Impact among Convolutional Layers

Lucas Lamy[1], Paulo Henrique Siqueira[2]

[1,2]*Programa de Pós-Graduação em Métodos Numéricos em Engenharia (PPGMNE)*
Universidade Federal do Paraná (UFPR), Curitiba - PR, Brasil (lamy@ufpr.br)

Abstract: Convolutional neural networks have a significant role in image classification. However, finding the best set of hyperparameters is a challenging task. In this paper, we evaluated the impact of a convolution layer on other convolution layers over a variety of hyperparameters by the outcome of Taguchi's experimental table and graphical resources. As a result, the second and third convolutional layers have more interactions than other layers playing an essential role in the network.

Keywords: Hyperparameter optimization; Convolutional layer; Convolutional neural network; Image classification; Deep learning.

INTRODUCTION

Convolutional neural networks (CNNs) are one of the most fundamental artificial neural networks (ANNs) currently used. Since their fundamental ideas - which appeared in research on brain functions (Hubel and Wiesel, 1959, 1962, 1968), then their first application in digit recognition (LeCun et al., 1989) - CNNs are important because of their ability to deal with a large amount of information and solve complex problems that classic ANNs do not solve.

Especially for image classification, CNNs were a breakthrough in the field of ANNs. The fast development took place after AlexNet (Krizhevsky et al., 2017) on Image Large Scale Visual Recognition Challenge (ILSVRC) – with the first deep neural network (DNN), which achieved an error rate below 16% on the ImageNet dataset (Jia Deng et al., 2009). Since then, the architecture of CNN has grown over time, becoming deeper with ZFNet (Zeiler and Fergus, 2013), VGGNet (Simonyan and Zisserman, 2014), and even reaching a milestone of recognition better than humans with ResNet-152 (He et al., 2015a). Throughout this progress, CNNs have evolved for more complex architectures and different tasks, such as medical issues (Erdenebayar et al., 2019; Li et al., 2014), object detection (Girshick, 2015; Girshick et al., 2014; Ren et al., 2017), face recognition (Taigman et al., 2014), fake news detection (Monti et al., 2019), wind forecasting (Harbola and Coors, 2019) and many other applications.

Despite remarkable achievements, there are still CNN optimizations to be done. This paper aimed to understand how one convolution layer affects other convolution layers and how the hyperparameters affect the outcome of CNN. For this purpose, the following hyperparameters were studied: Filter size, Number of filters, Learning rate, Momentum, Batch size, and Activation function. The experimental methodology of Taguchi was used, which enables finding the best combination of hyperparameters in a faster way than the traditional full factorial experiments. The CNN was evaluated by observing the network's accuracy in two datasets and four architectures. The results were analyzed using the output of the design of experiment (DOE), main effects plots, interactions plots, analysis of mean (ANOM), and heatmaps.
In this way, our paper can be summarized in the following key contributions:

- The use of an experimental method for engineering in CNNs.
- An interpretation of the network's hyperparameters by graphical techniques.
- A new analysis for CNNs, looking at the effects between the hidden layers.

RELATED RESEARCH

An essential aspect of CNNs is the ability to extract small features from input and propagate them toward the deeper layers; thus, the extracted features' complexity increases with the network's depth. For instance, in face detection (Albawi et al., 2017), basic characteristics, such as straight lines, can be detected

in the first layers, whereas the deeper layers recognize more complex forms such as an eye. Two aspects affect network performance in this process: the set of hyperparameters and the architecture. A network can extract the features from input and generalize the dataset properly depending on the settings chosen for both aspects.
The relation among depth, the necessity of fewer or more nodes in a Fully connected (FC) layer, and their influence on network errors were explored by Basha et al. (2019) using CIFAR-10 and CIFAR-100 (Krizhevsky, 2009), Tiny ImageNet (Wu et al., 2017) and CRCHistoPhenotypes (Sirinukunwattana et al., 2016). The results presented that shallow CNNs require more nodes in FC layers than deeper CNNs, and wider datasets need more FC layers and are better with shallow architecture; on the other hand, deeper datasets are better with deep architectures and need fewer FC layers.

One of the hyperparameters related to the performance of CNNs is the Batch size. Using The Mixed National Institute of Standards and Technology (MNIST) (LeCun Yann et al., 2012) and CIFAR-10 datasets (Krizhevsky, 2009), the research of Radiuk (2018) was based on well-known architectures of the literature - such as Lanet-5 (Lecun et al., 1998) - defined a range of values for Batch size to evaluate the influence on accuracy. As a result, the accuracy increases as the Batch size grows.

Regarding the Activation function, there are many options: Linear, Sigmoidal, Tanh, ReLU (Nair and Hinton, 2010), PReLU (He et al., 2015b), Leaky ReLU (Maas et al., 2013), and ELU (Clevert et al., 2016). According to Li et al. (2021), the Linear activation functions are the worst possibility. The accuracy of the ELU function is slightly better than that of ReLU, Leaky ReLU, and PReLU, but the functions can present different behaviors depending on the dataset used. Using other CNNs, Xu et al. (2015) found that the PReLU loss is always the lowest and introduced the RReLU function, an efficient way to combat overfitting.

To construct a better architecture, Maitra et al. (2018) studied the impact of Filter size, Number of filters, and Activation functions for the Diabetic Retinopathy dataset (Trivino et al., 2018). They showed that a small number of filters performed better than a large number of them, and filters of small size produced higher accuracy. Additionally, Activation function does not have a significant impact on the network. On the other hand, the work of Khanday and Dadvandipour (2020) using the MNIST dataset and Ahmed and Karim (2020) using the KTH dataset (Schuldt et al., 2004) reach better performance with a higher number of filters in comparison to a small number of filters but agree that smaller filters are better than larger ones.

This way, hyperparameters are a fundamental part of any ANN since they guide the network answer and the learning process through the training step. However, finding the best set of hyperparameters is quite challenging. For example, if we have 5 hyperparameters, each with 10 values, the running procedure must be executed 100 thousand times to find an appropriate set of values (Aggarwal, 2018). Moreover, as mentioned in Maitra et al. (2018), Khanday and Dadvandipour (2020), and Ahmed and Karim (2020), when the hyperparameters increase, the runtime also grows; depending on the settings chosen, a deep architecture or an extensive range of hyperparameters, sometimes it is impossible to run all trials because of the amount of time involved in training.

Many methods have been developed to find the best set of hyperparameters for a CNN. The most common are greedy search and random search. In addition, some methods can use other kinds of algorithms to help these search, such as genetic algorithms (Loussaief and Abdelkrim, 2018; Aszemi and Dominic, 2019), evolutionary algorithms Bochinski et al. (2017), metaheuristics of swarms (Bacanin et al., 2020; Serizawa and Fujita, 2020), deterministic RBF surrogates (Ilievski et al., 2017) or even mesh adaptive direct search (Lakhmiri et al., 2019).

One greedy method is to test all possibilities for a determined set of hyperparameters, but it can be impossible due to the time needed. However, the experimental tables of Taguchi (Zhang et al., 2020) can reduce the number of runs to find the best hyperparameter combination. This robust method uses orthogonal arrays (Taguchi and Konishi, 1987) and is widely used in other applications (Lamy and Chaves Neto, 2017; Miloradović et al., 2019; Tsiolikas et al., 2017) and can also be helpful in CNNs (Akbarzadeh et al., 2019).

MATERIALS AND METHODS

Once the CNN is built by layers, each layer has the decode function according to which part that layer belongs and propagates the previous result to the consecutive layers. Therefore, a technique that visualizes the whole neural network and how the

layers interact can provide powerful insights. This section provides the proposed optimization method, describes the architectures of CNNs, the orthogonal arrays for DOE, and the interpretation method for the outcome of the experimental design of Taguchi.

Architectures

Four different architectures were chosen, as shown in Figure 1. All architectures dismiss a high-end CPU; since the objective is the CNN optimization, evaluating the method in an award-winning CNN architecture - which is already efficient and well-optimized – does not match to desired results.

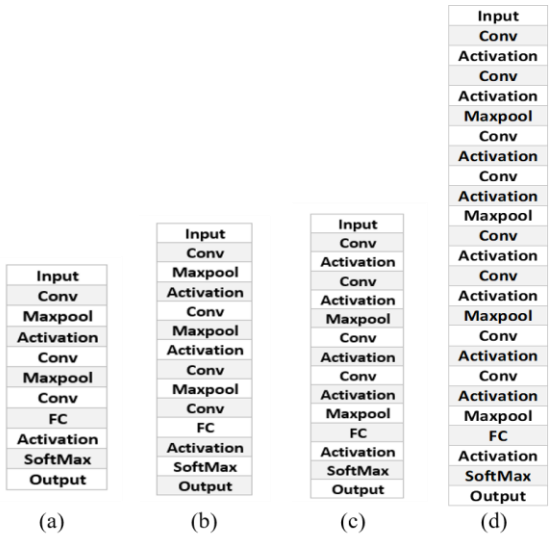

Fig. 1. An overview of all architectures.

Legend: The architectures use convolutional layers (conv), Activation functions (Activation), Max pooling layers (Maxpool), Fully connected layers (FC), and a SoftMax layer (SoftMax). (a) Architecture of Model 1. (b) Architecture of Model 2. (c) Architecture of Model 3. (d) Architecture of Model 4.

The first model is based on LaNet-5 (Lecun et al., 1998); the architecture uses a Max pooling layer after the convolutional layer, except for the last convolution layer, which has an FC layer after it. Model 2 is an expansion of Model 1 by adding one convolutional layer, following the same pattern of layers. Model 3 explores the structure of more than one convolutional layer in sequence, i.e., the Max pooling layer comes after two convolutional layers. The last, Model 4, follows the same pattern as Model 3 but has double the convolutional layers. All models at the end have a SoftMax Function.

Design of Experiment

The purpose of DOE is to investigate the significance of the factors and explain the relationship between them. In other words, it determines the influence of each factor on the response and evaluates different experimental configurations. For CNN, the factors are the hyperparameters, and the levels are their different values.

The DOE of Taguchi is less expensive than the full factorial design because it uses orthogonal arrays, i.e., the columns – the factors – are independent and balanced for the levels. These properties lead to only a fraction of the runs without reducing effectiveness (Taguchi et al., 2004). However, the computational cost must be considered when applying Taguchi's design for the CNN since the cost increases with the number of factors and levels.

The analyzed factors in this paper were Filter size (Fs), Number of filters (Nf), Learning rate, Momentum, Batch size, and Activation function. Fs varies from 3x3 to 7x7, while the number of filters ranges from 16 to 1,024, depending on the model. Two values were set for Learning rate (0.001 and 0.0001) and Momentum (0.8 and 0.9). For Batch size, values of 128 or 256 were set. Finally, Activation function was chosen as ReLU and Tanh.

The levels of factors used in each experimental case are described in Table I. The hyperparameters related to each layer are Fs *i* and Nf *i*, where *i* is the number of convolutional layers. For practical reasons, the hyperparameters: Learning rate, Momentum, Batch size, and Activation function are sometimes called method hyperparameters.

All experimental tables[1] (Kacker et al., 1991) have 2 levels for each factor - Model 1 uses the L_{12} orthogonal array, Model 2 and Model 3 use the L_{16}, and Model 4 uses the L_{32}.

Analysis

ANOM is a statistical technique used to classify DOE factors. This resource, through the difference of mean obtained for each factor-level analyzed, ranks the factors in order of importance; a higher difference between levels has high relevance, and a factor that presents no difference between your levels is not significant. This way, it is possible to choose the hyperparameter that has more impact on the experiment. The main effect plot represents the factors and the differences between their levels.

[1] All experimental tables and algorithms can be found at https://github.com/lamyluc/Impact-among-Convolution-Layers.git.

TABLE I. FACTOR-LEVEL OF HYPERPARAMETERS FOR MODELS 1 TO 4

Hyper-parameter	Model 1		Model 2 and 3		Model 4	
	Level 1	Level 2	Level 1	Level 2	Level 1	Level 2
Fs 1	3x3	7x7	3x3	7x7	3x3	7x7
Nf 1	16	32	32	64	32	128
Fs 2	3x3	7x7	3x3	7x7	3x3	7x7
Nf 2	32	64	64	128	32	128
Fs 3	3x3	7x7	3x3	7x7	3x3	7x7
Nf 3	64	128	128	256	128	256
Fs 4			3x3	7x7	3x3	7x7
Nf 4					128	256
Fs 5					3x3	7x7
Nf 5					256	512
Fs 6					3x3	7x7
Nf 6					256	512
Fs 7					3x3	7x7
Nf 7					512	1,024
Fs 8					3x3	7x7
Nf 8					512	1,024
Learning rate	0.001	0.0001	0.001	0.0001	0.001	0.0001
Momentum	0.8	0.9	0.8	0.9	0.8	0.9
Batch size	128	256	128	256	128	256
Activation function	ReLU	Tanh	ReLU	Tanh	ReLU	Tanh

It is usual to analyze one factor alone, but it is also possible to study the interaction between the factors. As Fowlkes and Creveling (1995) and Taguchi et al. (2004) describe, there can be 3 types of interaction: strong, mild, and no interaction. Any kind of interaction is hard to classify – mainly mild interactions - and it is done visually with an interaction plot and numerically with the same procedure of ANOM. A small difference in ANOM, i.e., a slight inclination between factors, can be interpreted as greater or less importance guiding a misleading evaluation.

To better understand interactions and measure them precisely, the following procedure was made: first, an interaction graph was plotted, followed by ANOM of the same interactions without a hierarchy of importance; lastly, the previous numerical results were plotted as a heatmap.

The transformation of the interaction plot in a heatmap (Figure 2) takes an overview of all interactions, removes the interpretation directed to only one interaction, allows an easier interpretation, and eliminates the deceiving classifications where slight differences exist from each other, leading to the understanding of the whole network.

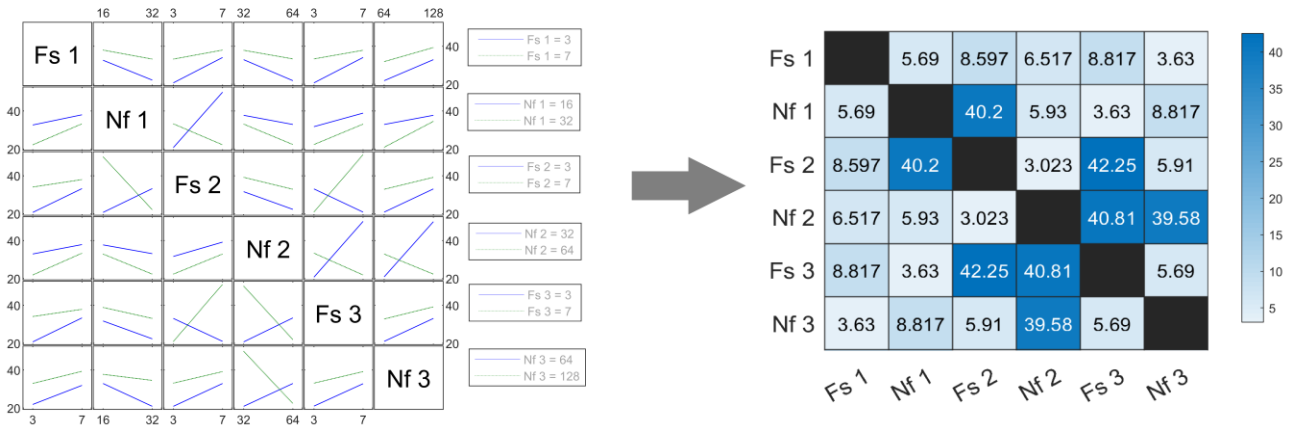

Figure. 2. Transformation of the interaction plot (left) into a heatmap (right).

Legend: Left: left and right axis are accuracy of CNN; top and bottom axis are column factors levels. Right: Strong interactions are in dark blue, while mild interactions are in light blue; the left and bottom axis are the factors; the right bar is the intensity of the interaction.

Datasets and System Specification

The CIFAR-10 (Krizhevsky, 2009) and MNIST (LeCun Yann et al., 2012) datasets were chosen for this paper. These datasets are globally applied for evaluating different machine learning techniques, and they are simple datasets, allowing easy reproduction of the results.

The MNIST dataset is a black and white handwritten digit with a set of 60 thousand examples for training and 10 thousand examples for testing. The set comprises 28x28-pixel images, and the classes are numbers from 0 to 9 with their respective labels. CIFAR-10 is a color dataset with 60 thousand images, each measuring 32x32 pixels, separable into 10 different categories where each class has 6 thousand images; its standard form is 50 thousand images for training and 10 thousand for testing.

All experiments were implemented in a single NVIDIA GeForce® RTX 2070 SUPER (8 GB) GPU – algorithm available[1]. All models were trained with stochastic gradient descent (SGD) with Momentum by 30 epochs for the MNIST dataset and 100 epochs for the CIFAR-10 dataset. Training used early-stop of 5 epochs.

RESULTS AND DISCUSSION

The analysis is performed by the output of the experimental table and the heatmaps. All results[1] were measured in training accuracy, training loss, validation accuracy, and validation loss. The following results are based only on validation accuracy. The graphs have been suppressed, and the results are described below.

For MNIST dataset results[1], validation accuracy on the experimental table can change between 9.8% and 98.89%. Models 1 and 2 present a significant variance in the results; half of the running reaches over 85% accuracy, and the better results have the Learning rate set to 0.001 and Tanh. Model 3 presents the highest validation accuracy and is the model that shows the lowest variance in results; additionally, it has higher outputs with Learning rate of 0.001. Model 4 can perform better with the same hyperparameters as Models 1 and 2. The higher accuracies of Models 2, 3, and 4 have Fs 7x7, and Nf increases, or remains the same, in the following layer.

The experimental result[1] of the CIFAR-10 dataset shows that the top results on Models 2, 3, and 4 present better output with ReLU function. For Models 3 and 4, the worst result used a Learning rate of 0.0001 and Tanh; on the other hand, the best results of Model 2 used Learning rate of 0.0001. In the same way as the MNIST dataset, Nf must increase or remain the same in the following layer for a better CNN output.

The main effect plot[1] shows a network overview pointing to the best level of performance. The results are summarized in Table II; the best value for each hyperparameter is described from the first to the last layer. In general, Models 1 and 2 obtained a higher mean validation accuracy using Tanh and a Learning rate of 0.0001, while Models 3 and 4 performed better with the ReLU Activation function and Learning rate of 0.001. In addition, the MNIST dataset answered better with Fs 7x7, except for the last layers of Model 4.

TABLE II. BEST FACTOR LEVEL FOR EACH MODEL/DATASET BY LAYER.

		MNIST	**CIFAR-10**
Model 1	Fs	7, 7, 7	3, 3, 7
	Nf	16, 32, 128	32, 64, 64
	Learning rate	0.0001	0.0001
	Momentum	0.8	0.9
	Batch Size	128	128
	Activation Function	Tanh	Tanh
Model 2	Fs	7, 7, 7, 7	3, 3, 7, 3
	Nf	32, 64, 128, 128	64, 128, 128, 512
	Learning rate	0.0001	0.0001
	Momentum	0.8	0.8
	Batch Size	256	128
	Activation Function	Tanh	Tanh
Model 3	Fs	7, 7, 7, 7	3, 3, 7, 3
	Nf	64, 128, 256, 256	64, 128, 128, 256
	Learning rate	0.001	0.001
	Momentum	0.8	0.8
	Batch Size	128	128
	Activation Function	ReLU	ReLU
Model 4	Fs	7, 7, 7, 7, 7, 7, 3, 3	7, 3, 7, 7, 7, 7, 3, 3
	Nf	128, 128, 128, 128, 256, 256 1025, 512	128, 128, 128, 128, 256, 512, 1024, 512
	Learning rate	0.001	0.001
	Momentum	0.9	0.9
	Batch Size	128	128
	Activation Function	ReLU	ReLU

Meanwhile, the CIFAR-10 dataset presented higher accuracy using 3x3 Fs for Models 1, 2, and 3. Model 4 showed better results, with 7x7 Fs. Batch size of 128 was the best in almost all models and datasets — the exception was Model 2 for the MNIST dataset. The best Momentum value for Models 2 and 3 is 0.8 for both datasets; in the same way, Model 4 for a value of 0.9; Model 1 performed differently over the datasets. The Nf does not show a possible pattern.

The ANOM results are summarized in Table III. For the common hyperparameters to all models, it is noticed that Activation function is the most important factor for more than half of the models; when it is not the most relevant, it is in the top three. The Learning rate is in the following position of importance, except

for Model 1. Batch size also proved to be an important factor but with less impact than Activation function and Learning rate. Analyzing only Nf and Fs, Fs 3 is the most significant hyperparameter in the networks, followed by Nf 1 and Fs2. The other hyperparameters show a minor influence when compared to other factors.

Observing the interactions between Nf and Nf, heatmaps[1] show that Nf 2 has more interactions than other layers and interacts mainly with Nf 3 and 4; Nf 1 has fewer interactions than the other Nfs. For the interactions Fs by Fs, Fs 1 is more interactive than the other factors, followed by Fs 2. The interactions of Fs and Nf on the method's hyperparameters, the heatmap shows that the Learning rate and Batch size especially affect Nf 1, the Momentum affects Nf 3 and Fs 2, the Learning rate and Activation function affect Fs 3, and the Batch size affects Fs 4.

Overall, Model 1 is more reactive among all models and in itself. Model 2 has a structure that is similar to Model 3, but Model 3 presents a better performance; Model 4 has an outcome comparable to Model 3, but the factors of layers 5 to 8 showed no strong interactions.

CONCLUSION

In this paper, we propose to analyze the impact of a convolutional layer on other convolution layers and their behavior concerning the hyperparameters Nf, Fs, Learning rate, Momentum, Batch size, and Activation function. This analysis allowed us to achieve two goals. First, guide the initial choice of these hyperparameters because this is usually defined by the researcher's experience on the subject. The second is introducing a method of optimization and analysis for CNNs.

The first goal is partially done using the output of DOE, ANOM, and the main effects plot. The experimental table alone provides insight into the behavior of CNNs, but it is an overall result. While the mean effect plot shows the best level for each factor, together with ANOM, it is a powerful tool for identifying the best set of hyperparameters. However, none of the previous techniques are concerned with how each hyperparameter affects the other layers.

The interaction map presents the view among factors necessary for the complete understanding of CNN, but even with a small number of factors/levels, the result can be confused. The translation of interaction maps into a heatmap simplified the reading and identifying of the interactions among layers; in this way, it was possible to find optimizations and solutions for CNNs, achieving the second goal.

The results of this research showed that Model 3 is the best choice for architecture since it has the best results and is deep enough for the task of classification with both datasets — in contrast to Model 4, which has doubled its size and requires more computational work to reach a similar result. Models 1 and 2 can achieve a good response, even though they present a significant variation in results — which is different from the slight variation of Models 3 and 4.

TABLE III. HIERARQUICAL IMPORTANCE FROM ANOM

	Model 1		**Model 2**		**Model 3**		**Model 4**	
	MNIST	**CIFAR-10**	**MNIST**	**CIFAR-10**	**MNIST**	**CIFAR-10**	**MNIST**	**CIFAR-10**
Learning rate	10	10	3	2	1	2	1	2
Momentum	8	4	10	6	11	8	10	9
Batch Size	2	2	12	5	4	3	4	4
Activation Function	1	1	1	3	3	1	2	1
Fs 1	7	7	4	7	8	12	9	11
Nf 1	4	9	11	4	6	5	6	5
Fs 2	9	5	5	8	10	6	12	8
Nf 2	3	8	8	9	7	7	8	15
Fs 3	5	6	2	1	2	4	3	3
Nf 3			6	12	12	11	15	16
Fs 4			9	10	5	9	5	6
Nf 4			7	11	9	10	14	18
Fs 5							18	19
Nf 5							13	13
Fs 6							11	10
Nf 6							16	20
Fs 7							17	7
Nf 7							19	17
Fs 8							20	14
Nf 8							9	12

A Learning rate of 0.0001 and Tanh as Activation function yield better responses in architectures that have one convolution layer followed by Max pooling layer and Activation function; on the other hand, a Learning rate of 0.001 and ReLU function work better with architectures that have a convolutional layer followed by sequence of Activation function, convolutional layer, Activation function, and Max pooling layer. Small models have higher accuracy with a Momentum value of 0.8, whereas a Momentum of 0.9 is better for bigger architectures. The best choice for Batch size is 128.

The MNIST dataset is better with 7x7 Fs, and the CIFAR-10 dataset had a good response with both sizes. Regarding Nf, the rule is double the quantity in relation to previous layers or retain the same.

Finally, heatmaps display that the third layer is the most critical and interactive related to the method's hyperparameters – changes related to this layer should be well assessed since any modification can affect the accuracy of CNN. The first and fourth layers strongly interact with Batch size – as long as the CPU memory is available, it is recommended to set up as large as possible - and the second layer interacts more with Momentum. Meanwhile, the second layer showed strong interaction among all layers related to Nf and Fs, followed by the fourth layer; it is recommended to consult ANOM before any changes.

ACKNOWLEDGEMENTS

This work would not have been possible without the financial support Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES). We thank all of those with whom we worked during this project.

REFERENCES

AGGARWAL, C. C. Neural Networks and Deep Learning. Cham: Springer International Publishing, 2018.

AHMED, W. S.; KARIM, A. AMIR A. The Impact of Filter Size and Number of Filters on Classification Accuracy in CNN. 2020 International Conference on Computer Science and Software Engineering (CSASE). Anais... . p.88–93, 2020. Duhok, Iraq: IEEE. Disponível em: <https://ieeexplore.ieee.org/document/9142089/>. .

AKBARZADEH, S.; AHDEROM, S.; ALAMEH, K. A statistical approach to provide explainable convolutional neural network parameter optimization. International Journal of Computational Intelligence Systems, v. 12, n. 2, p. 1635–1648, 2019.

ALBAWI, S.; MOHAMMED, T. A.; AL-ZAWI, S. Understanding of a convolutional neural network. 2017 International Conference on Engineering and Technology (ICET). Anais... . p.1–6, 2017. IEEE.

ASZEMI, N. M.; DOMINIC, P. D. D. Hyperparameter optimization in convolutional neural network using genetic algorithms. International Journal of Advanced Computer Science and Applications, v. 10, n. 6, p. 269–278, 2019.

BACANIN, N.; BEZDAN, T.; TUBA, E.; STRUMBERGER, I.; TUBA, M. Optimizing Convolutional Neural Network Hyperparameters by Enhanced Swarm Intelligence Metaheuristics. Algorithms, v. 13, n. 3, p. 67, 2020.

BASHA, S. H. S.; DUBEY, S. R.; PULABAIGARI, V.; MUKHERJEE, S. Impact of fully connected layers on performance of convolutional neural networks for image classification. Neurocomputing, v. 378, p. 112–119, 2019.

BOCHINSKI, E.; SENST, T.; SIKORA, T. Hyper-parameter optimization for convolutional neural network committees based on evolutionary algorithms. 2017 IEEE International Conference on Image Processing (ICIP). Anais... . p.3924–3928, 2017. IEEE.

CLEVERT, D. A.; UNTERTHINER, T.; HOCHREITER, S. Fast and accurate deep network learning by exponential linear units (ELUs), 2016.

ERDENEBAYAR, U.; KIM, H.; PARK, J. U.; KANG, D.; LEE, K. J. Automatic prediction of atrial fibrillation based on convolutional neural network using a short-term normal electrocardiogram signal. Journal of Korean Medical Science, v. 34, n. 7, p. 1–10, 2019.

FOWLKES, W. Y.; CREVELING, C. M. Engineering Methods for Robust Product Design. Addison-Wesley, 1995.

GIRSHICK, R. Fast R-CNN. Proceedings of the IEEE International Conference on Computer Vision, v. 2015 Inter, p. 1440–1448, 2015.

GIRSHICK, R.; DONAHUE, J.; DARRELL, T.; MALIK, J. Rich feature hierarchies for accurate object detection and semantic segmentation. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, p. 580–587, 2014.

HARBOLA, S.; COORS, V. One dimensional convolutional neural network architectures for wind prediction. Energy Conversion and Management, 2019.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep residual learning for image recognition. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, p. 770–

778, 2015a.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. 2015 IEEE International Conference on Computer Vision (ICCV), v. 2015 Inter, p. 1026–1034, 2015b. IEEE.

HUBEL, D. H.; WIESEL, T. N. Receptive fields of single neurones in the cat's striate cortex. The Journal of Physiology, v. 148, n. 3, p. 574–591, 1959.

HUBEL, D. H.; WIESEL, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. The Journal of Physiology, v. 160, n. 1, p. 106–154, 1962.

HUBEL, D. H.; WIESEL, T. N. Receptive fields and functional architecture of monkey striate cortex. The Journal of Physiology, v. 195, n. 1, p. 215–243, 1968.

ILIEVSKI, I.; AKHTAR, T.; FENG, J.; SHOEMAKER, C. A. Efficient hyperparameter optimization of deep learning algorithms using deterministic RBF surrogates. 31st AAAI Conference on Artificial Intelligence. Anais... . p.822–829, 2017.

JIA DENG; WEI DONG; SOCHER, R.; et al. ImageNet: A large-scale hierarchical image database. 2009 IEEE Conference on Computer Vision and Pattern Recognition. Anais... . p.248–255, 2009. IEEE.

KACKER, R. N.; LAGERGREN, E. S.; FILLIBEN, J. J. Taguchi's orthogonal arrays are classical designs of experiments. Journal of Research of the National Institute of Standards and Technology, v. 96, n. 5, p. 577–591, 1991.

KHANDAY, O. M.; DADVANDIPOUR, S. Convolutional Neural Networks and Impact of Filter Sizes on Image Classification. Multidiszciplináris Tudományok, v. 10, n. 1, p. 55–60, 2020.

KRIZHEVSKY, A. Learning Multiple Layers of Features from Tiny Images. Tech Report, 2009.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. ImageNet classification with deep convolutional neural networks. Communications of the ACM, v. 60, n. 6, p. 84–90, 2017.

LAKHMIRI, D.; DIGABEL, S. LE; TRIBES, C. HyperNOMAD : Hyperparameter optimization of deep neural networks using mesh adaptive direct search. , p. 1–30, 2019.

LAMY, L.; CHAVES NETO, A. Taguchi's Quality Measures (Loss Function and Signal-to-Noise Ratio) using an Application with a Graphical Interface. XXXVIII Iberian Latin-American Congress on Computational Methods in Engineering. Anais... , 2017. Florianópolis, SC, Brazil: ABMEC.

LECUN, Y.; BOSER, B.; DENKER, J. S.; et al. Backpropagation Applied to Handwritten Zip Code Recognition. Neural Computation, v. 1, n. 4, p. 541–551, 1989.

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. Proceedings of the IEEE, v. 86, n. 11, p. 2278–2324, 1998.

LECUN YANN; CORTES CORINNA; BURGES CHRISTOPHER. The MNIST database of Handwritten Digits.

LI, Q.; CAI, W.; WANG, X.; et al. Medical image classification with convolutional neural network. 2014 13th International Conference on Control Automation Robotics and Vision, ICARCV 2014, v. 2014, n. December, p. 844–848, 2014.

LI, Z.; LIU, F.; YANG, W.; PENG, S.; ZHOU, J. A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects. IEEE Transactions on Neural Networks and Learning Systems, p. 1–21, 2021.

LOUSSAIEF, S.; ABDELKRIM, A. Convolutional Neural Network hyper-parameters optimization based on Genetic Algorithms. International Journal of Advanced Computer Science and Applications, v. 9, n. 10, p. 252–266, 2018.

MAAS, A. L.; HANNUN, A. Y.; NG, A. Y. Rectifier nonlinearities improve neural network acoustic models. in ICML Workshop on Deep Learning for Audio, Speech and Language Processing. Anais... . v. 30, p.3, 2013.

MAITRA, S.; OJHA, R. K.; GHOSH, K. Impact of Convolutional Neural Network Input Parameters on Classification Performance. 2018 4th International Conference for Convergence in Technology, I2CT 2018. Anais... . p.1–5, 2018. IEEE.

MILORADOVIĆ, N.; STOJANOVIĆ, B.; MITROVIĆ, S.; VELIČKOVIĆ, S. Application of Taguchi method in the optimization of zinc based composite. Proceedings on Engineering Sciences, v. 1, n. 1, p. 104–109, 2019.

MONTI, F.; FRASCA, F.; EYNARD, D.; MANNION, D.; BRONSTEIN, M. M. Fake News Detection on Social Media using Geometric Deep Learning, 2019.

NAIR, V.; HINTON, G. E. Rectified linear units improve Restricted Boltzmann machines. ICML 2010 - Proceedings, 27th International Conference on Machine Learning. Anais... . p.807–814, 2010.

RADIUK, P. M. Impact of Training Set Batch Size on the Performance of Convolutional Neural Networks for Diverse Datasets. Information Technology and Management Science, v. 20, n. 1, p. 20–24, 2018.

REN, S.; HE, K.; GIRSHICK, R.; SUN, J. Faster R-

CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, v. 39, n. 6, p. 1137–1149, 2017.

SCHULDT, C.; BARBARA, L.; STOCKHOLM, S.-. Recognizing Human Actions : A Local SVM Approach ∗ Dept . of Numerical Analysis and Computer Science. Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on, 2004.

SERIZAWA, T.; FUJITA, H. Optimization of Convolutional Neural Network Using the Linearly Decreasing Weight Particle Swarm Optimization, 2020.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition, 2014.

SIRINUKUNWATTANA, K.; RAZA, S. E. A.; TSANG, Y.-W.; et al. Locality Sensitive Deep Learning for Detection and Classification of Nuclei in Routine Colon Cancer Histology Images. IEEE Transactions on Medical Imaging, v. 35, n. 5, p. 1196–1206, 2016.

TAGUCHI.G; KONISHI, S. Taguchi methods, orthogonal arrays and linear graphs: Tools for quality engineering. Dearbom, American supplier institute, 1987.

TAGUCHI, G.; CHOWDHURY, S.; WU, Y. Taguchi's Quality Engineering Handbook. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2004.

TAIGMAN, Y.; YANG, M.; RANZATO, M.; WOLF, L. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. 2014 IEEE Conference on Computer Vision and Pattern Recognition. Anais... . p.1701–1708, 2014. IEEE.

TRIVINO, M. C. A.; DESPRAZ, J.; SOTELO, J. A. L.; PENA, C. A. Deep Learning on Retina Images as Screening Tool for Diagnostic Decision Support. , 2018.

TSIOLIKAS, A.; TSIAMITROS, D.; KITSAKIS, K.; et al. optimization of neural network parameters using taguchi robust design: Application in plasma arc cutting process. Proceedings - 2017 4th International Conference on Mathematics and Computers in Sciences and in Industry, MCSI 2017. Anais... . v. 2018-Janua, p.57–61, 2017.

WU, J.; ZHANG, Q.; XU, G. Tiny ImageNet Challenge. Technical Report, 2017.

XU, B.; WANG, N.; CHEN, T.; LI, M. Empirical Evaluation of Rectified Activations in Convolutional Network, 2015.

ZEILER, M. D.; FERGUS, R. Visualizing and Understanding Convolutional Networks. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), v. 8689 LNCS, n. PART 1, p. 818–833, 2013.

ZHANG, X.; CHEN, X.; YAO, L.; GE, C.; DONG, M. Deep Neural Network Hyperparameter Optimization with Orthogonal Array Tuning. , p. 287–295, 2020.