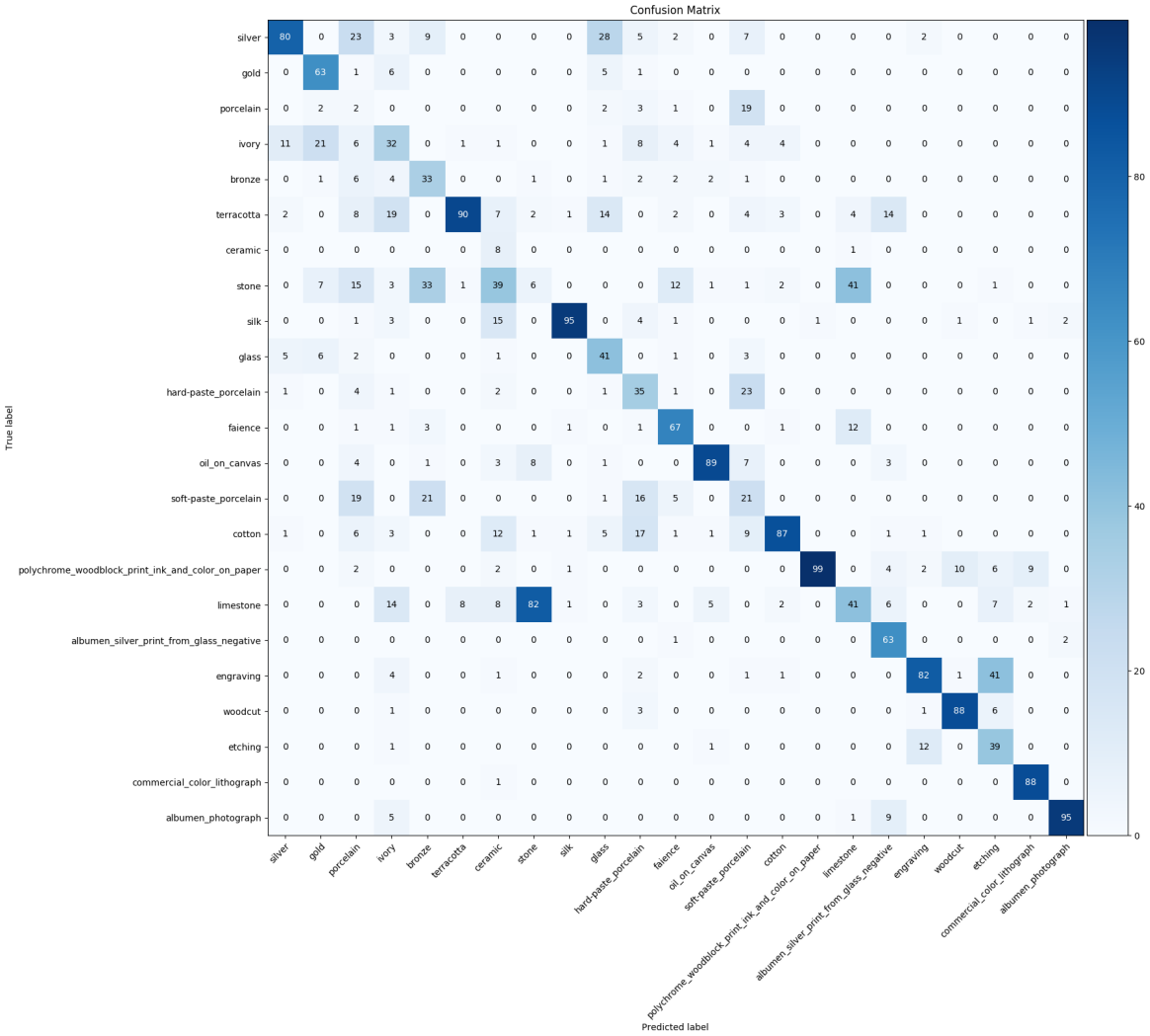


Figure 43. Confusion matrix of the B2B experiment



6.3 Mixed Precision Training

In an effort to accelerate the training process of the B2B experiment, mixed training precision was deployed to exploit the desirable effect of the decreased computational cost due to half precision arithmetic operations in conjunction with the decrease of the memory requirements. Our experiments were performed on NVIDIA V100 GPUs. In contrast to previous NVIDIA GPUs that supported only FP16 multiply-add operations, NVIDIA V100 GPUs

introduced Tensor Cores that support FP16 input matrices and product accumulation in either FP16 or FP32 ouputs. That way, they offer up to 8x more half precision arithmetic throughput when compared to single-precision, thus speeding up math-limited layers. However, even though we managed to roughly halve the memory consumption of the network by increasing the batch sizes of almost all the mini buckets by a factor of 2, the expected speed-up of the learning process was not achieved. This is attributed to the fact that that mixed precision training is not entirely integrated in the official version of TensorFlow, but is part of NVIDIA's latest TensrFlow version. So, the TensorCores were not enabled and the model was trained on the physical GPU of the system. In terms of convergence, the number of epochs needed for the model to converge is halved in comparison to the B2B experiment while reaching approximately the same accuracy as can be seen in Figure 44. We hypothesize that the faster convergence rate is attributed to the increase in batch size, which is the only change applied that differentiates the two models.

Figure 44. Accuracy of the B2B-MXP experiment

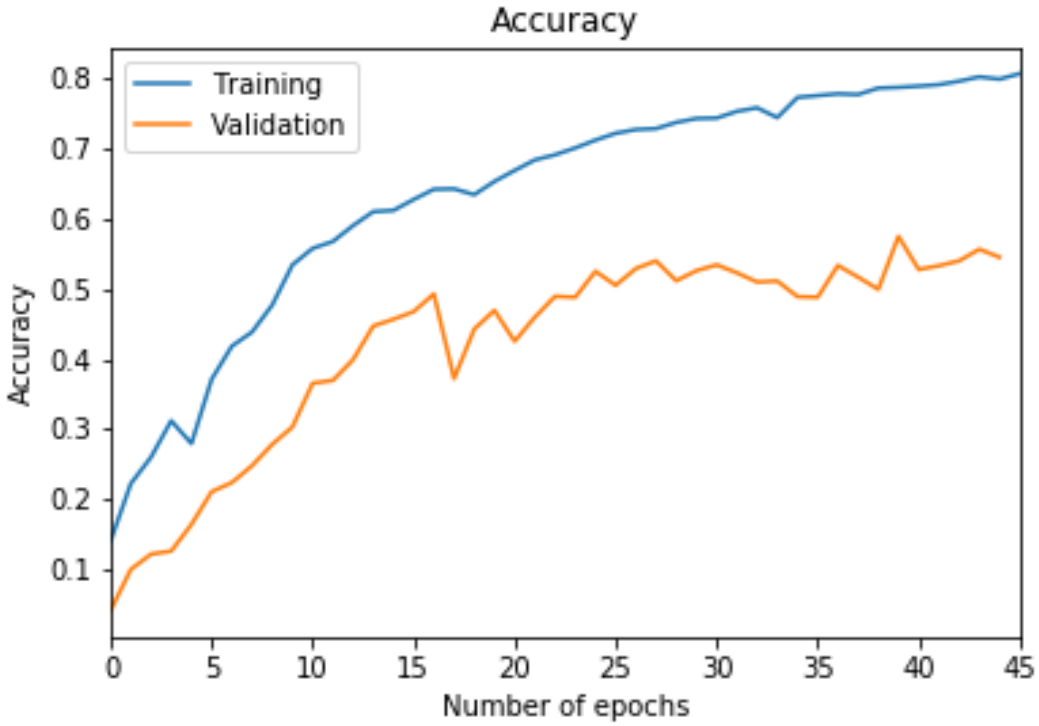
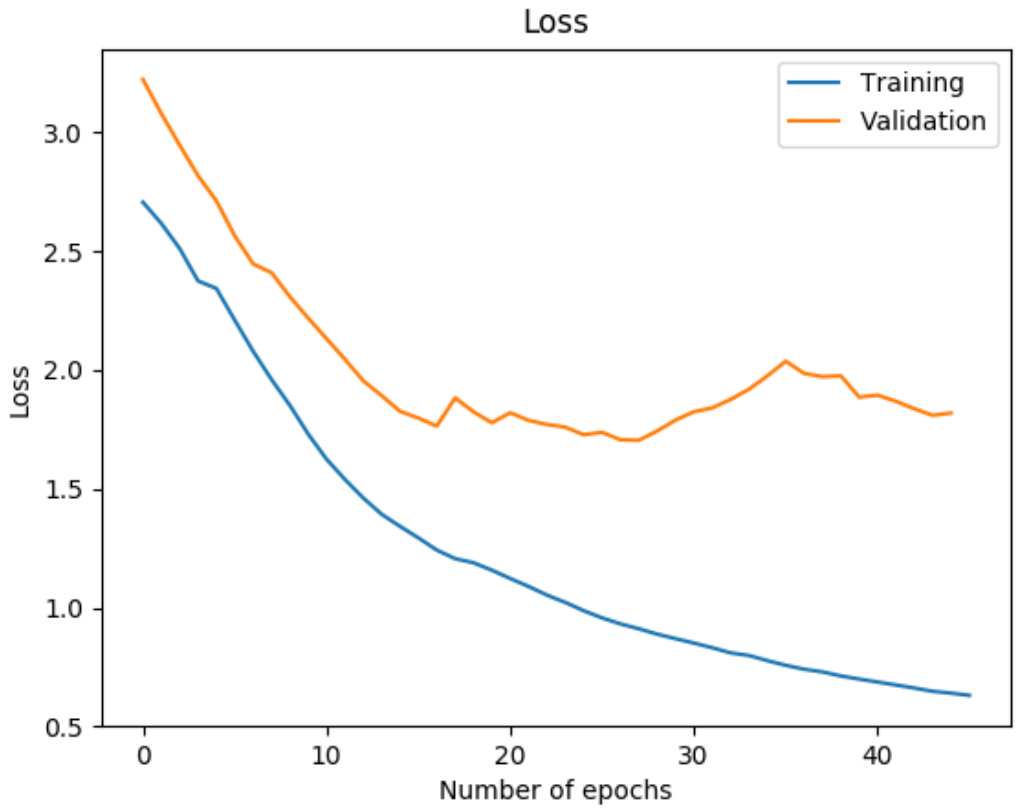


Figure 45. Loss of the B2B-MXP experiment



6.4 Spatial Pyramid Pooling

In the last experiment, in addition to B2B and MXP, SPP was introduced at the end of the network in order to enhance the performance of the model by capturing resolution at different scales. The last experiment provided the highest performance gains, both in terms of accuracy and convergence rate. Accuracy reached approximately 67%, while the number of epochs needed to reach convergence dropped to 35.

Going to the confusion matrix in Figure 48, we can see that compared to the B2B experiment because of the use of the SPP the model improved its ability to generalize the *porcelain* and *ceramic* materials. *Stone* remains a material very difficult to be recognized, probably because of its close affinity to other materials. In general SPP seems to be benefial for all the classes.

Figure 46. Accuracy of the combined B2B-MXP-SPP experiment

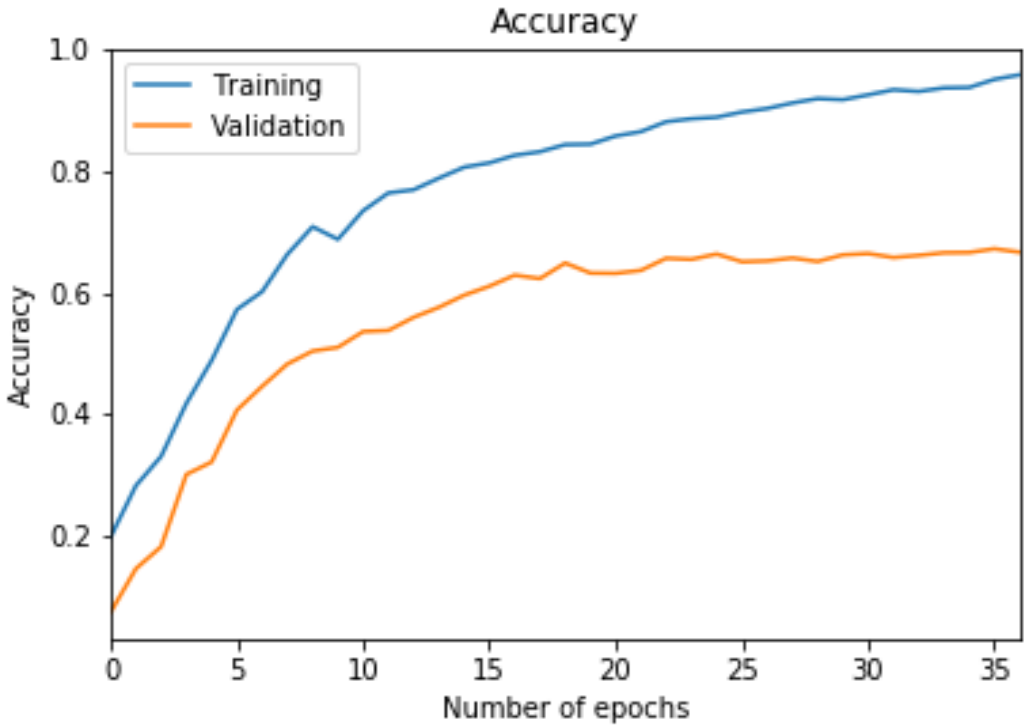


Figure 47. Loss of the combined B2B-MXP-SPP experiment

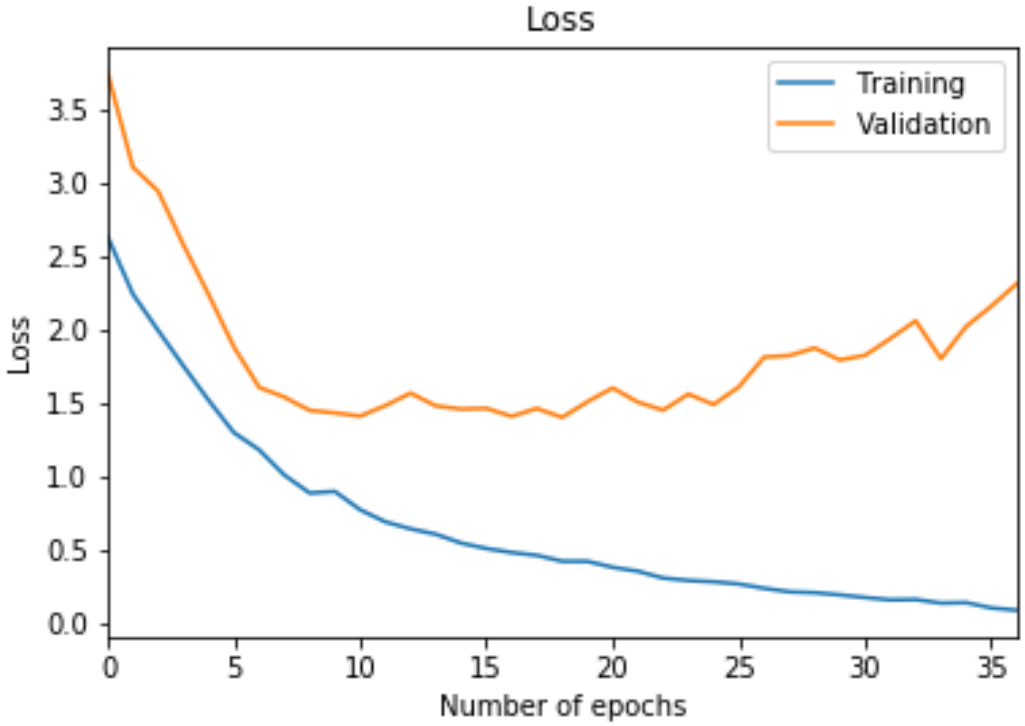
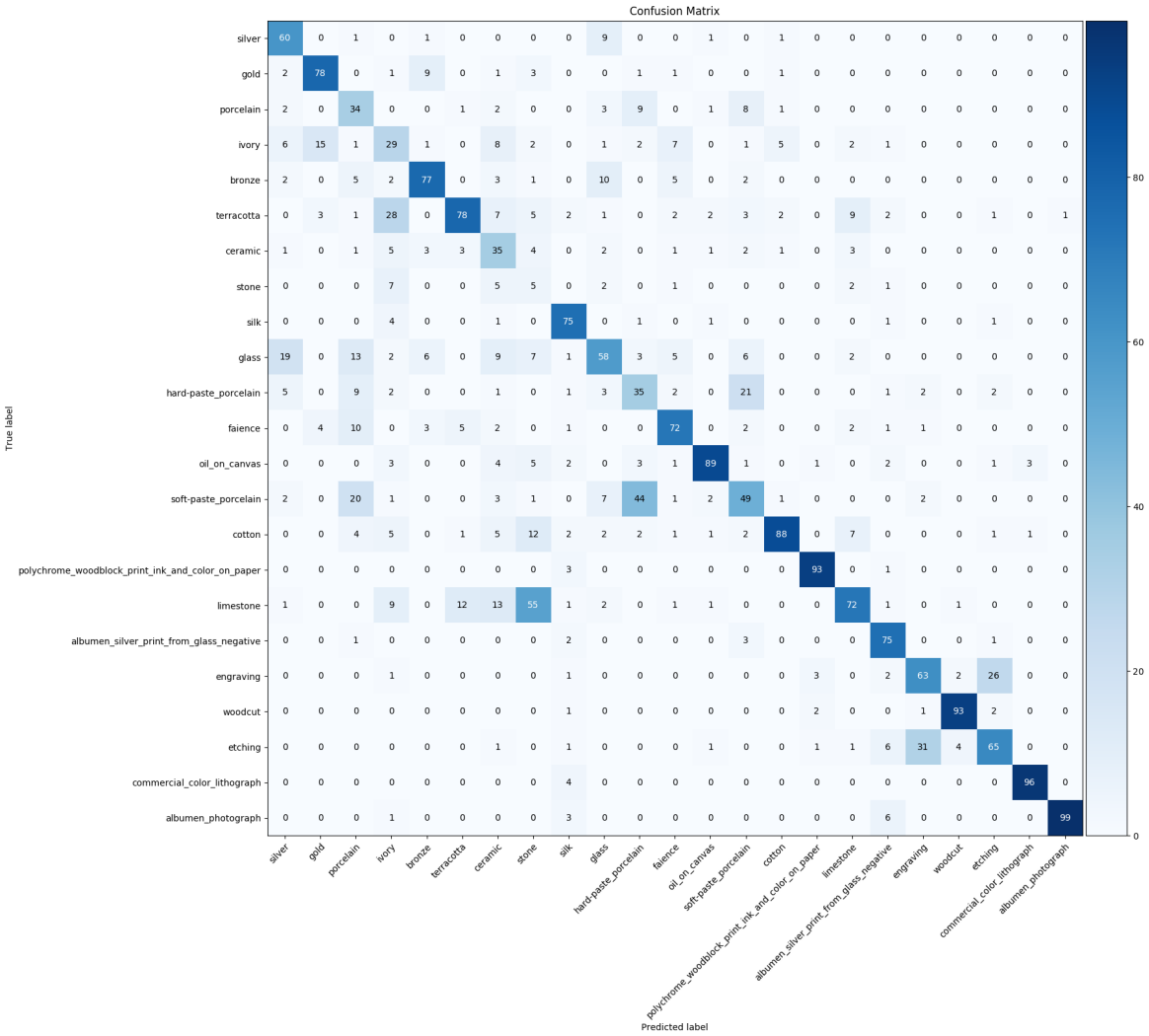


Figure 48. Confusion matrix of the combined B2B, MXP, SPP experiment



7 Conclusions

Several works have shown that CNNs can be easily adapted to different datasets and tasks. However, for extracting the deep features from these deep CNNs a fixed size input image is mandatory. In many fields, the most prominent of which is medical imaging, an increasing need of using high resolution images emerged because of their ability to provide more detailed visual representations. In this thesis, we present a novel mechanism for training variable sized high resolution images with CNNs. It is based on the B2B methodology that manages to process the images in their original form with a decreased computational cost. MXP was also applied on top of that to halve the memory requirements and accelerate training, and SPP to further enhance the generalization ability of the model by extracting features at variable scales. Our experimental results demonstrated that our method achieves performance comparable to that of the baseline experiment.

The results open up interesting avenues requiring further exploration. To draw safe conclusions and gain a more complete understanding of the B2B methodology, the experiments should be repeated also in other data sets. There are data sets that are probably more applicable for such an experiment. As it was already discussed in the data set used there is such close familiarity between some classes that makes it very difficult to be correctly classified.

Finally, one of the primary drawbacks of our method is that the training time is slow. Even though, this behavior is mainly subjected to hardware limitations, there is still room for improvement. Repeating the MXP experiment under the TensorFlow that resides in NVIDIA GPU Cloud (NGC) container registry and supports FP16 arithmetic in order to take profit of the acceleration in arithmetic operations is of high priority.

Bibliography

Cai, Zhaowei, Quanfu Fan, Rogério Schmidt Feris, and Nuno Vasconcelos. 2016. “A Unified Multi-scale Deep Convolutional Neural Network for Fast Object Detection”. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV,* 354–370. doi:10.1007/978-3-319-46493-0_22. https://doi.org/10.1007/978-3-319-46493-0%5C_22.

Çalik, Rasim Caner, and M. Fatih Demirci. 2018. “Cifar-10 Image Classification with Convolutional Neural Networks for Embedded Systems”. In *15th IEEE/ACS International Conference on Computer Systems and Applications, AICCSA 2018, Aqaba, Jordan, October 28 - Nov. 1, 2018,* 1–2. doi:10.1109/AICCSA.2018.8612873. https://doi.org/10.1109/AICCSA.2018.8612873.

Cetinic, E., and S. Grgic. 2016. “Genre classification of paintings”. In *2016 International Symposium ELMAR,* 201–204. doi:10.1109/ELMAR.2016.7731786.

Chen, Tianqi, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. 2016. “Training Deep Nets with Sublinear Memory Cost”. *CoRR* abs/1604.06174. arXiv: 1604.06174. http://arxiv.org/abs/1604.06174.

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. 2009. “ImageNet: A large-scale hierarchical image database”. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA,* 248–255. doi:10.1109/CVPRW.2009.5206848. https://doi.org/10.1109/CVPRW.2009.5206848.

Gong, Yunchao, Liwei Wang, Ruiqi Guo, and Svetlana Lazebnik. 2014. “Multi-scale Orderless Pooling of Deep Convolutional Activation Features”. *CoRR* abs/1403.1840. arXiv: 1403.1840. http://arxiv.org/abs/1403.1840.

Goyal, Priya, Piotr Dollár, Ross B. Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. 2017. "Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour". *CoRR* abs/1706.02677. arXiv: 1706.02677. http://arxiv.org/abs/1706.02677.

Goyal, Sachin, Can Zhao, Amod Jog, Jerry L. Prince, and Aaron Carass. 2018. "Improving self super resolution in magnetic resonance images". In *Medical Imaging 2018: Biomedical Applications in Molecular, Structural, and Functional Imaging, Houston, Texas, United States, 10-15 February 2018,* 1057814. doi:10.1117/12.2295366. https://doi.org/10.1117/12.2295366.

Gupta, Shikha, Deepak Kumar Pradhan, Dileep Aroor Dinesh, and Veena Thenkanidiyoor. 2018. "Deep Spatial Pyramid Match Kernel for Scene Classification". In *Proceedings of the 7th International Conference on Pattern Recognition Applications and Methods, ICPRAM 2018, Funchal, Madeira - Portugal, January 16-18, 2018.* 141–148. doi:10.5220/0006596101410148. https://doi.org/10.5220/0006596101410148.

Han, Xiaobing, Yanfei Zhong, Liqin Cao, and Liangpei Zhang. 2017. "Pre-Trained AlexNet Architecture with Pyramid Pooling and Supervision for High Spatial Resolution Remote Sensing Image Scene Classification". *Remote Sensing* 9 (8): 848. doi:10.3390/rs9080848. https://doi.org/10.3390/rs9080848.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition". *IEEE Trans. Pattern Anal. Mach. Intell.* 37 (9): 1904–1916. doi:10.1109/TPAMI.2015.2389824. https://doi.org/10.1109/TPAMI.2015.2389824.

Karpathy, Andrej, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Fei-Fei Li. 2014. "Large-Scale Video Classification with Convolutional Neural Networks". In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014,* 1725–1732. doi:10.1109/CVPR.2014.223. https://doi.org/10.1109/CVPR.2014.223.

Keskar, Nitish Shirish, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2017. “On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima”. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. https://openreview.net/forum?id=H1oyRlYgg.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. 2017. “ImageNet classification with deep convolutional neural networks”. *Commun. ACM* 60 (6): 84–90. doi:10.1145/3065386. http://doi.acm.org/10.1145/3065386.

Lawrence, Steve, C. Lee Giles, Ah Chung Tsoi, and Andrew D. Back. 1997. “Face recognition: a convolutional neural-network approach”. *IEEE Trans. Neural Networks* 8 (1): 98–113. doi:10.1109/72.554195. https://doi.org/10.1109/72.554195.

Lin, Min, Qiang Chen, and Shuicheng Yan. 2014. “Network In Network”. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. http://arxiv.org/abs/1312.4400.

Micikevicius, Paulius, Sharan Narang, Jonah Alben, Gregory F. Diamos, Erich Elsen, David Garcıa, Boris Ginsburg, et al. 2018. “Mixed Precision Training”. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. https://openreview.net/forum?id=r1gs9JgRZ.

Nandi, Debashis, Jayashree Karmakar, Amish Kumar, and Mrinal Kanti Mandal. 2019. “Sparse representation based multi-frame image super-resolution reconstruction using adaptive weighted features”. *IET Image Processing* 13 (4): 663–672. doi:10.1049/iet-ipr.2018.5139. https://doi.org/10.1049/iet-ipr.2018.5139.

Ren, Shaoqing, Kaiming He, Ross B. Girshick, and Jian Sun. 2017. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. *IEEE Trans. Pattern Anal. Mach. Intell.* 39 (6): 1137–1149. doi:10.1109/TPAMI.2016.2577031. https://doi.org/10.1109/TPAMI.2016.2577031.

Rhu, Minsoo, Natalia Gimelshein, Jason Clemons, Arslan Zulfiqar, and Stephen W. Keckler. 2016. “vDNN: Virtualized deep neural networks for scalable, memory-efficient neural network design”. In *49th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO 2016, Taipei, Taiwan, October 15-19, 2016,* 18:1–18:13. doi:10.1109/MICRO.2016.7783721. https://doi.org/10.1109/MICRO.2016.7783721.

Ronneberger, Olaf. 2017. “Invited Talk: U-Net Convolutional Networks for Biomedical Image Segmentation”. In *Bildverarbeitung für die Medizin 2017 - Algorithmen - Systeme - Anwendungen. Proceedings des Workshops vom 12. bis 14. März 2017 in Heidelberg,* 3. doi:10.1007/978-3-662-54345-0_3. https://doi.org/10.1007/978-3-662-54345-0%5C_3.

Shelhamer, Evan, Jonathan Long, and Trevor Darrell. 2017. “Fully Convolutional Networks for Semantic Segmentation”. *IEEE Trans. Pattern Anal. Mach. Intell.* 39 (4): 640–651. doi:10.1109/TPAMI.2016.2572683. https://doi.org/10.1109/TPAMI.2016.2572683.

Simonyan, Karen, and Andrew Zisserman. 2015. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.* http://arxiv.org/abs/1409.1556.

Tan, W. R., C. S. Chan, H. E. Aguirre, and K. Tanaka. 2016. “Ceci n’est pas une pipe: A deep convolutional network for fine-art paintings classification”. In *2016 IEEE International Conference on Image Processing (ICIP),* 3703–3707. doi:10.1109/ICIP.2016.7533051.