*Journal of Information Hiding and Multimedia Signal Processing* ©2024 ISSN 2073-4212
**Ubiquitous International** Volume **15**, Number **2**, June **2024**

# Evaluation of pruning according to the complexity of the image dataset

Jose Lozano, Diego Renza, and Dora Ballesteros

Telecommunications Engineering
Universidad Militar Nueva Granada
Carrera 11 101-80, Bogotá-Colombia
est.jose.lozano@unimilitar.edu.co, diego.renza@unimilitar.edu.co, dora.ballesteros@unimilitar.edu.co

Received March 2024; revised April 2024, accepted April 2024

Abstract. *In recent years, methods have been implemented that allow pruning a model trained with a dataset to reduce its size and/or its computational cost. It is clear that as the pruning of the model increases, its performance decreases, for example in terms of accuracy or F1-score. However, to date, the impact of dataset complexity on the pruned model, which in some cases allows further pruning without a significant change in its performance, has not been explored. For this reason, this study uses a metric to measure dataset complexity and evaluates the impact of pruning on the same network for different image classification problems (i.e., datasets of different complexity). It has been established that if the data set is of high complexity, the pruned model will lose performance faster, compared to the same network trained and pruned with the same pruning method, but with a lower complexity dataset. This allows for a better selection of the pruning percentage of the model, according to the complexity of the dataset.*

**Keywords:** pruning, dataset complexity, Convolutional Neuronal Network(CNN), deep learning, model compression, pruning evaluation, Spectral metrics.

## 1. Introduction.

In the field of artificial intelligence dedicated to image classification and object detection, convolutional neural networks (CNNs) have evolved as essential tools that enable machines to recognize and classify images. Despite their achievements, the challenge is to optimize these networks, making them more efficient without sacrificing their precision in the tasks assigned. The strategy of pruning neural connections and filters stands as a solution to reduce the computational cost of these networks, while maintaining their performance.

Pruning, or selective elimination of neural connections and filters, is presented as a key strategy in the optimization of CNNs. By reducing the complexity of these models, improved efficiency is achieved in terms of model size, inference latency, and computational resource usage. This process often involves identifying and removing connections or parameters that contribute minimally to the model's loss function, without significantly compromising its performance [1]. Additionally, there are structured and unstructured pruning techniques, with the former being preferable to maintain hardware efficiency by preserving regular connectivity patterns [2, 3].

Methods such as random pruning demonstrate that sparse networks can be obtained by randomly removing connections, as long as the sparsity level is carefully preselected [4]. Masked pruning addresses the challenges of federated learning by generating compact representations by combining pruning masks from different nodes [5]. On the other hand, techniques such as one-shot pruning leverage knowledge from previously trained models to efficiently extract subnetworks for new tasks [6]. Taken together, these advances make it possible to implement complex neural networks on constrained platforms without compromising their performance too much.

However, it is important to note that the performance of a pruned CNN may depend on the complexity of the classification task to be performed, i.e., the dataset. For example, a high pruning rate may very slightly affect the performance of a pruned network that had been trained for a low complexity dataset, but significantly affect the performance of the same type of network if the classification problem (dataset) changes. In other words, assessing the complexity of the dataset is crucial in determining the impact of pruning on CNNs. Therefore, it is important to be able to identify the complexity of the dataset prior to pruning in order to determine an appropriate pruning rate, not only for the type of network, but also for the specific classification problem to be addressed.

According to the above, this work examines how dataset complexity impacts pruning techniques and model efficiency. Comparative experiments utilize pre-trained models on CIFAR-10, CIFAR-100 and STL-10 to analyze adaptability across datasets of varying complexity. The central goal is investigating whether pruning can keep efficiency without compromising performance, even as data complexity increases. The experiments aim to provide insights into the interplay between dataset diversity, pre-trained models, pruning techniques and network optimization. By analyzing model adaptability and robustness to complexity changes, this work explores pushing the boundaries of efficiency gains through optimized pruning.

## 2. Background.

### 2.1. Convolutional Neural Networks.

Convolutional neural networks (CNNs) have revolutionized the field of deep learning, especially in image processing and computer vision applications. Inspired by the organization of the visual cortex of biological organisms, the architecture of CNNs takes advantage of the properties of translational invariance and hierarchical compositionality present in image and video data. Unlike fully connected networks, CNNs incorporate two key ideas: convolutions and pooling. Convolutions use filters that glide over the input extracting local features, allowing invariant patterns to be detected in small translations. As for pooling, it summarizes the information on local characteristics, identifying the most discriminating characteristics and reducing the dimensionality of the representations.

The hierarchical combination of these operations allows the extraction of more abstract and semantically higher-level features as the depth of the network increases. CNNs learn during training the filters and bias that best represent the input data for the desired task. Due to their ability to capture local and global patterns, CNNs have driven significant advances in image classification and segmentation tasks, object detection, scene recognition, among many other computer vision applications.

### 2.2. Prune and Pruning Methods.

Pruning in neural networks is a strategic process that consists of reducing connections and parameters within the network, thus reducing its internal complexity. This procedure not only increases processing speed, but also decreases the size of the model, making it especially valuable in edge computing applications or on resource-constrained devices [7, 8].

Pruning of convolutional neural networks can be performed at different levels: channel, layer, weights, and connection. Pruning by channel removes irrelevant channels in convolutional layers [9], pruning by layer removes redundant layers [10], pruning by weight removes weak connections, and pruning by connection simplifies the network topology.

On the other hand, there are various pruning approaches, including structured pruning, which are used in convolutional neural networks to reduce the number of filters and neurons in the convolutional layers [11]. In the case of unstructured pruning, it involves the elimination of individual connections, resulting in a network with an irregular structure [10].

Pruning can also be classified as local and global pruning, where local pruning removes connections at a specific layer, while global pruning removes connections across the entire network. Local pruning is usually applied to reduce the complexity of the convolutional layers, while global pruning seeks to reduce the complexity of the network as a whole [12].

The key advantages of network pruning include reducing computational parameters and operations, speeding up inference, reducing memory requirements, and improving performance by mitigating overfitting. However, excessive pruning can degrade model performance.

2.3. **Complexity assessment spectral metric.** Image classification is a fundamental task in computer vision that has seen tremendous advances in recent years thanks to deep convolutional neural networks (CNNs). However, training effective CNN models requires large labeled image datasets, the development of which takes significant effort. Because of this, it is crucial to be able to assess the inherent complexity of a given classification problem and determine the minimum data size required.

Several metrics have been proposed to measure the complexity of classification problems, but most of them were designed for nonimage problems and small datasets [13]. Recently, some work has explored methods for image ensembles, but they still have important limitations.

In this context, an approach called Cumulative Spectral Gradient (CSG), specifically designed to evaluate complexity in modern image classification problems had been presented. The CSG method employs spectral clustering on image features to derive a measure of inter-class overlap. A strong correlation of CSG with classification performance has been demonstrated using various CNN models on a variety of image sets. In addition, CSG is computationally efficient and shows significant improvements over existing techniques. The CSG metric allows characterizing the inherent difficulty in an image classification problem, with multiple potential applications such as data reduction, complex class analysis, model selection, etc. [14].

2.4. **Pre-trained models and transfer learning.** The use of pre-trained models and transfer learning has become an indispensable strategy in the field of convolutional neural networks and deep learning applied to computer vision. Transfer learning leverages pre-trained models on huge datasets (e.g. ImageNet) to significantly improve learning and performance on new tasks where data is limited.

Demonstrated benefits include significantly reducing the data and computational resource requirements on the target task, drastically speeding up training and improving the final performance, especially when data is sparse. Transfer learning has been widely adopted in the deep learning community, showing significant improvements in various computer vision applications such as image classification, object detection and segmentation. Finally, the use of pre-trained models enables efficient knowledge transfer between related tasks.

3. **Materials and methods.** The essence of this research lies in the exploration of how parameter pruning in convolutional neural networks (CNN) influences the performance of models intended for image classification tasks. In order to evaluate this impact, three image datasets widely used in computer vision were selected. The classification complexity

of each of these datasets was evaluated using a metric that does not require training models. Subsequently, classification performance was evaluated using pre-trained models for each of the datasets. Complementarily, these pre-trained models were pruned and their classification performance was also evaluated. These results allowed comparison and analysis of performance as a function of dataset complexity. The steps that guided this study are described below (see Figure 1):

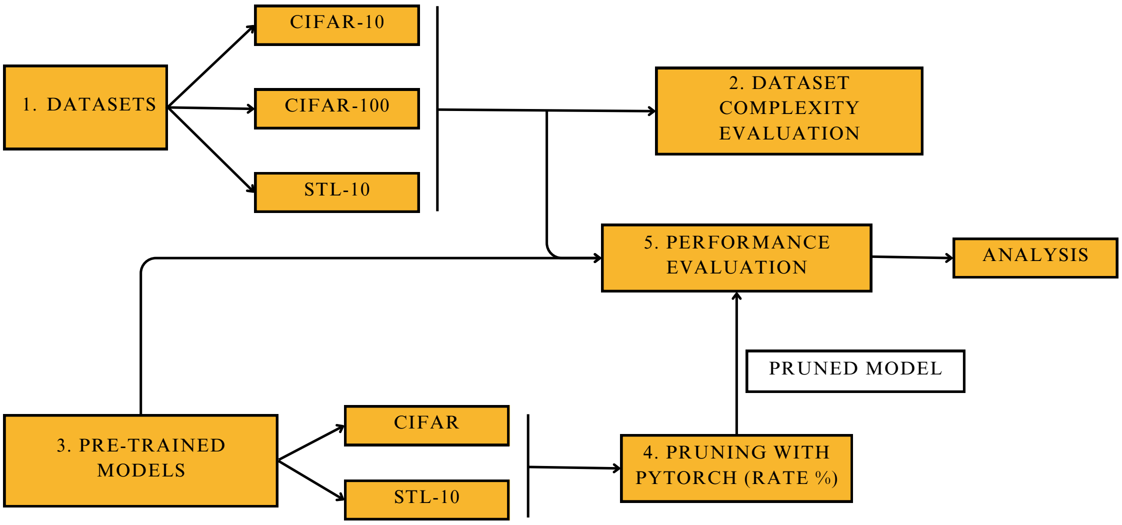
1. DATASETS
CIFAR-10
CIFAR-100
STL-10
2. DATASET COMPLEXITY EVALUATION
5. PERFORMANCE EVALUATION
ANALYSIS
PRUNED MODEL
3. PRE-TRAINED MODELS
CIFAR
STL-10
4. PRUNING WITH PYTORCH (RATE %)

Figure 1. Outline of the proposed methodology.

3.1. **Datasets.** In this study, we have selected three datasets: CIFAR-10, CIFAR-100 and STL-10. They differ from each other in two fundamental aspects: number of classes and image resolution. CIFAR-10 and STL-10 have 10 classes each, and CIFAR-100 has 100 classes. Therefore, the complexity of the datasets is different, as it is much easier for a model to identify patterns within a classification problem of 10 classes, than of 100 classes. Regarding image resolution, CIFAR-10 and CIFAR-100 have small images of $32 \times 32$ pixels, while STL-10 stands out with larger images of $96 \times 96$ pixels. Increasing the size of the images provides a richer and more detailed visual representation, adding perceptual complexity to the task.

Therefore, the selected datasets will allow us to assess the impact of pruning in different scenarios of data complexity.

3.2. **Dataset Complexity evaluation.** The CSG metric based on the spectral analysis of the correlation matrix between data samples was used to assess the complexity of the selected datasets. It allows one to reveal crucial information about the structure and distribution of the different classes in a vector space, and thus to better understand the impact of the dataset complexity on the performance of the pruned model. The result of this metric is a scalar, where higher values represent higher complexity and variability in the data; whereas, lower values suggest a simpler and more uniform structure. The results obtained using the spectral metric for the three datasets are shown in Table 1.

The spectral metric applied to the CIFAR-10 dataset returned moderate values of 3.76 for the training data and 4.50 for the test data. These values reveal a medium complexity in the correlation structure between the CIFAR-10 samples. The 10 classes that make up this dataset appear to have a reasonably uniform variability and distribution, without large disparities. The fact that the test set gets a slightly higher value could be due to

Table 1. Complexity metrics for the three selected datasets.

| | CIFAR-10 | CIFAR-100 | STL-10 |
|---|---|---|---|
| **Training / test data** | 3.76 / 4.50 | 26.08 / 29.64 | 3.48 / 3.67 |
| **Dataset complexity** | Medium | Very high | Low |

random fluctuations between the selected training and test samples. Overall, moderate complexity was expected in this balanced dataset.

On the other hand, the spectral metric showed a significantly greater complexity in CIFAR-100, with values of 26.08 and 29.64 for training and testing respectively. This was predictable given that CIFAR-100 has 100 classes, which introduces greater diversity and variability between categories. The high value likely reflects disparities in the distribution and frequency of certain classes. Even similar classes could manifest distinctive patterns of variation, increasing the overall complexity of the dataset. Again a slightly higher value is observed in the test samples.

Finally, the spectral metric applied to STL-10 returned low values of 3.48 and 3.67 for training and testing. Although STL-10 has the same number of classes as CIFAR-10, the higher resolution of the images (96 × 96 pixels) provides a richer visual representation. Therefore, it is the least complex dataset compared to CIFAR, according to the spectral metric. A possible explanation is that the classes in STL-10 are perceptually very different (planes, birds, cars, etc.), so the higher resolution does not increase the confusion between categories.

In-depth analysis of the spectral values for each dataset reveals findings consistent with the distinctive characteristics of each dataset, confirming CIFAR-100 as the most complex. This information laid the foundation for a better understanding of the impact of complexity on the performance of models subjected to pruning and retraining experiments.

3.3. **Pre-trained models.** In this research, it was decided to take full advantage of the power and versatility of pre-trained models. This strategic choice not only optimizes resources, but also significantly improves the effectiveness and breadth of analysis in research. The models used come from the “pytorch-playground” repository on GitHub (`https://github.com/aaron-xichen/pytorch-playground`). Pre-trained on large datasets and tested in various applications, these models offer improved accuracy and efficiency. Having been fine-tuned in different situations, these models prove to be more accurate and efficient in a variety of use cases. In particular, pre-trained models were used for the CIFAR-10, CIFAR-100 and STL-10 datasets. This choice is based on confidence in the quality of the models trained in the “pytorch-playground” repository, which simplifies the implementation and improves the quality of the results obtained in the research [15, 16, 17].

3.3.1. *Pre-trained models for CIFAR-10 AND CIFAR-100.* For the classification of CIFAR-10 and CIFAR-100 data, the same pre-trained architecture is used. In the first block of this architecture, a convolution layer with 3 input channels and 128 output channels is used, followed by batch normalization and a ReLU activation function. Subsequent blocks (second through seventh) follow a similar pattern, gradually increasing the number of channels to capture more complex features in deeper layers. Blocks 2, 4, 6 and 7 use a Max Pooling layer to reduce dimensionality. The network culminates with a fully connected layer that performs the final classification. The number of inputs of this dense layer is 1024 and the output corresponds to 10 or 100, representing the output classes for the CIFAR-10 and CIFAR-100 datasets, respectively (see Figure 2).

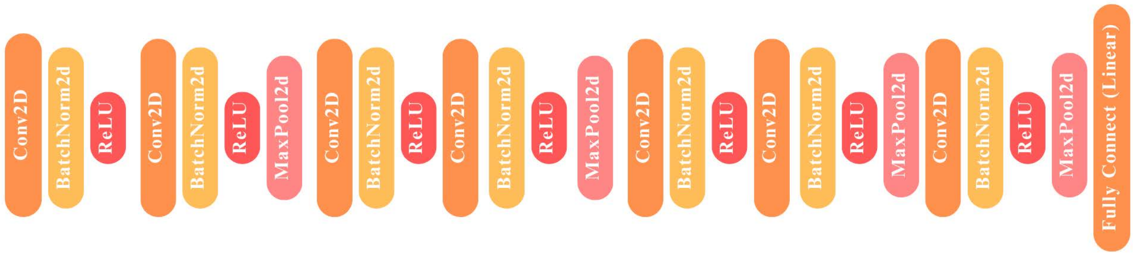

Figure 2. Pre-trained model use to classify CIFAR-10 and CIFAR-100 data.

3.3.2. *Pre-trained model for STL-10.* The pre-trained architecture used to classify STL-10 data is slightly different from the architecture for CIFAR, due to the larger dimensions of the input data. In the first block of the architecture, a convolution layer with 3 input channels and 32 output channels is used, followed by batch normalization, a ReLU activation function and a MaxPooling layer to reduce dimensionality. Subsequent blocks (second through sixth) follow a similar pattern, gradually increasing the number of channels to capture more complex features in deeper layers. Finally, the network culminates with a fully connected layer that performs the final classification. This linear layer has 256 inputs and 10 outputs, representing the output classes.

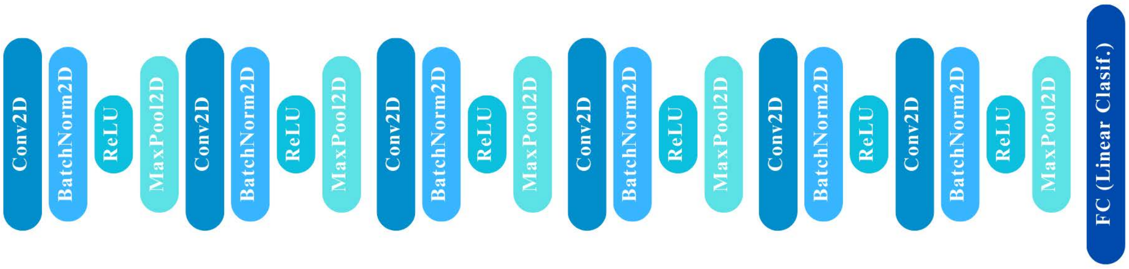

Figure 3. Pre-trained model use to classify STL-10 data.

The reuse of established and optimized CNN architectures for each dataset, through pre-trained models, represents a sound methodological decision that facilitated experimentation by focusing on the impact of pruning these models at different levels of data complexity.

3.4. **Pruning with PyTorch.** In the present investigation, progressive pruning of the models was performed, reducing from 20% to 90% (in increments of 10). The pruning was applied to the Conv2D layers of the pre-trained models in order to improve their computational efficiency. As explained so far, the procedure was developed in several stages, starting with both the selection of the model and the corresponding dataset.

Once the model and dataset were defined, the Conv2D layers to be pruned were identified. These layers were specified as a list of tuples that together with the parameter to be pruned (weights) and the pruning percentage, serve as inputs to the pruning PyTorch class.

The pruning technique used was “unstructured global pruning”, which globally prunes certain tensors by applying a specific pruning method. In our case, we defined the parameters to be pruned as the weights of the convolutional layers and the pruning method was based on the L1 norm. In this way, the selected pruning technique eliminates the individual connections (weights) that have the lowest values in terms of magnitude, thus preserving the most significant [10].

3.5. **Performance evaluation.** This section presents the performance results of the original model and of the pruned models for different pruning percentages, in the three classification problems of this study. We start by showing the performance of the model without pruning, which we will denote as "baseline" and arrive at the pruned model with a pruning percentage of 90%. In all cases, we will show the accuracy reduction with respect to the baseline model.

**Performance results of pruned models**

Table 2 shows the results in terms of accuracy of the pruned models, for the selected datasets.

Table 2. Evaluation of the performance of pruned models before fine-tuning. Negative values imply a decrease in accuracy with respect to the reference model (data shown in the first row, i.e., when the pruning percentage is 0%); while a positive value implies an increase in accuracy.

| Pruning percentage | CIFAR-10 | CIFAR-100 | STL-10 |
|---|---|---|---|
| **0** | 91.82 | 70.75 | 76.16 |
| **20** | -0.01 | 0.07 | 0.25 |
| **30** | -0.13 | -0.37 | 0.23 |
| **40** | -0.38 | -1.39 | 0.12 |
| **50** | -1.22 | -3.24 | -0.68 |
| **60** | -2.78 | -7.82 | -13.23 |
| **70** | -5.93 | -17.97 | -1.51 |
| **80** | -16.93 | -35.55 | -3.78 |
| **90** | -55.24 | -59.28 | -13.51 |

According to the results, the loss in accuracy is similar between the pruned models that were trained with the CIFAR10 and CIFAR100 datasets, for the different pruning percentages. Whereas, the pruned models that were trained with STL10 (the less complex dataset), had much lower losses for the same pruning percentages than the first two cases.

4. **Results and Discussion.** The experiments carried out by applying progressive elimination of parameters (from 20% to 90%) in CNN models trained with CIFAR-10, CIFAR-100 and STL-10 yield revealing results on the impact of data complexity on these optimization processes. We discuss how controlled parameter removal differentially affects and degrades classification accuracy based on the inherent complexity of the dataset.

4.1. **CIFAR-10.** The metric applied to CIFAR-10 delivered a value of 3.766834 in the training set, revealing a moderately complex structure in the relationships between classes and visual patterns. Extending this evaluation to the test set, the metric reached a value of 4.50780272, observing a slight increase in complexity. This increase could be attributed to specific variations in class distribution or to the presence of less frequent and more challenging patterns. The discrepancy between the training and test metrics underscores the importance of understanding the diversity of data that a model will face in different situations.

In terms of accuracy, the initial value of 91.82% obtained by the CNN model trained on the CIFAR-10 dataset is indicative of a good fit and generalization capability on this moderately complex dataset. For the untrained model, as the percentage of parameter pruning in the convolutional layers increased from 20% to 90%, a gradual decrease in model accuracy was observed, which is to be expected as synaptic connections are removed. However, analyzing the rate of decline in detail, it was remarkably small for pruning percentages less than or equal to 60%. In other words, the inherently not very high complexity of CIFAR-10, with only 10 classes and low image resolution, appears to allow the CNN model to preserve competitive accuracy even when subjected to aggressive parameter reduction through pruning. For percentages equal to or higher than 70%, the reduction is more than five percentage points, and is even reduced by more than half for pruning of 90%.

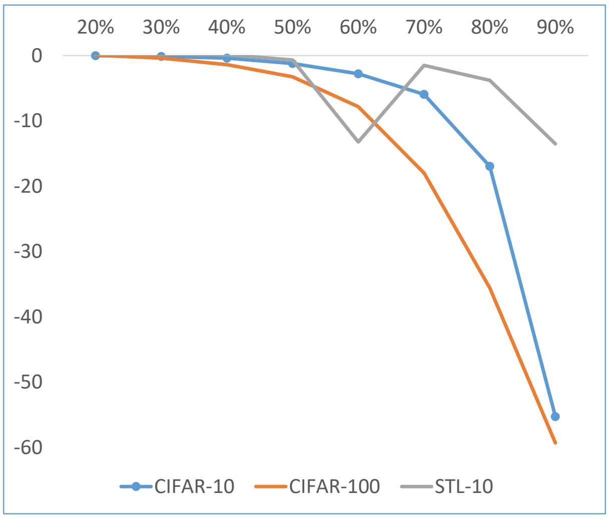

Figure 4. Performance (accuracy) versus percentage of pruning for pruned models and the three selected datasets.

4.2. **CIFAR-100.** The spectral metric returned a remarkably high value of 26,088 for the CIFAR-100 training data and 29,644 for the test data. These values, are substantially higher than those obtained for CIFAR-10, are indicative of a markedly higher inherent complexity in CIFAR-100. The higher value for the test data also suggests a slightly higher complexity for the evaluation samples. Taken together, these findings provide strong quantitative evidence that CIFAR-100 represents a significantly more complex classification challenge, given that it incorporates 10 times more classes than CIFAR-10. The substantially increased complexity captured by the spectral metric allows for a better understanding of why models trained with CIFAR-100 show an increased sensitivity to the pruning process.

Regarding accuracy, the baseline CIFAR-100 model achieved an accuracy of 70.75%, considerably lower than the 91.82% achieved in CIFAR-10. This initial metric reflects the greater complexity of discriminating 100 classes instead of 10. As the percentage of pruning increased, accuracy decreased more rapidly than in CIFAR-10, to a reduction of approximately 60 percentage points when 90% pruning is applied. This dramatic drop, provides tangible evidence of the model's marked sensitivity to parameter removal when trained on substantially more complex data. The additional complexity of CIFAR-100, captured quantitatively with the spectral metric, appears to cause the model to be unable to retain sufficient discriminative information under intense levels of pruning.

4.3. **STL-100.** The spectral metric values obtained for the STL-10 training and test data were 3.48 and 3.67, respectively. These values are indicative of a low inherent complexity in the STL-10 dataset. Despite containing higher resolution images than CIFAR-10, the spectral metric suggest that the perceptual complexity in discriminating the 10 classes in STL-10 is lower compared to that in CIFAR-10. Again, the slightly higher value for the test samples could reflect small fluctuations between the training and testing data. Together, these quantitative results provide evidence of the low complexity of the STL-10 dataset in relation to CIFAR.

While the spectral metric suggested low complexity in STL-10, the initial accuracy of the CNN model was only 76.16%, notably lower than the 91.82% achieved in CIFAR-10. This could be because the higher resolution of the images introduces representational challenges despite having the same number of classes. When pruning was applied, the drop in accuracy was more similar to that of CIFAR-10, except for a sudden drop when pruning at 60%. Even for pruning higher than 60%, the reduction in accuracy is lower than in the two previous cases. These results agree with the complexity value calculated for this dataset.

4.4. **Performance evaluation after fine-tuning of pruned models.** When using a pruning library, it is important to take into account the way in which the pruning process cancels the contribution of the pruned elements. Most of the available pruning libraries are based on setting minor parameters to zero, without restructuring the network [18]. Accordingly, once the model is retrained, it is possible to achieve similar levels of performance to the base model, which is mainly because the PyTorch pruning functions do not actually remove the connections of the pruned weights, but rather set them to zero.

That is, since the parameter connections were not really eliminated, when training is performed, these connections will again obtain a non-zero value, so that after a given retraining it is possible to achieve a performance similar to that of the baseline model. But in this case, such performance does not really correspond to a pruned model, but to an unpruned model (re-established connections). Either way, the reduction in model performance from pruning has implications for the performance of the retrained model, as Figures 4 and 5 show.

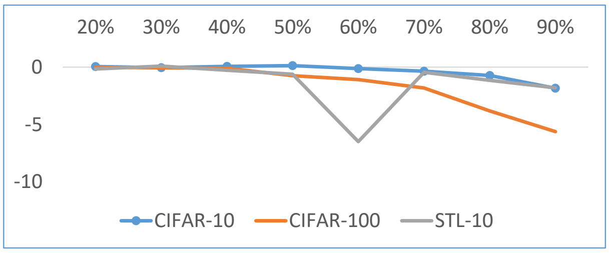

Figure 5. Performance (accuracy) versus percentage of pruning for fine tuned models (re-established connections)

5. **Conclusion.** Experiments using progressive parameter removal on several CNN models trained on the CIFAR-10, CIFAR-100, and STL-10 datasets reveal observations on the impact of data complexity on optimization processes. This analysis indicates that the removal of controlled parameters affects the classification accuracy differently, depending on the complexity of the dataset. Specifically, CIFAR-10, characterized by moderate complexity, shows robust resistance to degradation in accuracy even under progressive parameter pruning, indicating successful capture by the model. However, in CIFAR-100, significantly higher complexity leads to notable sensitivity to pruning, resulting in a dramatic decrease in accuracy, highlighting the challenges posed by datasets with greater class diversity. In STL-10, the least complex dataset according to the CSG metric, the

decrease in accuracy is similar to that observed in CIFAR-10, although less pronounced (except when the percentage of pruning is 60%, where the decrease is greater). In conclusion, it was found that for the three datasets evaluated, pruning percentages lower than 30% largely preserved the performance of the classification model, reducing its performance by no more than one percentage point. When the pruning percentage is greater than 30%, it was found that there is a direct relationship between the complexity of the dataset and the reduction in model performance. That is, the higher the complexity of the data set, the greater the reduction in performance of the pruned model.

**Acknowledgment.** This work was funded by "Universidad Militar Nueva Granada-Vicerrectoría de Investigaciones" under the grant INV-ING-3786 of 2023.

## REFERENCES

[1] Z. You, K. Yan, J. Ye, M. Ma, and P. Wang, "Gate decorator: Global filter pruning method for accelerating deep convolutional neural networks," *Advances in neural information processing systems*, vol. 32, 2019.

[2] Y. Hou, Z. Ma, C. Liu, Z. Wang, and C. C. Loy, "Network pruning via resource reallocation," *Pattern Recognition*, vol. 145, p. 109886, 2024.

[3] G. Fang, X. Ma, M. Song, M. B. Mi, and X. Wang, "Depgraph: Towards any structural pruning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16091–16101, 2023.

[4] S. Liu, T. Chen, X. Chen, L. Shen, D. C. Mocanu, Z. Wang, and M. Pechenizkiy, "The unreasonable effectiveness of random pruning: Return of the most naive baseline for sparse training," *arXiv preprint arXiv:2202.02643*, 2022.

[5] T. L. Gez and K. Cohen, "A masked pruning approach for dimensionality reduction in communication-efficient federated learning systems," *arXiv preprint arXiv:2312.03889*, 2023.

[6] H. Zhao and G. Long, "One-shot pruning for fast-adapting pre-trained models on devices," *arXiv preprint arXiv:2307.04365*, 2023.

[7] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[8] D. Filters'Importance, "Pruning filters for efficient convnets,"

[9] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 1389–1397, 2017.

[10] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," *arXiv preprint arXiv:1611.06440*, 2016.

[11] V. Vivancos Serrano, "Estudio de los efectos de métodos de pruning y dispersión de redes neuronales preentrenadas," B.S. thesis, Universitat Politècnica de Catalunya, 2023.

[12] D. Renza and D. Ballesteros, "Sp2ps: Pruning score by spectral and spatial evaluation of cam images," in *Informatics*, vol. 10, p. 72, MDPI, 2023.

[13] X. Hu, W. Liu, J. Bian, and J. Pei, "Measuring model complexity of neural networks with curve activation functions," in *Proceedings of the 26th ACM SIGKDD International Conference on knowledge discovery & data mining*, pp. 1521–1531, 2020.

[14] F. Branchaud-Charron, A. Achkar, and P.-M. Jodoin, "Spectral metric for dataset complexity assessment," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3215–3224, 2019.

[15] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz, "Importance estimation for neural network pruning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11264–11272, 2019.

[16] C. G. Pachón, D. M. Ballesteros, and D. Renza, "Senpis: Sequential network pruning by class-wise importance score," *Applied Soft Computing*, vol. 129, p. 109558, 2022.

[17] H. Tanaka, D. Kunin, D. L. Yamins, and S. Ganguli, "Pruning neural networks without any data by iteratively conserving synaptic flow," *Advances in neural information processing systems*, vol. 33, pp. 6377–6389, 2020.

[18] C. G. Pachón, D. M. Ballesteros, and D. Renza, "An efficient deep learning model using network pruning for fake banknote recognition," *Expert Systems with Applications*, vol. 233, p. 120961, 2023.