Indonesian Journal of Electrical Engineering and Computer Science
Vol. 24, No. 2, November 2021, pp. 1009~1016
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v24.i2.pp1009-1016

Dropout, a basic and effective regularization method for a deep learning model: a case study

Brahim Jabir, Noureddine Falih
LIMATI Laboratory, Sultan Moulay Slimane University, Beni Mellal, Morocco

Article Info

Article history:

Received May 3, 2021
Revised Sep 8, 2021
Accepted Sep 13, 2021

Keywords:

CNN
Deep learning
Dropout
Machine learning
Regularization

ABSTRACT

Deep learning is based on a network of artificial neurons inspired by the human brain. This network is made up of tens or even hundreds of "layers" of neurons. The fields of application of deep learning are indeed multiple; Agriculture is one of those fields in which deep learning is used in various agricultural problems (disease detection, pest detection, and weed identification). A major problem with deep learning is how to create a model that works well, not only on the learning set but also on the validation set. Many approaches used in neural networks are explicitly designed to reduce overfit, possibly at the expense of increasing validation accuracy and training accuracy. In this paper, a basic technique (dropout) is proposed to minimize overfit, we integrated it into a convolutional neural network model to classify weed species and see how it impacts performance, a complementary solution (exponential linear units) are proposed to optimize the obtained results. The results showed that these proposed solutions are practical and highly accurate, enabling us to adopt them in deep learning models.

Corresponding Author:

Brahim Jabir
LIMATI Laboratory
Sultan Moulay Slimane University
Mghila, BP 592 Beni Mellal, Morocco
Email: ibra.jabir@gmail.com

1. INTRODUCTION

Machine learning and deep learning are part of artificial intelligence. These approaches both result in empowering computers to make intelligent decisions. However, deep learning is a subcategory of machine learning because it relies on unattended learning, which is a form of learning based on mathematical approaches used to model data [1]. Deep learning applications are used in various sectors like: image recognition, automatic translation, autonomous car, medical diagnosis, personalized recommendations, automatic moderation of social networks, financial prediction and automated trading, identification of defective parts. The field that interests us and in which we have experimented is “agriculture” [2], [3], indeed, we can use it for the detection of weeds, water management, detection of insects and diseases.

Deep learning is a network that is made up of tens or even hundreds of “layers” of neurons, each receiving and interpreting information from the previous layer [4]. A set of theories and models have been brought in existence. Their unified goal is to reach higher accuracy levels that can be applied to solve problems in several fields, agricultural, industrial, health. This field always remains a subject of research. Thousands of experiments and thousands of scientific papers have been produced in deep learning and machine learning in all application fields in recent years. However, the research door is still open where scientists are still trying to reach better models that can gain an important degree of learning. They are trying to discover all the settings that affect it, starting from data collection, through its preparation and purification, to the output. Preparing a deep learning model that can reach a good performance is a troublesome issue [5]. A model with too little capacity cannot gain proficiency with the problem, while a model with an excess of capacity can learn it excessively well and overfitting [6].

Several approaches appeared to reduce the generalization overfit turn about using a larger model that may be important to utilize regulation during training that keeps the weight of the model little. These strategies diminish overfitting, yet they can likewise prompt faster model optimization and better performance. Among these techniques [7], we can distinguish empirical methods (dropout, dropconnect, stochastic pooling) and explicit strategies (weight degradation, network size adjustment) [8]. Still, these means must be subject to controls and rules that we will try to discuss. In this article, we will train a deep learning model on our prepared dataset, discuss overfitting in convolutional neural networks (CNN) training, propose a regularization solution, and conclude with a complementary solution, which makes it possible to reinforce and optimize the results obtained.

2. MATERIEL AND METHODS

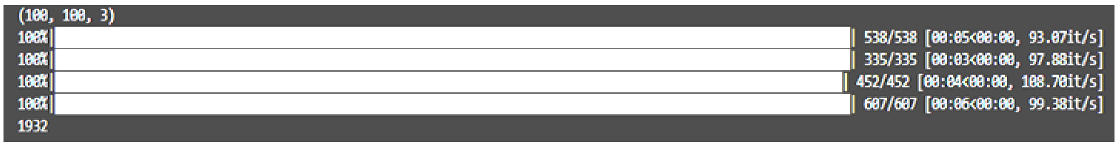
In this study, we used a set of techniques and methods that have a direct relationship with deep learning, where we used the libraries of Keras and Tensorflow for Python in order to build and train our model, we used Tensorboard to evaluate its performance, we used a dataset to conduct the training [9]. We will discuss the regularization methods used during this study to prevent training overfit, and we will try to stand at each one during the following paragraphs.

2.1. Tools and libraries

We have used Tensorflow and Keras libraries for our experimentation. Tensorflow is an open-source platform for artificial intelligence (AI). It has an extensive, adaptable environment of devices and libraries [10]. This tool helped us to build our CNN model. Keras is the most widely used Python tool in the world for deep learning. This open-source library, created by François Chollet, easily and quickly creates neural networks based on the main frameworks (Tensorflow, Pytorch and MXNET) [11]. Evaluation of the model is done on the Tensorboard tool, which is a tool to visualize the obtained results. It also allows us to view the structure of the model as a graph [12]. It is launched from a terminal command (*tensorboard --logdir = folder/*).

2.2. Dataset

The dataset is one of the significant factors affecting the quality of the training models, and whenever we have a large dataset well prepared, we have high training accuracy [13]. The dataset used during this experiment is a dataset that we previously used during a study on object detection that identifies weeds using CNN [14], [15]. It contains 1932 images of four types of weeds as shown in Figure 1; these images are collected in wheat fields with a professional canon camera EOS 700D. That offer large datasets in different domains intended for the data science community used to achieve data science goals. We made modifications to these images (resize, contrast and tile). We also added data-augmentation techniques (crop, rotation and flip) to produce more training images that can raise the model's accuracy [16]. We now have around 3000 images belonging to four different classes.



```
(100, 100, 3)
100%|██████████| 538/538 [00:05<00:00, 93.07it/s]
100%|██████████| 335/335 [00:03<00:00, 97.88it/s]
100%|██████████| 452/452 [00:04<00:00, 108.70it/s]
100%|██████████| 607/607 [00:06<00:00, 99.38it/s]
1932
```

Figure 1. The dataset samples

2.3. Regularization

The FC layers (fully connected) take up most of CNN's memory. Moreover, the concept of FC creates an exponential memory problem called "overfitting" ("over-connection" leading to over-learning), slowing down the processing of information, which pushes the model to fit too well to the training set, but difficult to generalize to new examples that were not in the training set. In another way, the model perceives specific images in the training set rather than general patterns and the training accuracy will be higher than the validation [17]. Regularization is a process aimed at avoiding this problem of over-learning, which results from an excessive adaptation of the model to the training data [18]. There are regularization methods to reduce the overfitting classified between empirical and explicit methods as shown in Table 1. We focus on dropout in this paper and try with a suitable solution to optimize the model.

The dropout is used to randomly "turn off" or "ignore" neurons (with a predefined probability, often every other neuron) as well as peripheral neurons. When neurons are randomly "turned off" from the network during learning, the other neurons will have to step in and handle the representation required to make predictions for the missing neurons [25]. Thus, with fewer neurons, the network is more responsive and can learn faster. At the end of the learning session, the "turned off" neurons are "turned back on" (with their original weights). The closer the fully-connected layer is to the source image, the fewer neurons will be extinguished as shown in Figure 2.

During the learning phase, for each iteration, a neuron is kept with a probability ***p***. Otherwise, it is deleted. During the test phase, all neurons are kept, so we want the neurons' outputs at the time of testing to be the same as their outputs at the time of learning. For example, in the case where the value of dropout - 0.25, neurons must reduce their production (output) by 25% at the time of the test to have the same output as during the training.

Table 1. Regularization methods

Method Empiric	Method Explicit
Dropout [19]	Weight degradation [22]
DropConnect [20]	Adjust the network size [23]
Pooling stochastique [21]	Batch normalization [24]

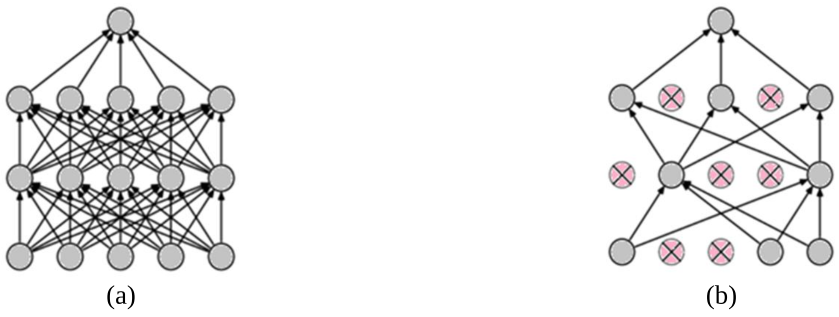


Figure 2. Applying dropout to prevent neural networks from overfitting [26]; (a) standard neutral net and (b) after applying dropout

2.4. Model architecture

Our goal during this study is to discover the regularization methods and how dropout is an important factor that eliminates overfitting; for this, we implemented a simple CNN model and defined their layers and hyperparameters in order to train them on our prepared dataset [27]. The architecture used in this study is structured: Conv → Pool → Conv→ FC → Output as shown in Figure 3. First, we did not integer dropout to analyze the results before and after dropout. The model is run on 20 epochs and gave us results that will be discussed in the next section.

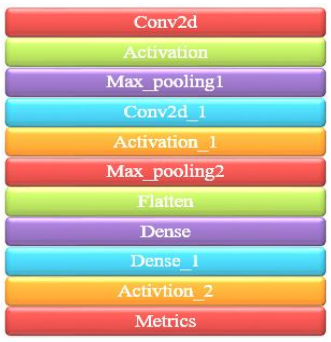


Figure 3. Model architecture before dropout method

3. RESULT AND DISCUSSION

We run the model on our dataset in 20 steps using Python. The results obtained are visualized on the Tensorboard tool. The validation of the model is to evaluate the capacity of the trained model to generalize to new samples. The accuracy of the training was also used as an estimator for ranking the model in a variable selection approach. The Figure 4 shows the results of the training, represented by the orange and blue curves.

The gap between the validation accuracy (orange curve) and the training accuracy (blue curve) demonstrates the quantity of overfitting as shown in Figure 4. Two potential cases have appeared in the chart. The orange curve (validation accuracy) shows exceptionally little approval accuracy compared to the training accuracy. Showing high overfitting implies we need to regularize the model with techniques (dropout for our case) and gather more information. The second conceivable case is when the validation accuracy tracks the training accuracy genuinely well [28]. This case demonstrates that the model capacity is not sufficiently high. It got us thinking about making the model larger by expanding the number of parameters. The following section shows us the solutions proposed to eliminate overfitting and achieve better performance.

3.1. Dropout solution

The following model is the same architecture but regularized with the technique of dropout. Dropout 0.5 means that each output neuron from the fully-connected layer has a 50% chance of being kept. Moreover, another dropout layer at the exit of the first dense with the probability of 25%. Dropout 0.75 was also placed after the pooling phase as shown in Figure 5. This technique randomly deactivates neurons in our network so that it is redundant and can find new ways to solve the overfitting problem.

From Tensorboard, we get the graph of training accuracy and validation accuracy, after training of our modified model above equipped with dropout. The Figure 6 shows the evolution of the validation represented by the blue curve, and the evolution of the training represented by the orange curve, these results are explained in the next part. From the results, we can immediately see that the model with dropout achieves a good performance. With model 2, the validation curve (orange) tracks that of the train (blue), which means overfitting is quickly minimized even with fewer neurons during learning (the effect of dropout). This effect can be seen in the precision where the percentage of good classification reached and 60% after 6 epochs. This precision continues to increase slowly, reaching 85% after 20 epochs. We can release that the easiest way to limit overfitting is to introduce the dropout technique. The location of this technique (dropout) can influence the results, even if we can be applied for each layer of the network or after selected layers, so we have to try different combinations to get the best results. Also, the fixed dropout probabilities directly influence the overfitting. The gap were partially reduced however, it needs more regularization and need improving the ability of the model to generalize. The proposed solution for these problems will be discussed in the next section.

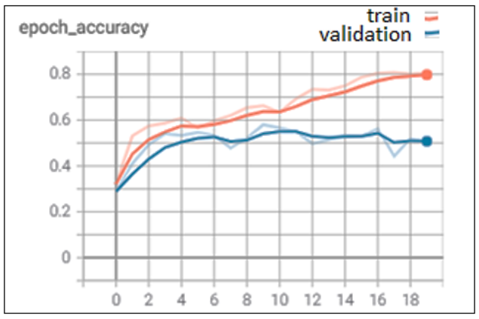


Figure 4. Training and validation accrurary

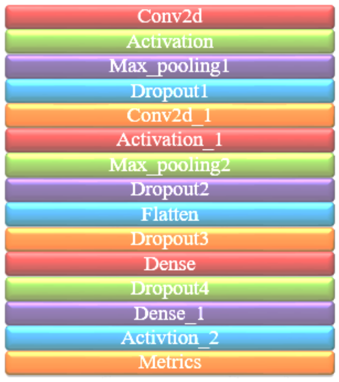


Figure 5. Model architecture after dropout method

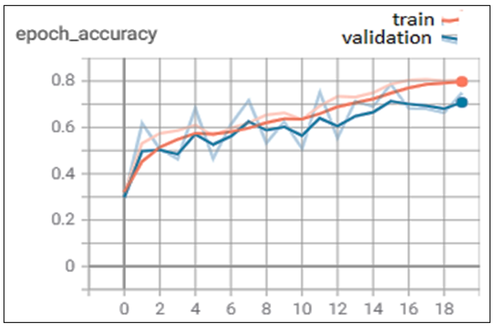


Figure 6. Training and validation accrurary after dropout

3.2. Optimization

The dropout technique results are interesting, but overfit is eliminated, and the accuracy needs to be boosted even more. For this, we will propose an optimization solution that can be accompanied with the dropout technique. In this part, we will discuss a solution for optimizing the results obtained based on the activation function. Currently, the most popular activation function for neural networks is (ReLu). The ReLu (rectified linear unit) activation function is the identity for positive inputs and zeroes otherwise. The main advantage of ReLu is that it solves the vanishing gradient problem. However, the Figure 7 shows that the ReLu (pink curve) is not negative and therefore has an average activation greater than zero. Units that have an average activation other than zero act as a bias for the next layer. If these units do not cancel each other out, the training causes a bias shift for the units in the next layer. Exponential linear units (ELU) like ReLu, addresses the vanishing gradient problem with identity for positive inputs. The same figure indicates that, compared to ReLu, ELU (purple curve) improves learning because it has negative values that allow it to push the average unit activations closer to zero, which speeds up learning and leads to better precision in classification. The ELU function is defined in the python code with the following line « *Keras.layers.ELU(alpha=1.0)* ». As per the (1), The ELU function with 0 < α is,

$$f(x) = \begin{cases} x \; if x > 0 \\ \alpha(\exp(x) - 1) \; if \; x \leq 0 \end{cases} \tag{1}$$

We will introduce ELU in our network and keep other layers and their order as in the previous model, then the architecture shows in Figure 8. We have trained the new model on 20 epochs. We then obtain the results of Figure 9. We will compare the ReLU model as shown in Figure 5 and that of the ELU model performances as shown in Figure 8 and see what changes will occur.

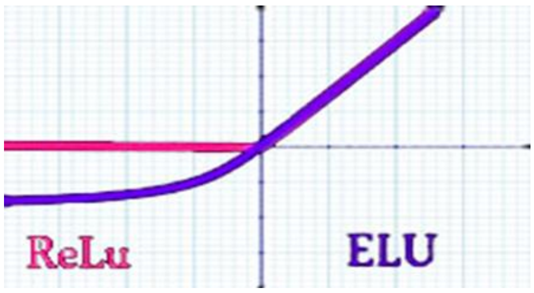


Figure 7. Relu and ELU (elu, alpha=1)

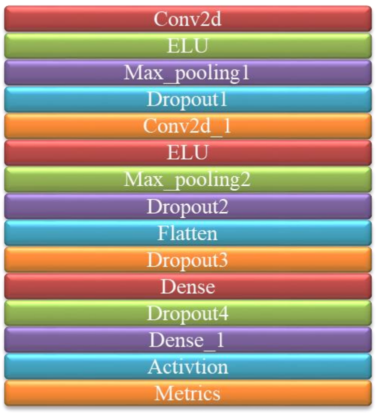


Figure 8. Replacing the activation function ReLu by ELU

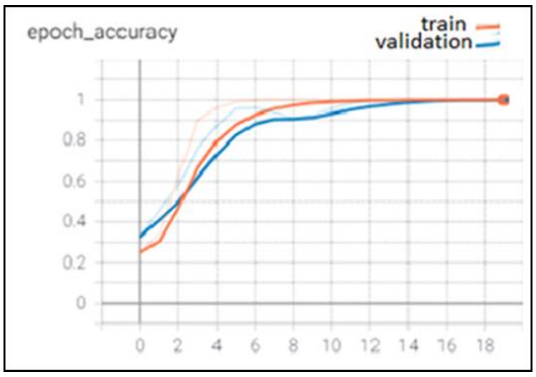


Figure 9. Learning evolution of the ELU model

The above graph shows the ability of ELU to generalize where the score of 97% is reached after 20 epochs. We then realize that the ELU activation function accelerates learning compared to ReLu and improves the network's ability to generalize. ELU keeps some of the positive things to fix some of the ReLU function problems [29]. There are several interpretations of these techniques, and there are many reasons why they are practical and effective solutions:

- The dropout can be interpreted as a sampling of a neural network in the complete neural network and update only the parameters of the sampled network. The difference is that these subnets share settings.
- Others see dropout as a form of data augmentation by artificially corrupting the entries in each layer. This greatly expands the number of examples the model will use when training to help protect against overfitting.
- Another interpretation is that this is a form of bagging in which a set of patterns is only trained on a small subset of data.
- ELU addresses the vanishing gradient problem with identity for positive inputs.
- ELU improves learning because it has negative values that allow it to push the unit's average activations closer to zero, which speeds up learning and leads to better precision in classification.

Now we have obtained our final model where the dropout and ELU are two essential parameters that effectively reduce overfitting and increase performance. Since we plan to get the most out of our network, we plan to let the learning go on for a very long time and back up the settings at each epoch until there is no further improvement in performance across the data. Finally, we can say that identifying overfitting is useful, yet it does not tackle the issue. Luckily, we have more choices to attempt in addition to dropout and ELU and the other regularization methods. Those other methods also can limit overfitting and increase the accuracy, we cite:

- Cross-validation: Utilize the initial training data to produce numerous more minor than expected train-test splits. Utilize these parts to tune the model [30].
- Increase the database size: Train the model with further data can improve algorithms to identify the signal better [31].
- Early stopping: consists of stopping the training step as soon as the validation loss reaches a plateau or increases [32].
- Batch normalization: Each layer observes inputs produced by layers preceding it. It would be advantageous to be centered and reduced inputs for each layer [33].
- Global average pooling (GAP) was proposed to replace the multi-layered perceptron part. The idea is to generate a feature map for each corresponding category. Instead of adding a perceptron after the feature maps, we take the average of each of them, and the result is inserted into the Softmax function. An advantage of GAP over perceptron layers is that there are no parameters to be trained; therefore, over-learning is avoided [34].

4. CONCLUSION

CNN is a multi-layered neural network that is used in pattern and image recognition problems. Many problems encountered during learning affect the model performance and negatively influence results obtained; among these problems, we find overfitting, which happens when a model learns the detail and noise in the training data. In our work, we used a CNN for images classification, and we trained it on a dataset, we discover the problem of overfitting when training it, and then we proposed dropout as a regularization technique that prevents to overfit problem. The results showed that the choice of the probability of dropout and its place significantly influences results. But the results are not sufficient for that we suggested adding an ELU function to optimize the accuracy. Indeed, it positively affected the results, enabling us to reduce the overfit and reach 97% accuracy. In future work, we will study the other techniques that address overfitting and bring out better results. They will be recorded and loaded into an intelligent raspberry-based system, enabling the real-time identification of weeds in the agricultural environment and allowing them to be sprayed locally.

REFERENCES

[1] M. Akour, H. Alsghaier, and O. Al Qasem, "The effectiveness of using deep learning algorithms in predicting students achievements," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 19, no. 1, pp. 387-393, 2020, doi: 10.11591/ijeecs.v19.i1.pp388-394.

[2] B. Jabir, N. Falih, and K. Rahmani, "Accuracy and Efficiency Comparison of Object Detection Open-Source Models," *International Journal of Online & Biomedical Engineering*, vol. 17, no. 5, pp. 165-184, 2021, doi: 10.3991/ijoe.v17i05.21833.

[3] Z. Ünal, "Smart farming becomes even smarter with deep learning-A bibliographical analysis," *IEEE Access*, vol. 8, pp. 105587-105609, 2020, doi:10.1109/ACCESS.2020.3000175.

[4] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Computer Science Review*, vol. 40, p. 100379, 2021, doi: 10.1016/j.cosrev.2021.100379.

[5] S. Rehman, S. Tu, O. Rehman, Y. Huang, C. Magurawalage, and C.-C. Chang, “Optimization of CNN through Novel Training Strategy for Visual Classification Problems,” *Entropy*, vol. 20, no. 4, p. 290, avr. 2018, doi: 10.3390/e20040290.

[6] M. H. Jopri, A. R. Abdullah, J. Too, T. Sutikno, S. Nikolovski, and M. Manap, "Support-vector machine and Naïve Bayes based diagnostic analytic of harmonic source identification," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 20, no. 1, pp. 1-8, 2020, doi: 10.11591/ijeecs.v20.i1.pp1-8.

[7] O. Deniz, A. Pedraza, N. Vallez, J. Salido, and G. Bueno, “Robustness to adversarial examples can be improved with overfitting,” *International Journal of Machine Learning and Cybernetics*, vol. 11, no. 4, pp. 935-944, 2020, doi: 10.1007/s13042-020-01097-4.

[8] V. Kodkin and A. S. Anikin, "The experimental identification method of the dynamic efficiency for frequency regulation algorithms of AEDs," *International Journal of Power Electronics and Drive Systems (IJPEDS)*, vol. 12, no. 1, pp. 59-66, 2021, doi: 10.11591/ijpeds.v12.i1.pp59-66.

[9] T. Doleck, D. J. Lemay, R. B. Basnet, and P. Bazelais, “Predictive analytics in education: a comparison of deep learning frameworks,” *Education and Information Technologies*, vol. 25, no. 3, pp. 1951-1963, 2020, doi: 10.1007/s10639-019-10068-4.

[10] B. Pang, E. Nijkamp and Y. N. Wu, “Deep learning with tensorflow: a review,” *Journal of Educational and Behavioral Statistics*, vol. 45, no. 2, pp. 227-248, 2020, 10.3102/1076998619872761.

[11] A. Farkas, G. Kertesz and R. Lovas, “Parallel and Distributed Training of Deep Neural Networks: A brief overview,” in *2020 IEEE 24th International Conference on Intelligent Engineering Systems (INES)*, Reykjavík, Iceland, July 2020, pp. 165-170, doi: 10.1109/INES49302.2020.9147123.

[12] D. C. Vogelsang and B. J. Erickson, “Magician’s Corner: 6. TensorFlow and TensorBoard,” *Radiology: Artificial Intelligence*, vol. 2, no. 3, p. e200012, May 2020, doi: 10.1148/ryai.2020200012.

[13] A. A. Ojugo and R. E. Yoro, "Forging a deep learning neural network intrusion detection framework to curb the distributed denial of service attack," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 2, pp. 1498-1509, 2021, doi: 10.11591/ijece.v11i2.pp1498-1509.

[14] B. Jabir and N. Falih, "Digital agriculture in Morocco, opportunities and challenges," in *2020 IEEE 6th International Conference on Optimization and Applications (ICOA)*. IEEE, 2020. pp. 1-5, doi: 10.1109/ICOA49421.2020.9094450.

[15] B. Jabir, N. S. Falih, and A. Tannouche, “A Strategic Analytics Using Convolutional Neural Networks for Weed Identification in Sugar Beet Fields," *AGRIS on-line Papers in Economics and Informatics*, vol. 13, no. 1, pp. 49-57, 2021, doi: 10.7160/aol.2021.130104.

[16] R. Poojary, R. Raina, and A. Kumar Mondal, "Effect of data-augmentation on fine-tuned CNN model performance," *IAES International Journal of Artificial Intelligence (IJAI)*, vol. 10, n. 1, pp. 84-92, 2021, doi: 10.11591/ijai.v10.i1.pp84-92.

[17] Q. Xu, M. Zhang, Z. Gu, and G. Pan, “Overfitting remedy by sparsifying regularization on fully-connected layers of CNNs,” *Neurocomputing*, vol. 328, pp. 69-74, 2019, doi: 10.1016/j.neucom.2018.03.080.

[18] G. Fatima and S. Saeed, "A Novel Weighted Ensemble Method to Overcome the Impact of Under-fitting and Over-fitting on the Classification Accuracy of the Imbalanced Data Sets," *Pakistan Journal of Statistics and Operation Research*, vol. 17, no. 2, pp. 483-496, 2021, doi: 10.18187/pjsor.v17i2.3640.

[19] C. Garbin, X. Zhu, and O. Marques, "Dropout vs. batch normalization: an empirical study of their impact to deep learning," *Multimedia Tools and Applications*, vol. 79, no. 19-20, pp. 12777-12815, 2020, doi: 10.1007/s11042-019-08453-9.

[20] Y. Sakai, B. U. Pedroni, S. Joshi, S. Tanabe, A. Akinin, and G. Cauwenberghs, "Dropout and DropConnect for Reliable Neuromorphic Inference Under Communication Constraints in Network Connectivity," in *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 4, pp. 658-667, Dec. 2019, doi: 10.1109/JETCAS.2019.2952642.

[21] X. Jiang, M. Lu, and S. H. Wang, “An eight-layer convolutional neural network with stochastic pooling, batch normalization and Dropout for fingerspelling recognition of Chinese sign language,” *Multimedia Tools and Applications*, vol. 79, no. 21, pp. 15697-15715, 2020, doi : 10.1007/s11042-019-08345-y.

[22] Q. Zheng, M. Yang, J. Yang, Q. Zhang, and X. Zhang, "Improvement of Generalization Ability of Deep CNN via Implicit Regularization in Two-Stage Training Process," in *IEEE Access*, vol. 6, pp. 15844-15869, 2018, doi: 10.1109/ACCESS.2018.2810849.

[23] X. Xiao, M. Yan, S. Basodi, C. Ji, and Y. Pan, "Efficient hyperparameter optimization in deep learning using a variable length genetic algorithm," arXiv preprint arXiv: 2006.12703, 2020.

[24] I. Nurhaida, V. Ayumi, D. Fitrianah, R. A. M. Zen, H. Noprisson, and H. Wei, “Implementation of deep neural networks (DNN) with batch normalization for batik pattern recognition,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 2, p. 2045, avr. 2020, doi: 10.11591/ijece.v10i2.pp2045-2053.

[25] P. Dileep, D. Das and P. K. Bora, "Dense Layer Dropout Based CNN Architecture for Automatic Modulation Classification," *2020 National Conference on Communications (NCC)*, 2020, pp. 1-5, doi: 10.1109/NCC48643.2020.9055989.

[26] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929-1958, 2014.

[27] P. Lu, B. Song, and L. Xu, "Human face recognition based on convolutional neural network and augmented dataset," *Systems Science & Control Engineering*, vol. 9, no. sup2, pp. 29-37, May 2021, doi: 10.1080/21642583.2020.1836526.

[28] Ò. Lorente, I. Riera and A. Rana, "Image Classification with Classic and Deep Learning Techniques," *Computer Vision and Pattern Recognition*, 2021, arXiv preprint arXiv:2105.04895, 2021.

[29] X. Liang and J. Xu, "Biased ReLU neural networks," *Neurocomputing*, vol. 423, pp. 71-79, 2021, doi: 10.1016/j.neucom.2020.09.050.

[30] S.-C. Kim, P. Ray, and S. R. Salkuti, “Islanding detection in a distribution network with distributed generators using signal processing techniques,” *International Journal of Power Electronics and Drive Systems (IJPEDS)*, vol. 11, no. 4, pp. 2099-2106, 2020, doi: 10.11591/ijpeds.v11.i4.pp2099-2106.

[31] A. Rácz, D. Bajusz, and K. Héberger, "Effect of Dataset Size and Train/Test Split Ratios in QSAR/QSPR Multiclass Classification," *Molecules*, vol. 26, no. 4, p. 1111, 2021, doi: 10.3390/molecules26041111.

[32] C. Moodley, B. Sephton, V. Rodríguez-Fajardo, and A. Forbes, "Deep learning early stopping for non-degenerate ghost imaging," *Scientific Reports*, vol. 11, no. 1, p. 8561, Déc. 2021, doi: 10.1038/s41598-021-88197-5.

[33] S. Shrestha, A. Alsadoon, P. W. C. Prasad, I. Seher, and O. H. Alsadoon, “A novel solution of using deep learning for prostate cancer segmentation: enhanced batch normalization,” *Multimed Tools Appl*, vol. 80, no. 14, pp. 21293-21313, Juny 2021, doi: 10.1007/s11042-021-10779-2.

[34] R. L. Kumar, J. Kakarla, B. V. Isunuri, and M. Singh, “Multi-class brain tumor classification using residual network and global average pooling,” *Multimed Tools Appl*, vol. 80, no. 9, pp. 13429-13438, avr. 2021, doi: 10.1007/s11042-020-10335-4.

BIOGRAPHIES OF AUTHORS

Brahim Jabir was born in Azilal, Morocco in May 1, 1990. He received his Master degree in 2015 in computer engineering and systems at the Sultan Moulay Slimane University in Beni Mellal, Morocco. Currently, He is a Ph.D. student in the Faculty of Sciences and Technics of the same University and he is working as a teacher of computer science in the regional centers of education and training professions in Beni Mellal, Morocco. His research interests are Digital Agriculture, Deep learning, Strategic Analytics and Information Systems.

Noureddine Falih was born in Rabat, Morocco in 1977. He received his PhD on Computer Sciences from Faculty of Sciences and Technologies of Mohammedia, Morocco in 2013. He is an associate professor in Polydisciplinary Faculty of Sultan Moulay Slimane University at Beni Mellal, Morocco since 2014. He has 18 years of professional experience in several renowned companies. His research interests are Information System Governance, Business Intelligence, Big Data Analytics and Digital Agriculture.