Assessing the interest of a fine-grain recognition approach for plankton taxonomic classification

Antoine Olivier Salaün - Supervision : Alexandre de Skowronski, Matthieu Salzmann - January 15, 2023

***Abstract*—Plankton population assemblages are restructuring under ocean warming [1]. In this context, Plankton taxonomic classification must be automatised to accelerate the study of ZooPlankton. This paper aims at testing the insterest of a fine-grain recognition approach on plankton classification. Specifically, the DFL-VGG16 architecture [8], known for it performance on fine-grain recognition tasks, is tested on the ZooScanNet dataset [4].**

It achieves a 83.7% top-1 accuracy and fails at outperforming the current state-of-the-art network on this task, ResNet-18, that reaches 85.5% on this dataset. More generaly, the following experiment seem to indicate that learning dsicriminative patch convolutional filters is less performant than learning globally.

I. Introduction

The goal of the following study is to classify plankton according to taxonomic groups using visual data captured by a ZooScan device [4]. The dataset contains 1,433,278 images of variable sizes labelled in 93 classes. It has been collected by Laboratoire d'Océanographie de Villefanche [2].

II. Fundations

A. Understanding the dataset

The dataset contains 1.5 million images classified according to taxa. Taxa are taxonomic groups that can be discriminated based visual information. Even if it is related to spieces, taxas are a different concept. For instance, eggs and adults are labelled sperately.

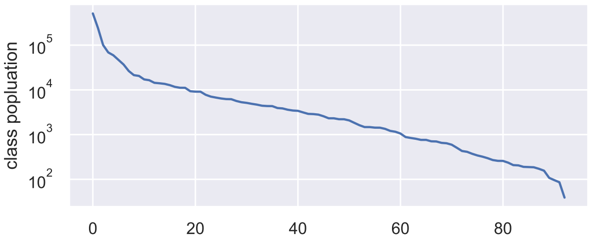


Fig. 1: Distribution of class populations

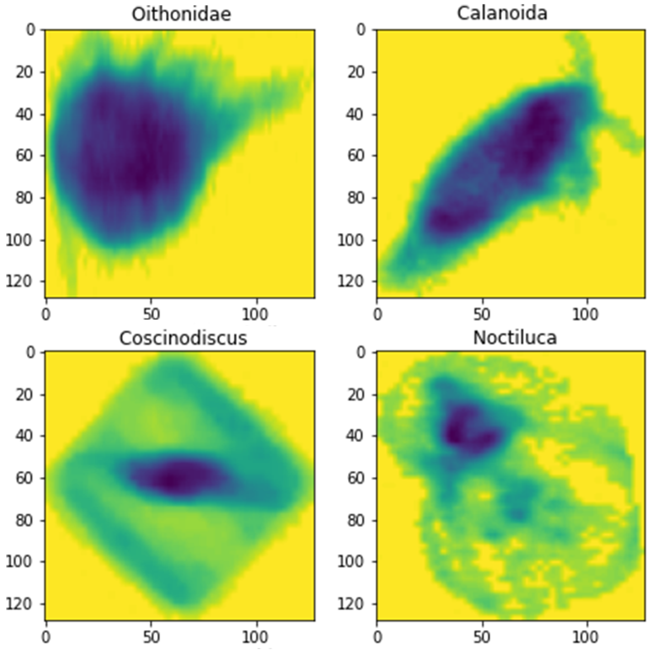


Fig. 2: Samples from ZooSanNet [2]

The dataset is highly imbalanced. Some classes are a than a 1000 time more frequent than others (Fig. 1).

B. Pre-processing

Some classes in the dataset refer to non-living artefacts such as air bubbles or plastic detritus (the most populated class). The following classes are removed to focus the learning on Zooplankton : `seaweed,badfocus_Copepoda,artefact,` `badfocus_artefact,bubble,detritus,` `fiber_detritus,egg_other,multiple_other`

After these deletions, the dataset contains 84 different classes for a total of 703'821images. Then, the images are normalized to a mean of 0.5 and a variance of 0.5. Finally, the images are reshaped to square objects of 128x128 pixels for BaseNet, ResNet18 and ResNet34 and to 448x448 pixels for DFL-VGG16. This implies deformation and the loss of the scaling information. The network architectures are detailed in part III.

C. Architecture and implementation

The training is done with a classical deep learning PyTorch implementation architecture containing 6 classes : Model, Net, PlanktonLoader, Expermient and Runner. All the code related to this paper is available on Gihtub and can be run easily by following the README's instructions : https://github.com/AntoineSalaun/PlanktonTaxaClassifier.

The trainings are run on a EPFL's High Perfomance Computing infrastructure : SCITAS [3] through a SSH protocol.

In general, unless specified, the number of epochs is 20, the loss is Cross-entropy, the dataset is split with 70% for training and 30% for validation and the learning rate varies from 10^{-3} to 10^{-5}. The data is batched and the batch sizes vary depending on the model and on the capacity of the GPU used for training. The model was optimized using ADAM or SGD with variable weight-decays.

III. Models

A. BaseNet

The first model implemented is a basic convolutional neural network. It serves as a baseline in order to compare it with the following network architectures. It is made of 2 layers composed by a convolutional layer, a max pooling and a RELU activation followed by a fully connected layer and a softmax (sigmoïd activation).

In total, it contains 2'067'204 training parameters and it could be trained in 3 hours on recent laptop.

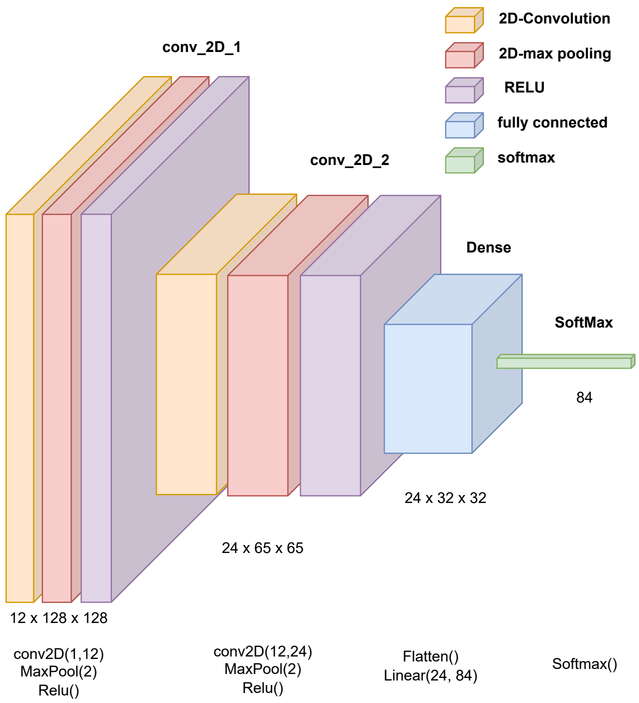


Fig. 3: BaseNet is a basic classification convolutional neural network architecture

B. ResNet-18 and ResNet-34

Residual Neural Networks [7] are a natural choice for image classification. They are historically known for being performant at this task since 2015. It is the most cited Neural Network of the 21st century [6].

It uses skip connections to jump over some layers in order to solve the problem of vanishing gradient on deep neural networks. ResNet contain nonlinearities (ReLU) and batch normalization in between layers. Two ResNets are implemented: ResNet-18 and ResNet-34. They differ by their sizes containing respectively 11.4 and 21.5 million training parameters. They are considered the state-of-the-art architecture for plankton classification [5].

C. Discrimative filter bank applied on a VGG16

Fine-grain object recognition tasks involve distinguishing sub-categories of the same super-category such as classifying cars by brand or birds by species. The state-of-the-art architectures on Fine-grained recognition solutions often use information from localized regions to capture subtle differences in images. Those CNN are able to use sub-patches of the images and asses their discriminative power. Then, it can build mid-level representations of the most discriminative patches and use them as features.

Discriminative Filter Bank (DFL) [8] has the specificity of explicitely learning the discriminative patches in a mid-way Cx1x1 convolutional layer. At this point in the network, it learns C different patches and creates a reponse map that assesses the discrimination power of each patch (Fig. 4)

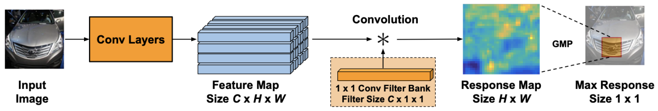


Fig. 4: A discriminative patch can be discovered by convolving the feature map with the 1 × 1 filter and performing Global Max Pooling (GMP) over the response map (figure from [8])

This DLF network (fig. 5) uses a multibranch architecture. The response map is learnt in parallel to a classic CNN architecture. It assists the main CNN learning by giving more or less importance to different parts of the image. Here the CNN used is VGG16. The main branch uses a non-random weight-initialiszazion. The pre-trained weights of VGG16 are used to facilitate the learning process. Morover, VGG16 ability to extract feature is used as VGG is split in two branches.

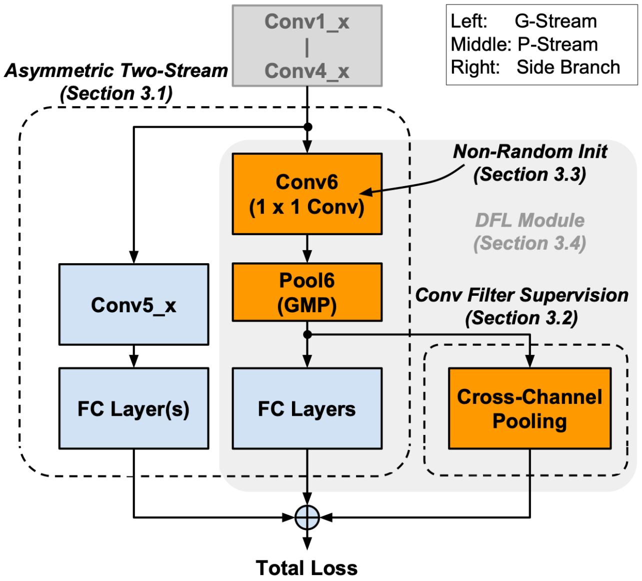


Fig. 5: DFL-VGG16's architecture has three branches. From left to right : G-Stream, P-Stream and Conv Filter Supervision

The architecture (Fig. 5) is divided in three main branches.

1) P-Stream: The middle branch learns the discriminative patches through the 1x1 convolutinal filter (conv6). The filter outputs k class-specific 92x92 patches for each of the 84 class. It has a total of kM channels (where M is the number of classes). k can be changed easily but we will used k=1 beacause of computation ressource limiations (1 filter per class).

The filter is followed by a Global Max Pooling (GMP) over the response map and then a classifier (a Fully Conected Layer and a Softmax Layer). This main branch uses the VGG weights and its job is to :

- filter semantic patches and build the response map
- build patches from the response map
- classify the resulting patches

2) Conv-Filter Supervision: The right branch helps supervise the learning of the 1x1 convolutional filter. Hence, conv6 is not guaranted to target the most discriminative patches. Thus, the cross-Channel Pooling averages the values across every group of k dimensions as the response of a certain class, resulting in an M-dimensional vector.

By feeding the pooled vector into a softmax loss, the conv filter is encouraged to train meaningful discriminative patches for each class. This branch is added with a 0.1 coefficient to the network.

3) G-Stream: The G of G-stream stands for Global. The left-hand branch is a global classifier that is applied on the whole 448x448 images. It is similar to BaseNet in the way that it is a very classic CNN architecture : convolutions, batch normalizations, RELU and average pooling followed by a fully-connected layer.

Finally, the branches are summed (with a 0.1 multiplying coefficient for the Conv-Filter Supervision) and the log-softmax serves as the model's prediction.

IV. Results and Discussion

A. ADAM vs SGD ?

For a proper training, one must choose the right optimizer. ResNet-34 is trained with two optimizers for three different learning rates with a weight decay of 0.2.

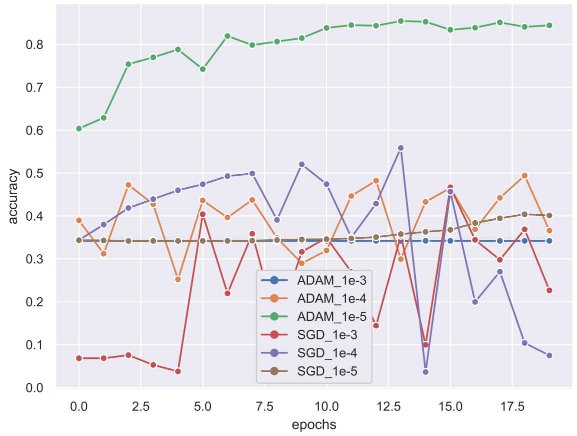


Fig. 6: RenNet34's validation accuracy per epoch for 2 different optimizers (ADAM and SGD) with three different learning rates : lr = 1e-3, 1e-4, 1e-5

Fig. 6 shows the importance of choosing the right optimizer. The model's ability to learn is highly sensitive to both the optimizer and its learning rate.

The tuning of the hyper-parameters shows that **the optimal optimizer for ResNet-18 and ResNet-34 is ADAM with a learning rate of 1e-5**. On the other hand BaseNet and DFL-VGG16 were able to learn using SGD.

B. L2-Regularization

As seen on Fig. 6, the ability of the model to learn the network weights is not simple. Many training of ResNet-18 and ResNet-34 atttained accuracies above 85%. However, their accuracies increased in very chaotic ways with big loss jumps from epochs to epochs. This instability in the trainingcaracterizes a situation of overfitting and justifies the used of L_2-Regularization.

As the model used are implemented from litterature, it was solved by adding a L_2-Regularization. It is done through a weight decay in the optimizer. In order to tune the weight-decay, Fig. 7 shows the top-1 accurcy per epoch for different values of weight decay with a ResNet-18 architecture.

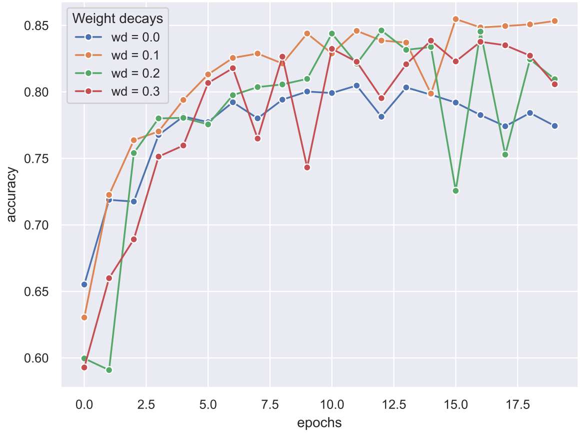


Fig. 7: ResNet18's validation accuracy per epoch for 4 different values of decay using ADAM and a learning rate of 1e-5

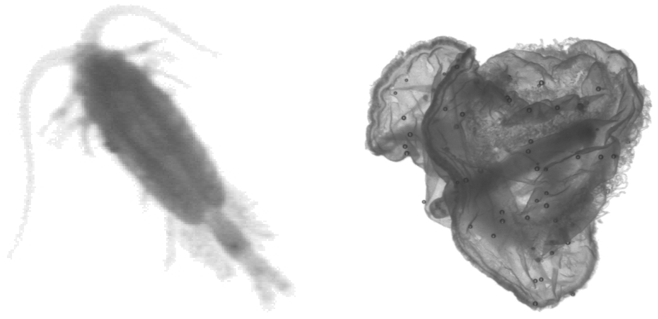
Fig. 7 show that a weight decay lets us achieve the highest performance (85.5%). Without 0 weight decay, the model fails at reaching 80% which could be explained by overfitting. The model learns the dataset instead of actual semantic features. For weight decays of 0.2 and 0.3, the model's accuracy does big jumps from epoch to epochs and starts decreasing after about 10 accuracies. Thus, in additing to achieving the highest top-1 accuracy, a weight decay of 0.1 shows the most stable learning.

The observation made on Fig. 7 is confirmed by the study of train and validation losses presented in the **??** Appendix :

1) The training loss seems to increase after 10 epochs
2) When the weight decay increases this increasing effect strenghtens
3) When the weight decay increases, the validation loss follows more closely the training loss
4) When the weight decay increases, instabilities appear in the validation loss

C. Imbalanced dataset

As the dataset is highly imbalanced (Fig. 8), one could fear that the learning would be limited to the most populated class, especially on smaller models.



(a) Calanoida

(b) Ctenophora

Fig. 8: The most populated class is Calanoida and contains 241’495 images while the least populated class is Ctenophora with only 39 images

In order to check for it, the accuracy is computed on each class for the BaseNet and ploted on Fig. 9.

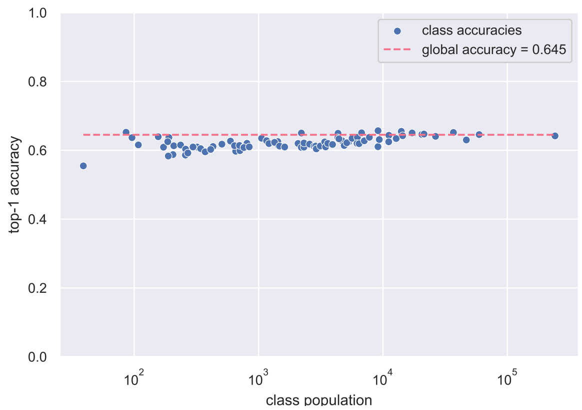


Fig. 9: Class-specific validation accuracies with respect to class popluations for BaseNet.

Fig. 9 shows a slight exponential corelation between class accuracy and class population. However, except for Ctenophora, this effect is not significant in comparaison to the class accuracy standard deviation.

Even for the model with the less trainable weigths, **learning on all classes was not an issue**. For this reason, the cross-entropy loss could be kept throughout all computations. There was no need to switch to weighted cross-entropy.

D. BatchSize

Even using SCITAS’s GPU : Titan V (100 TFlops), computational power was a struggle as the cluster is limited to 12 hours runs. Training 20 epochs of DFL-VGG16 on the full dataset required 50 hours of computation and was done on CVlab’s cluster.

In this context, the choice of BatchSize is crucial. A too small BatchSize slows down computation and training would overflow the time limit of 12 hours. On the other hand, if the batch size was chosen too big, it would overflow the 32 Gb memory limit of the Titan V and raise a CUDA error.

The optimal batch sizes in this configuration are : **256 for BaseNet, 128 for ResNet-18 and ResNet-34, 32 for DFL-VGG16**.

E. Classifying performance

Finally, after being tuned, all four models could be compared on this task. Their top-1 accuracies over 20 epochs is ploted on Fig. 10.

Network	**Batch Size**	**Optim.**	**Learning rate**	**Weight Decay**	**Acc**
BaseNet	256	SGD	1e-3	0	61.5
ResNet-18	128	ADAM	1e-5	0.1	85.5
ResNet-34	128	ADAM	1e-5	0.2	84.4
DFL-VGG16	32	SGD	1e-3	0	83.7

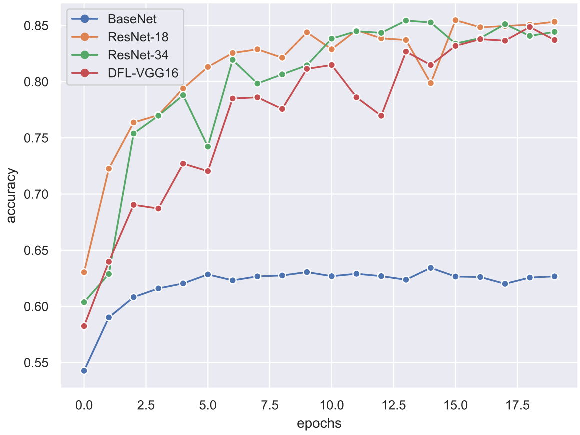


Fig. 10: Validation accuracy per epoch

The best performing model is ResNet-18 with a top-1 validation accuracy of 85.5%. Its big brother ResNet-34 and the dicriminative filter method follow closely. This shows that the discriminative filter bring no additional value to classical RNNs.

The advantage of DFL has been proven on BIRDS and STAFORD CARS [8] but could not be translated to this non-conventional plankton dataset. Hence, its power is in fine-recognition tasks. In Stanford Cars, the convolutional filter learns to target the car logo to discriminate cars based on this fine detail. For birds, it does better than classical RNNs by targeting bird's paws and beaks.

In this dataset, DFL-VGG16 is unable to find highly semantic patches in the images as plankton do not have such discriminative fine features. The classification of plankton is easier by looking at the general aspect of it (its size and gloabl shape). DFL-VGG16 was still able to achieve decent results as it contains the G-stream that learns on the whole image. The contribution of filters did not improve the classification and that explains why it acheived similar results to ResNets.

Another potential explanation for the performance of DFL on BIRDS and STANFORD CARS and ZooScanSet is the following. The patching mechanisms works thanks to the translation equivariance of convolutional neural network. It can learn from a patch wherever it is on the picture. However, in this dataset, each plankton sample is rotated differently and the convolutional network is not rotational-invariant (on contrary to STANFORD CARS where the brand's logo is usually in the same orientation). Thus, the CNN might struggle to learn these semantic patches as they will be percieved differently depending on the angle. It might be interesting to test a rotation-invariant architecture such as Cylindrical Convolution Layers [**CyCNN**] to see if it improves the classification.

Another possible improvement would be to build a network able to task as inputa samples without resizing them. Hence, ZooPlanktons vary a lot in scale and the pre-processing loses this scaling information when it resizes the images to a constant 128x128 size. The scaling information could be very handy for taxonomic classification.

F. *Assessing the importance of pre-processing*

The classification performance of ResNet-18 could be in part be attributed to the decision of eliminating unrelevant classes from the dataset in the pre-processing. Fig. 11 controls for this effect.

Hence, even if, without the preprocessing the dataset is bigger, the accuracy drops from 85.5 % to 80.3%. It can be explained by the lower semantic level of rejected classes such as 'detritus' or 'bad_focus'.

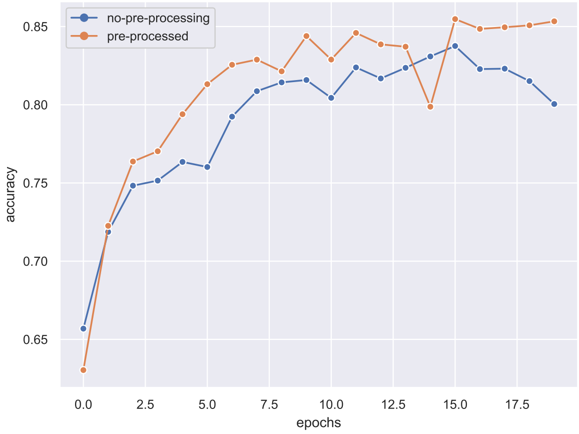


Fig. 11: Validation accuracy per epoch of ResNet-18 with and without pre-processing

V. Summary

To sum it up, the fine-grain recognition approach did not bring value to the classification as the task is not really a fine-recognition task. However, that statement was not obvious and I believe that this should be the main take-away of this study : **Plankton taxonomic classification is not a FineGrain Recognition task and global models seem to perform better than the discrimination of local features.**

The best performing model for plankton classification achieves a top1-accuracy of 97% but is limited to ZooPlankton from a specific lake with only 35 different taxa [5]. Other, state-of-the-art networks for plankton classification achieve accuracies 96% [5]. The most performant model on the ZooScanNet dataset achieves a top-1 accuracy of 90% using ResNet-18 [5].

Even though some of our trainings achieved classification accuracies above 90% at some epochs, our study acheives a stable reproducible validation accuracy of 85.5% using a ResNet-18 architecture with ADAM, a learning rate of 1e-5, a weight decay of 0.1 and cross-entropy as a loss.

References

[1] Vogt M. Elizondo U.H. et al. Benedetti F. *Major restructuring of marine plankton assemblages under global warming.* 2021. DOI: https://doi.org/10.1038/s41467-021-25385-x. URL: https://www.nature.com/articles/s41467-021-25385-x#citeas.

[2] Jalabert Laetitia Olivier Marion Romagnan Jean-Baptiste Costa Brandao Manoela Lombard Fabien Llopis Natalia Courboulès Justine Caray-Counil Louis Serranito Bruno Irisson Jean-Olivier Picheral Marc Gorsky Gaby Stemmann Lars Elineau Amanda Desnos Corinne. *ZooScanNet: plankton images captured with the ZooScan*. 2018. DOI: 10.17882/55741. URL: https://www.seanoe.org/data/00446/55741/.

[3] EPFL. *Scientific IT Application Support*. 2012. URL: https://www.epfl.ch/research/facilities/scitas/.

[4] Hydroptic. *ZooScan: a ZooPlankton sampling device*. 2016. URL: http://www.hydroptic.com/index.php/public/Page/product_item/ZOOSCAN.

[5] Kusworo Adi Esa Prakasa Arief Rachman Ovide Decroly Wisnu Ardhi Tri Retnaningsih Soeprobowati. *Deep Learning Methods for Plankton Identification: A Bibliometric Analysis and General Review*. 2022. DOI: 10.1109/APICS56469.2022.9918707. URL: https://medium.com/mini-distill/effect-of-batch-size-on-training-dynamics-21c14f7a716e.

[6] Jürgen Schmidhuber. *The most cited neural networks all build on work done in my labs*. 2021. DOI: https://doi.org/10.1038/s41467-021-25385-x. URL: https://people.idsia.ch/~juergen/most-cited-neural-nets.html.

[7] Kaiming He; Xiangyu Zhang; Shaoqing Ren; Jian Sun. *Deep Residual Learning for Image Recognition*. 2016. DOI: 10.1109/CVPR.2016.90. URL: https://ieeexplore.ieee.org/document/7780459.

[8] Larry S. Davis Yaming Wang Vlad I. Morariu. *Learning a Discriminative Filter Bank within a CNN for Fine-grained Recognition*. 2018. DOI: 10.1109/CVPR.2018.00436. URL: https://arxiv.org/pdf/1611.09932.pdf.

VI. Appendix

Fig. 7 shows ResNet-18's validation accuracy per epoch for different weight decays using ADAM and a learning rate of 1e-5.

Fig. 12, Fig. 13, Fig. 14, and Fig. 15 refer to the associated losses per epoch and support the observation made in IV-B.

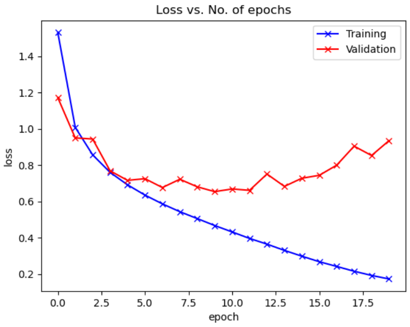


Fig. 12: ResNet-18 with weight decay = 0,0

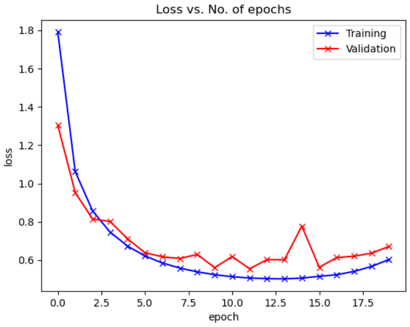


Fig. 13: ResNet-18 with weight decay = 0,1

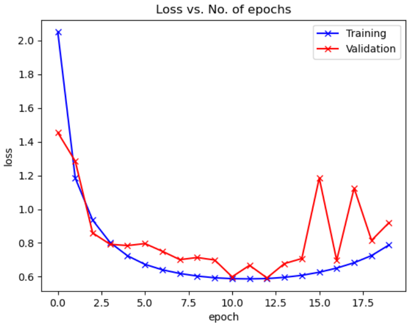


Fig. 14: ResNet-18 with weight decay = 0,2

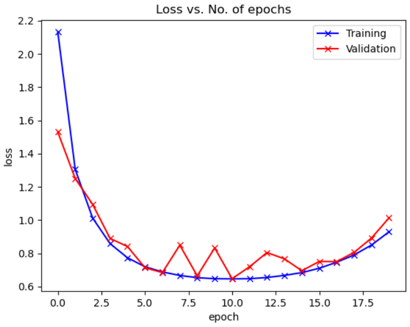


Fig. 15: ResNet-18 with weight decay = 0,3