Testing data augmentation techniques and VGG convolutional neural networks for multiclass image classification on the CIFAR-10 dataset

Julian Cabezas Pena
Assignment 2, Deep Learning Fundamentals, 2020
University of Adelaide. Student ID: a1785086
SA 5005 Australia
julian.cabezaspena@student.adelaide.edu.au

Abstract

In the last years, the field of image classification has been heavily invluenced by the apperance of convolutional neural networks (CNN), that use the technique of convulution to extract data from a small portion of the image, usually called kernel, limiting the number of paramaters compared with traditional neural networks and archiving very good performance. One of the key adspects of the application of a CNN is the inclusion of data augmentation techniques, that can help reduce overfitting and increase performance, spetially in small dataset The VGG architecture of CNN appeared in 2015 and reached the state of the art in the ILSVRC challenge, using 3x3 kernels and from 11 to 19 convolutional layers. In this research, different data augmentation methods were tested using two different versions of the VGG architecture, with the objective of determining whether the inclusion of data augmentation techniques can have a larger or smaller effect that the increase on the deepthness of the neural network. For this purpose, the CIFAR-10 dataset was used to test the methods. The results indicated that the data augmentation methods can have a significant impact in the performance of this model, archieving the best performance using a random horizontal flip together with a random crop of the image, using the VGG19 architecture. The final model reached a accuracy of 88.82% on the test set, outperforming other implementation that do not include data augmentation but being far away from the state of the art, dominated by transfer learning models. This research showed that practitioners should invest significant attention into the application of data augmentation methods, especially in small datasets such as CIFAR-10.

1. Introduction

In the field of machine learning and pattern recognition, classification problems are one of the most common tasks. In these kind of problems, an algorithm takes a vector of data, and assigns it to one or many discrete classes [2]. One of the fields where machine learning algorithms are extensively used is the field of image classification, where rapid advances in the field have been driven by an accelerated development of the web, that each day generates an immense amount of image data. The field of image classification aims to organize imagine data into a limited amount of classes, based on the characteristics or features of the image. [18]

Early approaches to image classification include the Perceptron, that corresponded to a binaly linear classifier [12], or, in the beginning of the century, the use of Support Vector Machine (SVN) Support Vector Machine (SVM) [17] to adress non-linear boundaries between classes and artificial neural networks (ANN) of various kinds [10]. Although these algorithms can produce relatively good results depending on the dataset, in the beginning of the 2010 decade, the paradigm of image classification shifted towards the use of deep convolution networks, that could in some occasions halve the error rate of previous methods [10].

A convolution neural network is tipically defined as a neural network that presents one or more convolutional layers. A convolutional layer is a neuron than cover an adjacent portion of an image, typically groups of 3x3 or 5x5 pixels, transforming the image using a shared set of weights . This technique can cause a major reduction of the number of parameters when comparing it with fully connected neural networks [1]. In general terms, the convolutional neural networks consists in a series of convolution layers followed by activation functions, between the convolutional layers polling layers are used, and in the end a series of fully connected layers, followed by a soft-max function are used to predict the class

The inclusion of the convolutional layers made a variety of architectures of neural networks appear. One of the most popular architectures of deep convolutional networks is the

VGG family of networks (named after the Visual Geometry Group of the University of Oxford), that introduced the use of a series of very small (3x3 pixels) convolutional layers, archiving the state of the art in the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) of 2015 [15]

Altough the complexity and deepness of the neural network is usually a big factor in the performance of the neural network [5], the training of these deep networks usually can produce overfitting, and relies on big quantities of data such as ImageNet, while in many domains of applications this data is not available. To overcome these problems, different data augmentation techniques can be used. These techniques aim to enhance the quality or size of the data contained in the image, partially solving the limited data problem [14]

The objective of this research is to implement two different VGG architectures woth different number of layers (VGG11 and VGG19) to perform multi class classification of images using the CIFAR-10 dataset as study case, these architectures will be tested using easily available and common data augmentation methods to study how these techniques can increase performance and decrease overfitting and training time. A secondary objective of this study is to determine whether the number of layers of a neural network can be more or less important than the correct application of data augmentation techniques in this case.

2. Background

As the CIFAR-10 is a relatively known database, multiple neural network architectures has been tested on it. Liu and Deng [11] applied the VGG16 neural network architecture to the abovementioned dataset, adapting the fully connected layers to match the size (32x32) of the images of this dataset, while at the same time increasing the dropout rate to avoid overfitting, archiving 78.29%, the same network, using batch normalization, archived an accuracy of 91.55%

The current (2020) state-of-the-art for image classification on this dataset is held by Google Research, that implemented a transfer model called Big Transfer (BiT), accomplishing a 99.4% of accuracy. This algorithm consists in the application of a pre-trained model (BiT-L), that was trained on the JFT-300M dataset, consisting in 300 M labelled data, and then applied on the CIFAR-10 dataset for fine-tuning [8]

The most closely related competitors of the VGG could be AlexNet and ResNet, that were developed in 2012 and 2015 respectively. Alexnet consists in eight layers, the first five are convolutional layers, with kernel sizes of 11x11 (first layer), 5x5 (second layer), and 3x3 (third to fifth layer) , that are followed by max pulling layers in the case of the first, second and fifth convolutions, each convolution is activated using a ReLU function, that proved more effectuve than a sigmoid or tanh function. This neural network finishes with three fully connected layers, that output the predicted class [9]. This neural network can be considered the precursor of the VGG architecture, that modified it by using a larger number of convolution layers of kernel size 3x3, and the dropout regularization to avoid overfitting.

On the other hand, the ResNet is the architecture that came after VGG, managing to solve the problem of the gradient vanishing, this probken is recurrent problem in deep neural networks, that refers to the problem that occurs when, thought the process of gradient descent, the weight updates became smallers as the layer is farther from the output layer, due to the repeated derivation of the fuinctions that are involved in the process becoming in each step of the backpropagation [1, 16]. To approach this phenomena, the ResNet presents the concept of residual learning, that implement shortcuts or skipping between blocks of layers (residual blocks), using a identity function, adding the output of the previous layer to the layer ahead of the residual block. In practice this means this neural network can be trained using more layers than what previouwsly presented, implementing ResNets of more than one thousand layers, pushing the limits of was considered depth in CNN and avoiding the vanishing gradients problem [5, 6]

Even though the VGG results are not comparable to more modern neural networks, such as ResNet, that trough residual learning can archive up to 152 layers in the original paper [5], or the above mentioned state of the art transfer learning techniques [8]. VGG stand for its simplicity and for counting with variations of the same structure of different number of convolution layers, that can help approach the tradeoff between data augmentation and number of layers that is one of the objectives of this study.

3. Methods

3.1. CIFAR-10 Dataset

The CIFAR-10 dataset (named after the Canadian Institute of Advanced Research), consists in 60000 images retrieved from online search (using Google or Altavista), and manually picked to only consider photo-realistic images, and that contain only one prominent instance of the object being classified, these images were manually labelled into 10 mutually exclusive classes [10]:

- Airplane
- Automobile ()excluding truck or pickup truck)
- Bird
- Cat
- Deer
- Dog

- Frog
- Horse
- Ship
- Truck (excluding pickup truck)

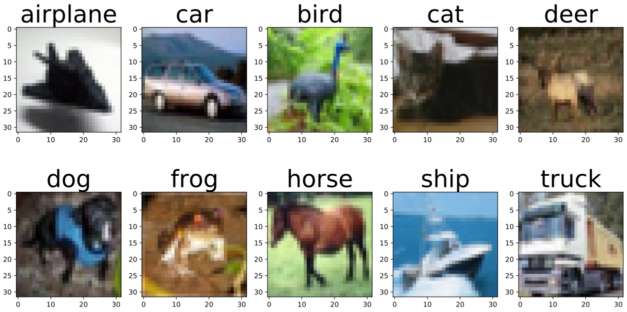

Figure 1. Samples of the ten classes in CIFAR-10 dataset

The database contains 6000 images of each class, that have a resolution of 32 x 32 images . Of the total database, 50000 images (5000 for each class) were considered for the train dataset and the remaining 10000 for the test dataset (1000 for each class)

3.2. VGG11 and VGG19 neural networks

The VGG architectures were introduced in 2015 by the Visual Geometry Group of the University of Oxford. In these architectures, the images are passed to several stacks of convolutional layers of 3x3 kernel size, with a stride of 1 pixel, causing that the spatial resolution of the image is preserved, each convolution layer is followed by a ReLU activation function. After each stack of convolutional layers, a maximum pooling layer with a window size of 2x2 and a stride of 2 is used [15], in practice halving the spatial resolution of the image.

Finally, after all the convolution layers are applied, three linear fully connected layers are applied, using a dropout regularization process in the first two fully connected layers (in the training process). The last of the fully connected layers outputs a number of values equal to the number of classes, finalizing with a softmax function [15], that determines the predicted value looking at the maximum output value.

In Figure 2, it is possible to appretiate that the two VGG architecture share a common structure, with the VGG19 having 8 extra convolution layers.

3.3. Loss function and optimization criterion

To get the predicted value of the neural network, a softmax function was used:

$$Softmax(x_i) = \frac{\exp(f(x_i))}{\sum_{j=1}^{n} \exp(f(x_j))} \tag{1}$$

where x_i is the i output element result of the last fully connected layer of the VGG neural network, as it is divided by the sum of the exponentials of the outputs, the sum of the outputs of the softmax function is always 1 [16].

To calculate the loss of the output, the cross-enthropy function for a set of parameters θ is used [3]:

$$Loss(\theta) = -\sum_{i=1}^{n} y_i log(f(x_i)) \tag{2}$$

where y_i is the one-hot encoded vector of true labels.

This loss function was optimized using a mini batch Stochastic Gradient Descent, as in the original VGG publication [15], using a batch size of 10. The weights of the convolutional layers were initialized using the Kaiming method, that has proven to be useful when dealing with non linear activation functions such as ReLU [4]

3.4. Data Augmentation techniques

As data augmentation techniques are described in literature as very important to avoud overfitting and to take advantage of small datasets[14], such as CIFAR-10, three data augmentation techniques and its combinations were tested on the three architectures:

Random Horizontal Flip: This simple flipping data augmentation technique consists in a 180 degrees flipping over the horizontal axis of the image, it has proven effective in ImageNet and CIFAR-10 [14], and was used by Simonyan and Zisserman [15] in the original formulation of the VGG architecture

Random Clip: Random cropping or clipping can be a very effective way to simulate the effect of a translation or a feature not being in the center [14],, in this case a maximum padding of 4 was selected, and then the image was clipped to preserve the dimensions of the image, filling the padded pixels with a zero value

Color Jitter: A color space transformation was applied to simulate different conditions of brightness and contrast of the image [14], this tool was configured to modify the brightness, contrast and saturation of the image, with multipiers ranging from 0.5 and 1.5.

The only data augmentation that was applied to all the simulations is the normalization of the values of the images using $mean = 0.5$ and $sd = 0.5$, that was applied after the transformation, making the values range from -1 to +1

3.5. Hyperparameter tuning

In order to select the best combinations of data augmentation methods and architectures (VGG11 and VGG19), all the possible combinations of the above mentioned methods

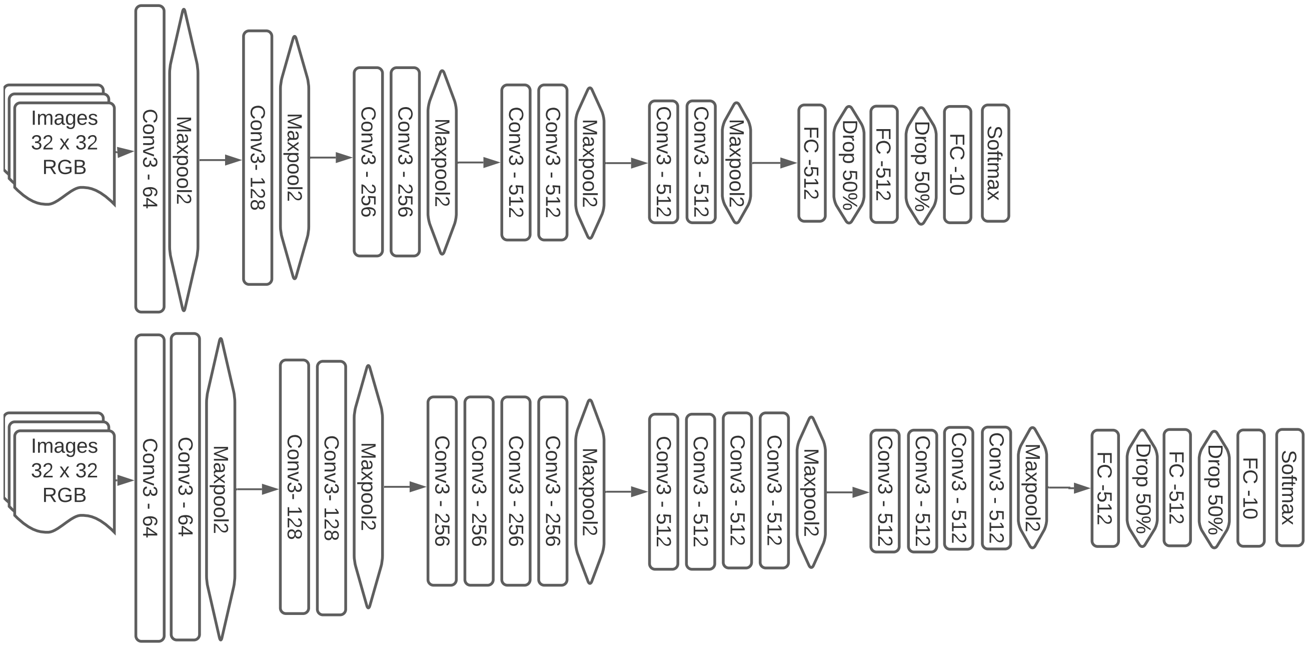

Figure 2. VGG11 (top) and VGG19 (bottom) architectures used in this study, the convolution layer are denoted as conv(kernel size) - (output channels), each convolution layer is followed by a ReLu activation function [15]

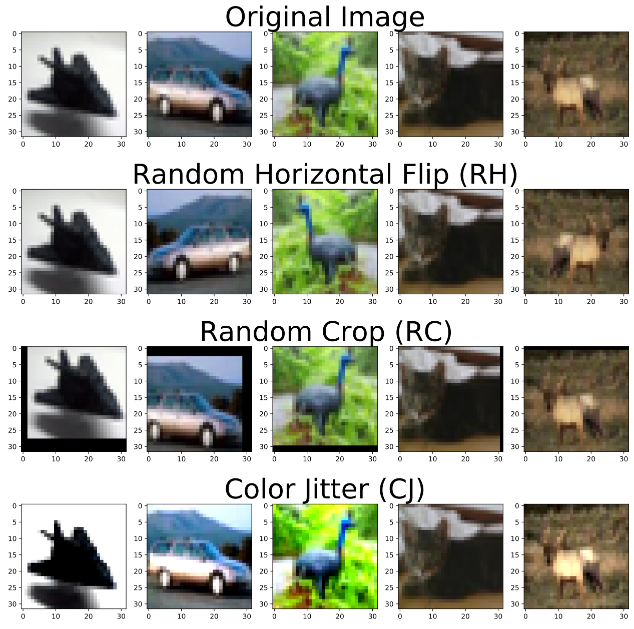

Figure 3. Samples of the ten classes in CIFAR-10 dataset

will be used to train a model in the train set to perform predictions, calculating the accuracy on the train and validation set for each of the epochs. As the number of combinations is high (8 for each architecture), only 50 epochs will be used, and the best combinations will be selected from the one with the maximum accuracy in the validation set.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions} \tag{3}$$

The selected combination of methods will be then trained using 100 epochs to test three different learning rates (0.01, 0.075 and 0.005), selecting this parameter using the higher accuracy in the validation set

4. Code and processing

The code, along with the requirements and setup instructions for the reproduction of the above-mentioned method, can be found in the following GitHub repository: `https://github.com/juliancabezas/Convolutional_Neural_Networks_CIFAR10`

All the processing was done using Google Colab (`https://colab.research.google.com/`) free tier GPU processing capabilities, that include a NVIDIA K80 graphical unit with CUDA support

5. Results and discussion

5.1. Data augmentation

In Figure 4 it is possible to appreciate that the VGG11 implementation without data augmentation techniques tends to generate overfitting early, archiving 99.07% training accuracy in epoch number 36, while at the same time presenting a validation accuracy of 76.87%, that will be close to the maximum validation accuracy (78.26% in epoch 49). On the other side, all the combinations of data augmentation techniques excepting the color jitter transformation help decrease the overfitting effect, while at the same time greatly improving the validation accuracy. The best results are produced with the combination of all three data augmentation techniques, presenting a maximum validation accuracy of 84.99% and with the combination of random horizontal flip (RH) and random crop (RC), with a maximum validation accuracy of 85.56%. Showing that

The behaviour of the accuracy and loss in the case of the VGG19 architecture (Figure 5)follows a similar patter as in the VGG11 implementation, with the version of the neural network without data augmentation reaching 99.08% training accuracy in epoch 40, and a maximum validation accuracy of 81.12%, and showing better validation performance than the one with the color jitter transformation. In this case, the best performance is also accomplished by the combination of the random horizontal flip and the random crop, archieving 86.36% validation accuracy in epoch 17

When comparing the VGG11 and VGG19 implementations without extra data augmentation techniques, is its possible to notice that the increase of convolutional layers generated an increase of around 3% in the maximum validation accuracy (78.26% in VGG11 and 81.12% in VGG19), but, once comparing both neural networks with the random crop and random horizontal flip augmentations, the performances of both algorithms are close, with less than 1% difference in accuracy (85.56% in VGG11 and 86.36% in VGG19). These results show that, in order to increase the performance of the models, choosing the correct data augmentation methods can produce more benefits than the increase of convolution layers, while at the same time being simpler, as the architecture of the network does not have to be modified.

As abovementioned, of the 16 combinations of architecture and data augmentation methods the best result was archived by the VGG19 netural network with random horizontal flip and random crop methods, and it was choosen to make the final model

5.2. Learning rate

The results of the best learning rate for the chosen model showed that in this particular case, the different learning rates that were tested no not produce great increases in performance or training time (Figure 6), with the best performance being archived in the epoch 98 (88.35% validation accuracy)

5.3. Testing of the final model

The VGG19 architecture, trained using 100 epochs and a learning rate of 0.01 with random horizontal flip and random crop data augmentations method, gave the test accuracy results showed in Table

Class	Test Accuracy
Airplane	91.00%
Car	94.30%
Bird	78.60%
Cat	82.10%
Deer	92.80%
Dog	77.40%
Frog	92.10%
Horse	91.20%
Ship	95.50%
Truck	93.20%
Overall accuracy	**88.82%**

Table 1. Model performance in the test data

When comparing the performance of the model with other methods that has been tested on the same data

As the CIFAR-10 is a relatively known database, multiple neural network architectures has been tested on it. Liu & Deng [11] applied the VGG16 neural network architecture to the abovementioned dataset, adapting the fully connected layers to match the size (32x32) of the images of this dataset, while at the same time increasing the dropout rate to avoid overfitting, archiving 78.29%, the same network, using batch normalization, archived an accuracy of 91.55%

Method	Acc.	Reference
BiT-L	99.4%	Kolesnikov *et al* [8]
GPipe	99.4%	Huang *et al* [7]
ResNet-1001	95.38 %	He *et al* [6]
ResNet-164	94.54%	He *et al* [6]
VGG16 with BN	91.55%	Liu & Deng [11]
VGG19-RH-RC	**88.82%**	**This Study**
VGG16	78.29%	Liu & Deng [11]
ResNet50	78.10%	Sharma *et al* [13]
GoogleLeNet	71.67%	Sharma *et al* [13]
AlexNet	36.08%	Sharma *et al* [13]

Table 2. Performance comparison with other studies

6. Conclusion

This research showed the importance of data augmentation, that in this particular case caused more benefits that the increase in the number of convolutional layers. This insight can be particularly important for deep learning practitioners, showing the community that time invested in testing different data augmentation methods can produce better results than designing deeper neural networks, specially considering the fat that the implementation of these methods is quite straightforward.

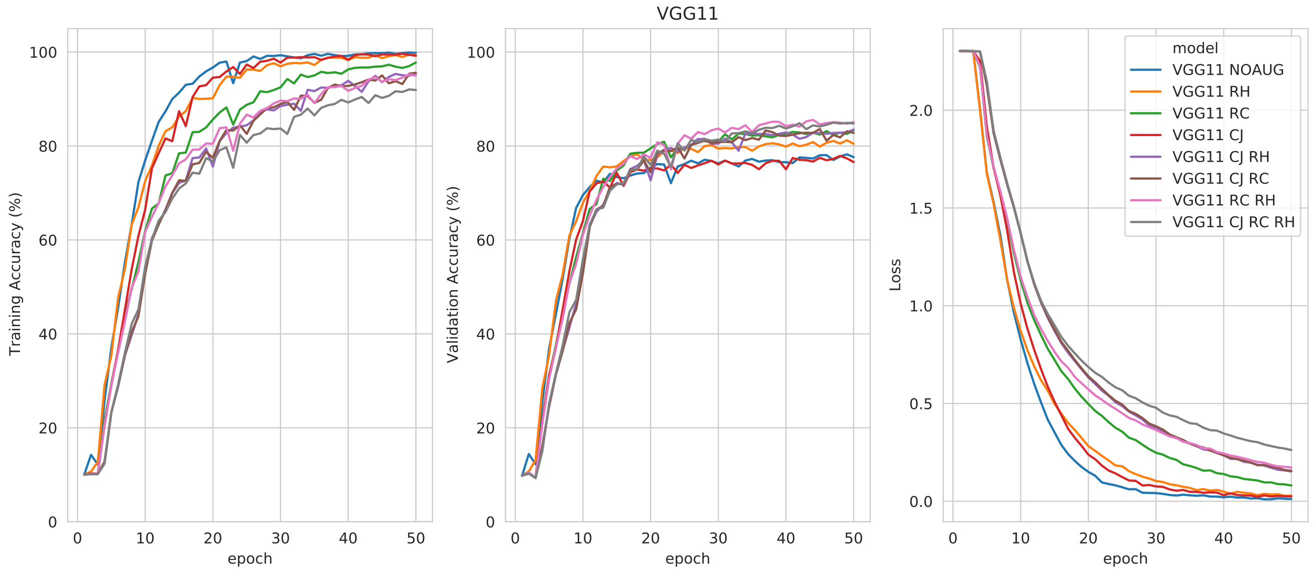

Figure 4. Training accuracy(left), Validation accuracy (centre) and Cross-entropy loss (right) of the VGG11 neural network using different data augmentation techniques. NO AUG: No augmentation, CJ: Color Jitter, RH: Random Horizontal Flip, RC: Random Crop

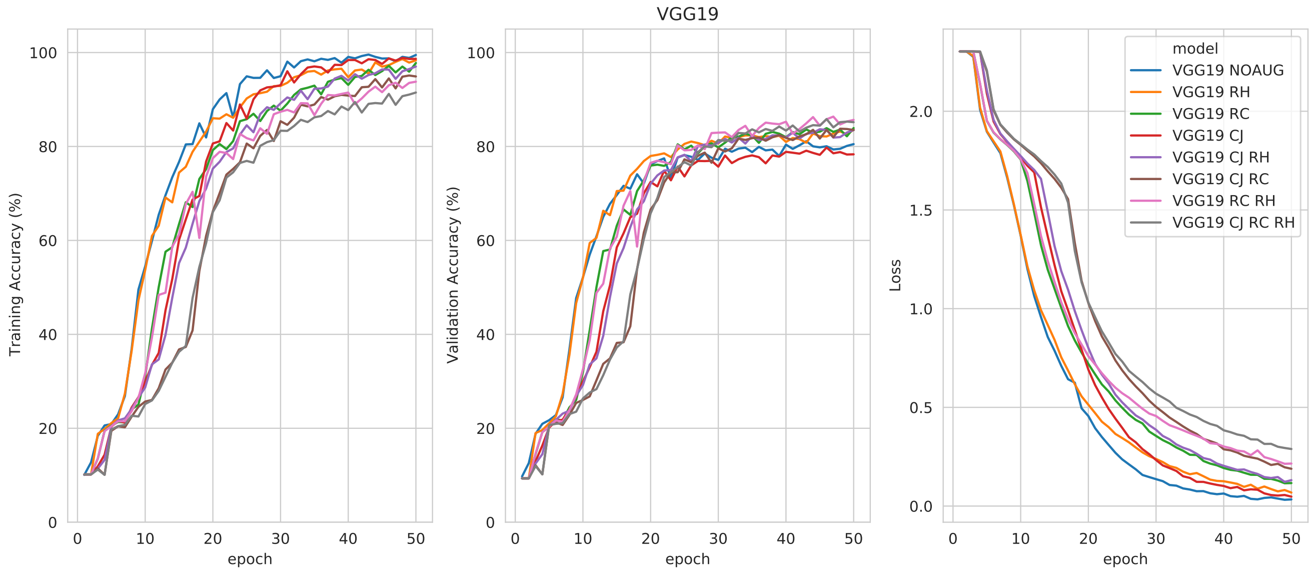

Figure 5. Training accuracy(left), Validation accuracy (centre) and Cross-entropy loss (right) of the VGG19 neural network using different data augmentation techniques. NO AUG: No augmentation, CJ: Color Jitter, RH: Random Horizontal Flip, RC: Random Crop

The VGG architecture of neural networks, despite its relative oldness, was capable of producing relatively acceptable results when the adjusting of hyperparameter and prerpocessing methods is well performed, showing the potential of simple architectures. Altough, despite thjis fact, transfer methods seem to outperform simple CNN implementations showing the trend that deep learning could follow in terms of image classification.

In future research, batch normalizatio n could be used to get better results, as weel as other commonly used data augmentation methods, aditionally, the normalization could be performind using the actual mean and standard deviation of the images in the dataset, proably improving the performance of the VGG models.

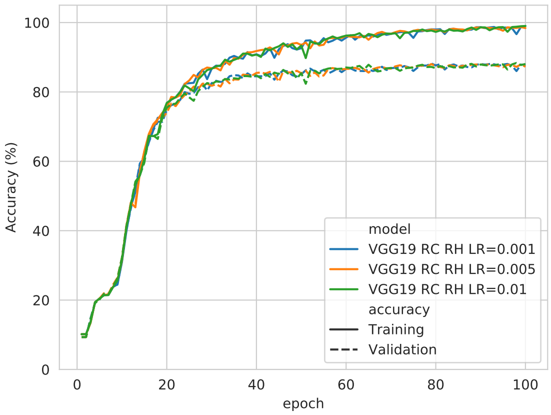

Figure 6. Learning rate tuning on the VGG19 with Random Horizontal Flip and Random Crop model

7. Bonus

This research provided insight over the importance of data augmentation methods, in this case over the increase of the numbe rof layers, using experimental results to back up this hypothesis.

References

[1] Hamed Habibi Aghdam and Elnaz Jahani Heravi. *Guide to Convolutional Neural Networks: A Practical Application to Traffic-Sign Detection and Classification*. 2017.

[2] Christopher M Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, NY, USA, 2006.

[3] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. 2009.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2015 Inter, pages 1026–1034, 2015.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-Decem:770–778, 2016.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. *Computer Vision ECCV 2016. ECCV 2016. Lecture Notes in Computer Science,*, 9908:630–645, 2016.

[7] Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Mia Xu Chen, Dehao Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V. Le, Yonghui Wu, and Zhifeng Chen. GPipe: Efficient Training of Giant Neural Networks using Pipeline Parallelism. *arXiv e-prints*, page arXiv:1811.06965, 2019.

[8] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big Transfer (BiT): General Visual Representation Learning. 2019.

[9] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, Toronto, Canada, 2012.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[11] Shuying Liu and Weihong Deng. Very deep convolutional neural network based image classification using small training sample size. *Proceedings - 3rd IAPR Asian Conference on Pattern Recognition, ACPR 2015*, pages 730–734, 2016.

[12] F. Rosenblatt. The Perceptron - A Perceiving and Recognizing Automaton, 1957.

[13] Neha Sharma, Vibhor Jain, and Anju Mishra. An Analysis of Convolutional Neural Networks for Image Classification. *Procedia Computer Science*, 132:377–384, 2018.

[14] Connor Shorten and Taghi M. Khoshgoftaar. A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, 6(1), 2019.

[15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pages 1–14, 2015.

[16] Sandro Skansi. *Introduction to Deep Learning: From Logical Calculus to Artificial Intelligence*. Springer, Cham, Switzerland, 2018.

[17] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, NY, USA, 1995.

[18] Dengsheng Zhang. *Fundamentals of Image Data Mining*. 2019.