Int. J. Appl. Math. Comput. Sci., 2023, Vol. 33, No. 4, 663–672
DOI: 10.34768/amcs-2023-0047

CHOICE OF THE p–NORM FOR HIGH LEVEL CLASSIFICATION FEATURES PRUNING IN MODERN CONVOLUTIONAL NEURAL NETWORKS WITH LOCAL SENSITIVITY ANALYSIS

ERNEST JECZMIONEK [a], PIOTR A. KOWALSKI [b,c,*]

[a] Doctoral School
AGH University of Krakow
al. A. Mickiewicza 30, 30-059 Krakow, Poland
e-mail: ejec@agh.edu.pl

[b] Faculty of Physics and Applied Computer Science
AGH University of Krakow
al. A. Mickiewicza 30, 30-059 Krakow, Poland
e-mail: pkowal@agh.edu.pl

[c] Systems Research Institute
Polish Academy of Sciences
ul. Newelska 6, 01-447 Warsaw, Poland
e-mail: pakowal@ibspan.waw.pl

Transfer learning has surfaced as a compelling technique in machine learning, enabling the transfer of knowledge across networks. This study evaluates the efficacy of ImageNet pretrained state-of-the-art networks, including DenseNet, ResNet, and VGG, in implementing transfer learning for prepruned models on compact datasets, such as Fashion MNIST, CIFAR10, and CIFAR100. The primary objective is to reduce the number of neurons while preserving high-level features. To this end, local sensitivity analysis is employed alongside p-norms and various reduction levels. This investigation discovers that VGG16, a network rich in parameters, displays resilience to high-level feature pruning. Conversely, the ResNet architectures reveal an interesting pattern of increased volatility. These observations assist in identifying an optimal combination of the norm and the reduction level for each network architecture, thus offering valuable directions for model-specific optimization. This study marks a significant advance in understanding and implementing effective pruning strategies across diverse network architectures, paving the way for future research and applications.

Keywords: convolutional neural network, pruning, sensitivity analysis, transfer learning, ImageNet.

1. Introduction

In recent years, exploratory data analysis has undergone significant developments in both theoretical (Gui *et al.*, 2021) and practical (Saura, 2021) aspects. Data classification, a primary task in this domain, comprises two stages: model preparation and categorization of new, unknown data (Corrales *et al.*, 2018). This study focuses on using a supervised learning approach, wherein the training dataset contains input data and corresponding class labels representing the desired model outputs. Various classifiers are employed, including linear classifiers (naive Bayesian classifier, perceptron), k-nearest neighbors, decision trees, Bayesian networks, and neural networks (Kahani *et al.*, 2019).

Despite the predominance of numerical data processing, the complexity of real-world phenomena necessitates the utilization of diverse variable types, including binary, categorical, and mixed variables. Probabilistic methods, fuzzy logic, and interval information modeling can be applied to handle imprecise, uncertain, or inaccurate information (Cuzzocrea, 2020).

Image information, however, possesses a distinct

*Corresponding author

character. Algorithms dedicated to image analysis (e.g., cascade classifiers) are often employed to transform the obtained information into a numerical feature vector suitable for classical adaptive edge detection methods (Huang *et al*., 2005). Convolutional neural networks (CNNs) have emerged as a fundamental tool for image analysis tasks (Yamashita *et al*., 2018), capable of recognizing complex structures through multiple convolution operations.

A typical CNN architecture comprises multiple blocks for feature extraction and one to three dense layers for classification (Hidaka and Kurita, 2017). Transfer learning is a widely utilized technique that reuses a model trained on a large dataset for use as a feature detector in a target task, requiring only fine-tuning to adapt to the problem under study (Mishkin *et al*., 2017).

Despite their effectiveness, neural networks are often criticized as "black box" models due to their opacity (Tzeng and Ma, 2005). Comprehensive knowledge of the model and the ability to interpret individual components is crucial for a deeper analysis and model modification. In response, researchers have delved into neural structures using methods like Shapley additive explanations (SHAP) analysis (Lundberg and Lee, 2017), which assesses the influence of feature vector elements on machine learning algorithm results (Antwarg *et al*., 2021).

Another viable approach for neural structures is sensitivity analysis, which treats the neural network as a black box, studying input-output dependencies (Saltelli *et al*., 2008). Sensitivity analysis (SA) methods were initially applied to reduce the input layer of neural networks, such as multilayer perceptrons (MLPs) (Fock, 2014; Zurada *et al*., 1997), radial basis functions (RBFs) (Shi *et al*., 2005), and probabilistic neural networks (PNNs) (Kowalski and Kusy, 2018). Advances in SA enable deeper analysis and interpretation of individual network components (Kowalski and Kusy, 2017), leading to topological changes within the network, including modifying the number of neurons and adjusting weighting factors (Kusy and Kowalski, 2018).

This study introduces a novel approach utilizing local sensitivity analysis (LSA) to reduce the internal complexity of neural networks. The primary objective is to identify and eliminate redundant features within the flattening layer by examining the feature extraction process in the convolutional layers. To validate the proposed method, six pre-trained CNN architectures (DenseNet121, DenseNet169, ResNet50V2, ResNet101V2, VGG16, and VGG19) were employed, with transfer learning applied using benchmark datasets, including fashion MNIST, CIFAR10, and CIFAR100. During this research, the primary goal was to find the best possible value for the p-norm. This optimal value is needed to aggregate sensitivity data from various benchmarks, while also maintaining a high level of classification accuracy.

The paper is organized as follows. Section 2 provides a brief description of the datasets used for validation. Section 3 outlines transfer learning and pre-trained CNN structures. Sections 4 and 5 detail the Local Sensitivity Analysis and the main CNN structure reduction algorithm. The subsequent two sections present the numerical experiments' details and results. Finally, the paper concludes with a summary and discussion of the research on the proposed neural network reduction method.

2. Datasets

This paper uses the following open-access image classification datasets: fashion MNIST (Xiao *et al*., 2017), CIFAR10, and CIFAR100 (Krizhevsky, 2009). Fashion MNIST, designed as a replacement for the now-trivial MNIST, consists of 28×28 grayscale images representing 10 types of fashion clothing items. The dataset includes a training set of 60,000 elements and a test set of 10,000 elements. Both CIFAR datasets feature 50,000-element 32×32 color train sets and 10,000-element test sets. While CIFAR10 comprises 10 labels of animals and vehicles, CIFAR100 consists of 100 fine-grained categories grouped into 20 coarse-grained classes. This study utilizes CIFAR100's 100 fine-grained labels.

In this research, ImageNet serves as the dataset for pretraining all networks. ImageNet contains 181,167 training images and 50,000 validation images, organized into 1,000 categories. This dataset has posed a significant challenge and benchmark for currently developed networks. For the purpose of this study, ImageNet was preprocessed, and all images were center-cropped to a size of 256×256.

3. Convolutional neural networks

Developing large-scale CNNs from scratch is a laborious and time-consuming process, and competing against state-of-the-art networks presents its own challenges. Consequently, researchers often reuse well-established architectures. Training a network from a randomly-initialized state requires a greater number of epochs to achieve acceptable performance. To expedite the learning process for new problems, a common technique involves using pre-trained networks derived from other similar datasets, an approach called "transfer learning." In this paper, all the presented networks were pretrained on ImageNet (Russakovsky *et al*., 2014), a standard large-scale benchmark dataset.

DenseNet121 and **DenseNet169** (Huang *et al*., 2017) are deep networks consisting of 121 and 169 layers, respectively. This family of networks is based on dense

blocks connected to each other, which helps reduce the vanishing gradient problem by propagating features to deeper blocks. The concept of passing down features enables a decrease in the parameter count. DenseNets are designed to scale well with hundreds of layers, with the assumption that deeper networks yield higher accuracy. DenseNets are generally smaller in size and converge faster than other large networks.

ResNet50V2 and **ResNet101V2** (He *et al*., 2016b) contain improved residual blocks compared with the original ResNet(V1) architecture (He *et al*., 2016a). ResNetV2 utilizes identity connections to skip ReLU activation and pass input directly to the next residual block, which enhances performance and mitigates the vanishing gradient problem in very deep networks. Furthermore, post-convolution normalization results in unnormalized output, while batch normalization before convolution guarantees normalized input for convolution. These changes led to a TOP1 accuracy improvement of 1.1% and 0.8% for the ImageNet dataset (Keras, 2023). In this article, all ResNets belong to the V2 family, even if not explicitly mentioned.

VGG16 and **VGG19**, introduced in 2015 (Simonyan and Zisserman, 2015), are the oldest of the presented networks. The numbers in their names represent the total number of convolutional and fully connected layers. Both networks have three fully connected layers and either 13 or 16 convolutional layers. Unlike the previously mentioned architectures, VGGs lack skip connections and have a straightforward pipelined input. In terms of performance, VGGs exhibit slightly lower accuracy than DenseNets or ResNets and contain significantly more parameters.

Dropout and regularization are methods used to deal with overfitting in CNNs. The dropout technique works by temporarily leaving out a random selection of neurons during the training process. By doing this, the network becomes less reliant on any one neuron and spreads the learning across multiple neurons, which can help it to generalize better to new data. On the other hand, regularization is a method that introduces a penalty for large weights in the loss function. By adding this penalty, the method discourages the model from using extreme weight values, making it more stable and less likely to overfit. Thus, the model becomes more capable of working well not just on the training data but also on new, unseen data.

There are two principal strategies for network optimization: structural pruning, exemplified by methods such as LSA, and non-structural pruning, often implemented through regularization (Kowal *et al*., 2018). Structural pruning serves as a mechanism to curtail a network's complexity by selectively eliminating weights or, in some instances, entire neurons or layers (Kusy and Zajdel, 2021). This tactic reduces the overall size and intricacy of the network, resulting in a model that is simpler and potentially more interpretable. Conversely, non-structural pruning adopts a less aggressive approach. Instead of outright removal, it assigns zero values to specific weights, effectively minimizing their influence on the network's output. Notably, this method does not alter the fundamental architecture of the network, preserving its original structure while attenuating certain aspects of its functionality. In essence, while structural pruning offers the dual benefits of shrinking model size and lowering computational demand, non-structural pruning primarily affects computational processes. However, the impact of non-structural pruning is most evident when using specialized hardware or software.

4. Local sensitivity analysis

SA techniques, which determine how the uncertainty of a model's output can be attributed to the uncertainty in that model's input (Saltelli *et al*., 2004), are generally classified as global (GSA) or local (LSA). GSA methods, such as Sobol (Saltelli *et al*., 2010) or the extended Fourier amplitude sensitivity test (EFAST) (Saltelli *et al*., 2012), simultaneously modify all input variables to measure the impact of input on output variance. In contrast, LSA methods modify a single input directly to measure its influence on the output. While GSA methods have been used in similar applications in other works (Jeczmionek and Kowalski, 2021) and have yielded positive results in terms of accuracy, they have been found insufficient for state-of-the-art networks due to their extremely large complexity and lack of GPU acceleration.

In this article, LSA is applied as a partial gradient on the final layers after convolutions. For VGGs, this involves the classifier and the flattening layer; for DenseNets and ResNets, it involves the average pooling layer. Let us assume that the described layer has input vector x_i and output vector y_j, where $i \in 1, \ldots, I$ and $j \in 1, \ldots, J$. LSA then calculates the partial gradient of the layer $\frac{\partial y_j}{\partial x_i}$ for N forward propagations. Subsequently, an average of sensitivities is calculated for each neuron, i.e.,

$$S_{LSA} = \frac{1}{N} \sum_{i}^{N} \frac{\partial y_j}{\partial x_i}. \tag{1}$$

5. Prepruned transfer learning models

Transfer learning is a widely employed technique that facilitates the reuse of networks trained on different datasets. This section presents the application of various p-norms (Riesz, 1910) to the post-convolution layer (Algorithm 1). This method measures the importance of high-level features and investigates whether pruned networks, initially trained on large, diverse datasets, can

Algorithm 1. Prepruned transfer learning models.

1: train the selected neural network on the ImageNet training dataset
2: choose a p value for the p-norm and the reduction factor r
3: **for all** image in the ImageNet validation set **do**
4: run forward propagation for the image
5: calculate and store the average S_{LSA} values for each neuron in the first post-convolution layer
6: **end for**
7: apply the p-norm to all stored S_{LSA} vectors
8: sort the final S_{LSA} vector in ascending order
9: remove neurons corresponding to the lowest values in the sorted vector
10: **return** pruned network for use with other datasets

be directly transferred to smaller datasets. To perform the reduction task, S_{LSA} is calculated for 50,000 elements of the ImageNet validation set. Subsequently, the following p-norm is applied:

$$||S_{LSA}||_p = \Big(\sum_{n=1}^{50,000} |LSA_n|^p \Big)^{1/p}, \tag{2}$$

where n is the cardinality the of ImageNet validation set. In this approach, p covers values from 0.5 to 5 with step 0.5 with the addition of two special numbers for mathematics: e and π, i.e.,

$$p \in \{0.5, 1, 1.5, 2, 2.5, e, 3, \pi, 3.5, 4, 4.5, 5\}. \tag{3}$$

Common p-norms have their special names: $p = 1$ is known as the 'taxi-cab' or 'Manhattan' norm, $p = 2$ is called the 'Euclidean' (l^2) norm and $p \to \infty$ defines the 'maximum norm'.

Then with a vector of sensitivities, all values are sorted and reduction r is applied in a range of 0% to 50% with a step of 5%. This parameter represents the number of pruned features. Thus,

$$r \in \{0\%, 5\%, \ldots, 45\%, 50\%\} \tag{4}$$

and r neurons with the least sensitivity are pruned from the models. This procedure creates models described by p that are used by the norm and r connected to the reduction level.

6. Experiment

DenseNet121, DenseNet169, ResNet50, ResNet101, VGG16, and VGG19 are networks initially trained on the ImageNet dataset. Next, various permutations of p and r were applied to create pretrained networks according to Section 5. The resulting models were then trained on three smaller datasets: fashion MNIST, CIFAR10, and CIFAR100. This process generated a combination of six networks, twelve p-norms, ten reductions, and three datasets. The objective was to determine whether prepruning is an effective technique and, if so, identify the optimal pruning levels and norms for achieving the best results.

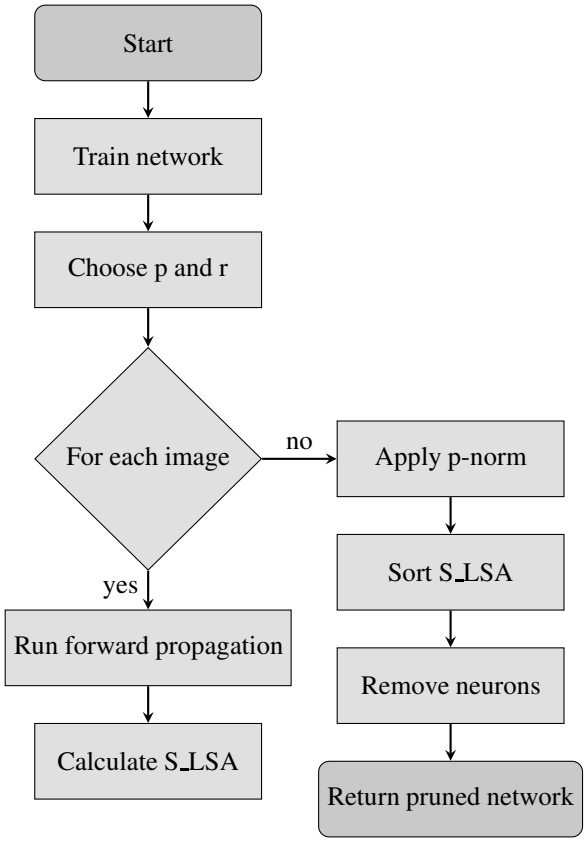

Fig. 1. Flowchart of the proposed procedure.

In this experiment, we utilized pretrained neural networks fine-tuned on the ImageNet dataset. However, the large number of output parameters from these networks posed a challenge of producing irrelevant outputs for the smaller datasets. To address this issue, we adjusted the number of neurons in the final layer of the classifier to correspond to the number of labels present in each dataset used for the experiment. As a result, initializing and training the classifier from scratch was necessary. Furthermore, due to limited computing resources, the number of epochs was set to 15, which significantly reduced the training time, considering the number of combinations to be tested. In this study, training a new model involved copying a pretrained model and applying random initialization of classification or fully connected layers at the end of the network. All parameters were trainable, and no freezing of layers was implemented. This approach not only limited the number of training epochs but also facilitated faster convergence.

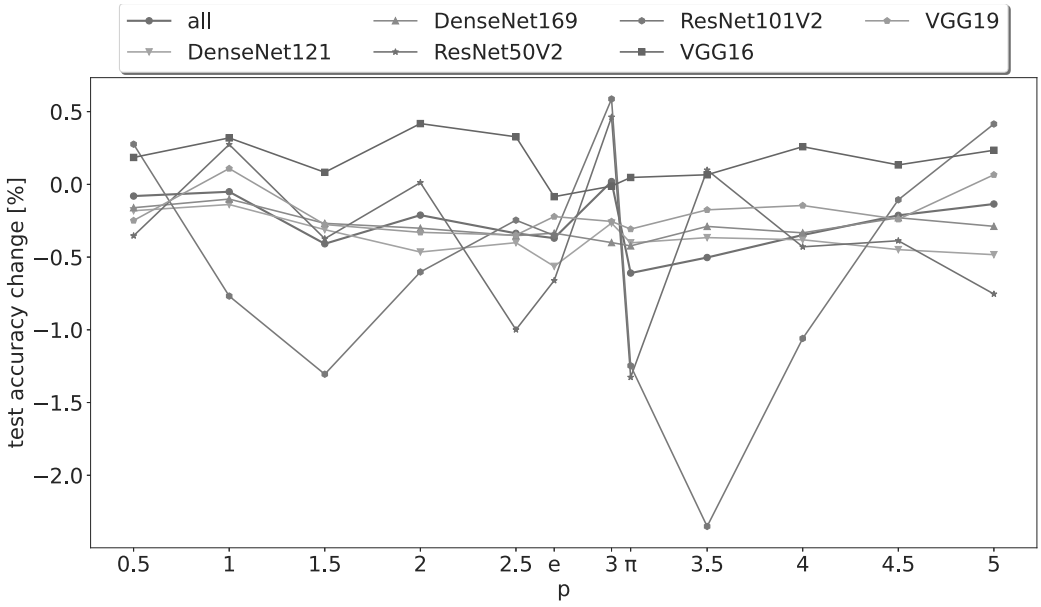

Fig. 2. Average accuracy results as a function of the change in p.

Training was conducted with a batch size of 128 and categorical cross-entropy as the loss function.

All datasets were preprocessed according to the specific network requirements. For VGG networks, the color space was converted from RGB to BGR, and color channels were zero-centered based on the ImageNet dataset. For ResNetV2 networks, image pixels were scaled between -1 and 1.

Additionally, DenseNet preprocessing involved scaling pixel values within a range of 0 to 1, after which each channel was normalized with respect to the ImageNet dataset.

7. Results

This section presents the results of LSA for high-level feature pruning. The objectives of this study are to determine an optimal p-norm value for transfer learning prepruning for individual networks and, in general, identify an optimal reduction level for achieving the best accuracy for the used CNNs, and discover the most effective combination of norm and reduction levels. Herein, the original test accuracy of unmodified networks serves as a benchmark for comparison, representing a 0% change in accuracy.

Figure 2 displays the dataset and reduction average test accuracy changes for individual networks and in general. Notably, VGG16 was the only network to maintain an average accuracy across various reduction levels, surpassing the benchmark of original accuracy. Interestingly, the larger sibling network, VGG19, exhibited results close to the average index for all networks. In contrast, the highest volatility was observed in the ResNet family networks. Both ResNets achieved peak accuracy improvements close to 0.5% for a p value of 3. However, these networks also experienced the most significant accuracy drops among all tested networks. Specifically, ResNet50 lost over 1% at p equal to π, while ResNet101 dropped over 2% at p equal to 3.5. Moreover, extreme off-accuracy values were observed for ResNet101 at extreme values of p. Notably, only a p-norm value of 3 could marginally increase accuracy for the researched networks as a whole.

Figure 3 presents the ratio of reduction to accuracy,

$$ratio_r = \frac{1}{(1-r)L} \sum_{norm} \sum_{dataset} (acc_{test} - acc_{original}). \tag{5}$$

This ratio is introduced to determine whether the accuracy drop is proportional to reduction. Zero indicates proportionality, positive values signify a disproportionately high reduction compared with a smaller accuracy drop (a positive scenario), and negative values imply a disproportionately low reduction compared with the accuracy drop (a negative scenario). This equation calculates the average difference between the benchmark accuracy and the accuracy for each dataset and norm. Here, the multiplication of the number of datasets and norms is denoted as L, and the fraction $\frac{1}{1-r}$, where r is a value between 0.05 and 0.5, scales the ratio proportionally for accuracy and reduction. Additionally, a reduction of $x\%$ compensates for the drop in accuracy by $x\%$.

Consistent with Fig. 2, VGG16 surpassed the

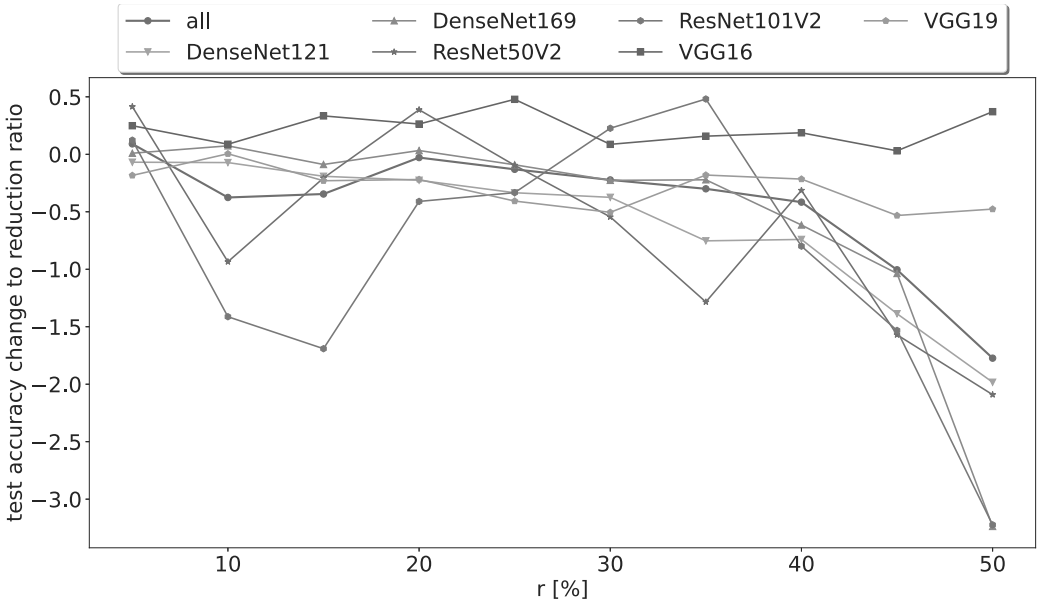

Fig. 3. Results of the average accuracy with respect to the reduction ratio.

benchmark for all reduction levels. In contrast, VGG19 displayed subpar results, although a 35% reduction exceeded the general index of all networks. The VGG family's increased resistance to high-level feature pruning might be attributed to the number of parameters; VGG16 has over three times more parameters than ResNet101 and nearly seven times more than DenseNet169. Like the previous case, ResNet networks exhibited high volatility, reaching an extreme range of the ratio.

In the experiment, all networks except for the VGG architecture experienced a significant drop at the 40% reduction. Interestingly, networks with a larger number of layers, such as DenseNet169 and ResNet101, declined faster than their smaller counterparts, i.e., DenseNet121 and ResNet50.

Table 1 reports the average accuracy for all networks across norm and reduction pairs. A *dotted line* border highlights the top-3 maximal accuracies for p-norm, while a *solid line* border indicates the top-3 maximum accuracies for the reduction level.

In this study, it was anticipated that lower reduction levels would yield the best accuracy results. As shown in the table, an average accuracy improvement of 0.8% was attained with a p value of 1 and a 10% reduction. Surprisingly, the second-best result of 0.7% increase was achieved by pairs with very different components. A 5% reduction is not far from the best result, but a p value of 5 significantly deviates from the p-norm range. The next top-performing solution had a p value of 3.5 and a larger reduction level of 25%.

In the table, a *dashed line* border marks positive average accuracy improvements for norm or reduction. As shown, only p-norms with a value of 3, on average, maintain the accuracy level of the original solutions. While it is unsurprising that a small reduction of 5% can, on average, increase accuracy, this is achievable only by 0.1%. Interestingly, a reduction of 20% resulted, on average, in solutions comparable to those of the original networks.

Table 2 presents the best accuracy changes and the best/worst scenarios for maximum pruning levels for each network. The VGG family achieved modest accuracy improvements of 1.3% and 1% for VGG16 and VGG19, respectively. Notably, a 50% reduction did not significantly impact accuracy. As displayed in the table, the difference between the best and worst solutions was 0.9% and 1.6% for VGG16 and VGG19, respectively.

Table 3 compares the output sizes from the convolutional layers, measured in terms of features or neurons. The outputs from these convolutional layers constitute the inputs to the classifier segment of the respective networks. This transition from the convolutional layers to the classifier is often termed as "lattening" the output. This process typically involves transforming a multi-dimensional tensor of feature maps into a one-dimensional tensor or vector. The neurons arising from this transformation are the focal points for the proposed pruning method. The specific features selected as input for the classifier are contingent upon the employed norm. Consequently, networks with larger classifiers, such as those in the VGG family, can particularly benefit from this pruning method. By reducing the input, the method effectively decreases the number of weights required in the subsequent layer.

Table 1. Results of average test accuracy depending on changing p and r. The dotted line border is the top-3 maximum accuracy for p, the solid line border is the top-3 maximum accuracy for r, the dashed line border denotes positive average accuracy improvements for p or r.

p/r	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	avg	max
0.5	-0.1%	0.0%	0.2%	0.3%	-0.3%	0.0%	0.4%	0.0%	-0.4%	-0.8%	-0.1%	0.4%
1	0.6%	0.8%	-1.3%	0.0%	-0.2%	-0.3%	0.6%	0.0%	-0.1%	-0.7%	-0.1%	0.8%
1.5	0.2%	-0.1%	-0.4%	-0.2%	-0.4%	-0.2%	-0.1%	-0.1%	-0.5%	-2.1%	-0.4%	0.2%
2	0.6%	-0.4%	-0.4%	0.5%	0.3%	-0.1%	-0.4%	-0.8%	-0.5%	-1.0%	-0.2%	0.6%
2.5	0.3%	-0.7%	-0.4%	-0.1%	-0.1%	-0.4%	0.1%	-0.8%	-0.6%	-0.6%	-0.3%	0.3%
e	0.2%	-1.0%	-0.2%	-0.1%	-0.6%	-0.3%	-0.1%	-0.3%	-0.6%	-0.8%	-0.4%	0.2%
3	-0.3%	0.5%	0.3%	0.0%	0.1%	0.3%	-0.2%	0.0%	-0.3%	-0.2%	0.0%	0.5%
pi	-1.0%	-0.9%	-0.5%	-0.2%	0.1%	-0.6%	-0.3%	-0.8%	-1.1%	-0.8%	-0.6%	0.1%
3.5	-0.3%	-2.0%	-1.3%	0.0%	0.7%	0.0%	-0.5%	-0.1%	-0.2%	-1.5%	-0.5%	0.7%
4	0.3%	-0.6%	0.3%	-0.3%	-0.5%	-0.6%	-0.5%	0.4%	-0.3%	-1.7%	-0.3%	0.4%
4.5	-0.2%	0.3%	0.3%	0.3%	-0.2%	0.2%	-0.5%	-0.8%	-1.4%	-0.2%	-0.2%	0.3%
5	0.7%	0.1%	-0.1%	-0.6%	-0.1%	0.2%	-0.8%	0.2%	-0.7%	-0.2%	-0.1%	0.7%
avg	0.1%	-0.3%	-0.3%	0.0%	-0.1%	-0.2%	-0.2%	-0.2%	-0.6%	-0.9%		
max	0.7%	0.8%	0.3%	0.5%	0.7%	0.3%	0.6%	0.4%	-0.1%	-0.2%		

The higher number of features likely contributes to the minimal accuracy impact from higher reduction levels. In this study, better results for 50% pruning were observed with p values close to 4, while the worst results were associated with p values near 2. As with previous cases, the ResNet family demonstrated high volatility in results. ResNet50 and ResNet101 achieved the highest accuracy improvements of 2.1% and 3.5%. However, the fluctuation of accuracy reached 5.5% and 8.1%, with the best results obtained using higher p values and the worst results with lower p values. The ResNet family is highly volatile when high pruning levels are desired; poor norm choices may lead to substantially worse results than other networks.

Our experiment further demonstrates that the DenseNet family's performance, in terms of accuracy, is inferior to the previously mentioned networks. However, DenseNet169 has the advantage of attaining higher pruning, reaching a level of 35% while still achieving the best accuracy for a given solution. Notably, when subjected to a 50% reduction, DenseNets were the only networks unable to surpass the benchmark accuracy.

This research primarily aimed at optimizing the p-norm value for transfer learning. The ultimate objective was to minimize the feature set necessary for classification, while concurrently sustaining high accuracy levels. Distinct from many existing methods that focus on pruning networks centered on their convolutional layers, the present approach assesses the significance of features yielded by the segment of a network designated for feature extraction. This methodological shift potentially enables a more nuanced understanding of the role and impact of various features within a network. Additionally, the study undertook a comprehensive examination of a broad range of norm formulas to discern whether the order of the norm can enhance the detection of less impactful features. This scrutiny helps not only to identify and eliminate superfluous features but also to better understand the dynamics of feature interactions within complex network structures.

The current research conducted provides insight into the interplay between p-norm values, reduction levels, and accuracy across various neural network architectures. However, to fully comprehend the dynamics behind these observations, a deeper investigation is required. The study underscores the importance of considering the complexity of a network and the count of its parameters when interpreting these results.

Evidently, networks with a higher number of parameters, like VGG16, seem to display more robustness against high-level feature pruning. This implies the presence of redundancy in high-level features of these networks, enabling them to endure larger pruning levels without a substantial decrease in accuracy. Conversely, ResNet architectures display greater fluctuations in

Table 2. Accuracy results for the investigated neural network structures.

	VGG16	VGG19	ResNet50	ResNet101	DenseNet121	DenseNet169
Best accuracy (p/r)	1.3% (2/15%)	1.0% (1/10%) (3.5/10%)	2.1% (2/5%) (4.5/20%)	3.5% (3.5/25%)	0.5% (0.5/15%)	0.5% (0.5/35%) (1.5/15%)
Best accuracy for max. reduction (p)	0.6% (4.5)	0.3% (4)	1.4% (3)	1.8% (4.5)	-0.4% (0.5)	-0.4% (0.5, 1)
Worst accuracy for max. reduction (p)	-0.3% (1.5)	-1.3% (2, e)	-4.1% (0.5)	-6.3% (1.5)	-1.8% (e)	-2.3% (5)

Table 3. Number of feature neurons as output of convolutional layers and input to the classifier.

Network	VGG	ResNetV2	DenseNet121	DenseNet169
Features for classifier	512	2048	1024	1664

performance, possibly due to their deeper structures and residual connections. These characteristics might interact with the pruning process in a more complex manner, impacting the performance outcomes. Furthermore, the research acknowledges the potential impact of the number of layers in a network on the efficacy of pruning, as demonstrated by DenseNet169 and ResNet101. Given the high degree of interconnectedness in these dense networks, specific layers or features might prove more vital to the overall accuracy, rendering them more vulnerable to performance degradation with escalating pruning levels. Looking ahead, the research intends to incorporate more state-of-the-art architectures in the investigation, including Inception (Szegedy *et al*., 2016), Xception (Chollet, 2017), EfficientNet (Tan and Le, 2019), and ResNext (Xie *et al*., 2017), to broaden the understanding of feature pruning implications across a more diverse set of network structures.

8. Conclusions

This study offers an analysis and evaluation of local sensitivity analysis transfer learning high-level feature pre-pruning based on the ImageNet dataset, focusing on three network architectures: DenseNet, ResNet, and VGG. The research objective was to identify the optimal p-norm and reduction levels for these networks when tested on three datasets: CIFAR10, CIFAR100, and Fashion MNIST. Our findings reveal that the average test accuracy changes fall within a range of -2.1% to 0.8% for all networks. Additionally, a more significant drop in accuracy is evident when 40% of all neurons are removed. Remarkably, the VGG16 network outperforms the benchmark of the unpruned network in terms of average accuracy for all norms and reduction levels. Although VGG networks typically do not yield the best results, they are relatively unaffected by changes in the investigated parameters. In contrast, ResNet networks exhibit substantial accuracy volatility, achieving both the highest and lowest accuracies among the analyzed networks depending on the chosen parameters. A 20% reduction with p equal to 3 proved neutral for average accuracy, average reduction, and their combination. The most substantial overall accuracy improvements were achieved with low reduction levels and extreme p values within the applied range, such as 1 or 5.

Acknowledgment

This work was partially supported by the program *Excellence Initiative—Research University* for the AGH University of Krakow and by a grant for statutory activities from the Faculty of Physics and Applied Computer Science of the AGH University of Krakow.

References

Antwarg, L., Miller, R.M., Shapira, B. and Rokach, L. (2021). Explaining anomalies detected by autoencoders using Shapley additive explanations, *Expert Systems with Applications* **186**: 115736.

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, USA*, pp. 1800–1807.

Corrales, D.C., Ledezma, A. and Corrales, J.C. (2018). From theory to practice: A data quality framework for classification tasks, *Symmetry* **10**(7): 248.

Cuzzocrea, A. (2020). Uncertainty and imprecision in big data management: Models, issues, paradigms, and future research directions, *Proceedings of the 2020 4th International Conference on Cloud and Big Data Computing*, pp. 6–9, (virtual).

Fock, E. (2014). Global sensitivity analysis approach for input selection and system identification purposes—A new framework for feedforward neural networks, *IEEE Transactions on Neural Networks and Learning Systems* **25**(8): 1484–1495.

Gui, J., Sun, Z., Wen, Y., Tao, D. and Ye, J. (2021). A review on generative adversarial networks: Algorithms, theory, and applications, *IEEE Transactions on Knowledge and Data Engineering* **35**(4): 3313–3332.

He, K., Zhang, X., Ren, S. and Sun, J. (2016a). Deep residual learning for image recognition, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, USA*, pp. 770–778.

He, K., Zhang, X., Ren, S. and Sun, J. (2016b). Identity mappings in deep residual networks, *in* B. Leibe *et al.* (Eds.), *Computer Vision—ECCV 2016*, Springer, Cham, pp. 630–645.

Hidaka, A. and Kurita, T. (2017). Consecutive dimensionality reduction by canonical correlation analysis for visualization of convolutional neural networks, *Proceedings of the ISCIE International Symposium on Stochastic Systems Theory and Its Applications, Fukuoka, Japan*, Vol. 2017, pp. 160–167.

Huang, G., Liu, Z., Van Der Maaten, L. and Weinberger, K.Q. (2017). Densely connected convolutional networks, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, USA*, pp. 2261–2269.

Huang, Y.-C., Tung, Y.-S., Chen, J.-C., Wang, S.-W. and Wu, J.-L. (2005). An adaptive edge detection based colorization algorithm and its applications, *MULTIMEDIA'05: Proceedings of the 13th Annual ACM International Conference on Multimedia, Singapore*, DOI: 10.1145/1101149.1101223.

Jeczmionek, E. and Kowalski, P.A. (2021). Flattening layer pruning in convolutional neural networks, *Symmetry* **13**(7): 1147.

Kahani, N., Bagherzadeh, M., Cordy, J.R., Dingel, J. and Varró, D. (2019). Survey and classification of model transformation tools, *Software & Systems Modeling* **18**(4): 2361–2397.

Keras (2023). *Keras Applications*, `https://keras.io/api/applications/`.

Kowal, M., Skobel, M. and Nowicki, N. (2018). The feature selection problem in computer-assisted cytology, *International Journal of Applied Mathematics and Computer Science* **28**(4): 759–770, DOI: 10.2478/amcs-2018-0058.

Kowalski, P.A. and Kusy, M. (2017). Sensitivity analysis for probabilistic neural network structure reduction, *IEEE Transactions on Neural Networks and Learning Systems* **29**(5): 1919–1932.

Kowalski, P.A. and Kusy, M. (2018). Determining significance of input neurons for probabilistic neural network by sensitivity analysis procedure, *Computational Intelligence* **34**(3): 895–916.

Krizhevsky, A. (2009). Learning multiple layers of features from tiny images, `https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`.

Kusy, M. and Kowalski, P.A. (2018). Weighted probabilistic neural network, *Information Sciences* **430**: 65–76.

Kusy, M. and Zajdel, R. (2021). A weighted wrapper approach to feature selection, *International Journal of Applied Mathematics and Computer Science* **31**(4): 685–696, DOI: 10.34768/amcs-2021-0047.

Lundberg, S.M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions, *Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, USA*, pp. 4768–4777.

Mishkin, D., Sergievskiy, N. and Matas, J. (2017). Systematic evaluation of convolution neural network advances on the imagenet, *Computer Vision and Image Understanding* **161**: 11–19.

Riesz, F. (1910). Untersuchungen über Systeme integrierbarer Funktionen, *Mathematische Annalen* **69**(4): 449–497.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. and Fei-Fei, L. (2014). Imagenet large scale visual recognition challenge, *International Journal of Computer Vision* **115**(2015): 211–252.

Saltelli, A., Annoni, P., Azzini, I., Campolongo, F., Ratto, M. and Tarantola, S. (2010). Variance based sensitivity analysis of model output. design and estimator for the total sensitivity index, *Computer Physics Communications* **181**(2): 259–270.

Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M. and Tarantola, S. (2008). *Global Sensitivity Analysis: The Primer*, John Wiley & Sons, New York.

Saltelli, A., Tarantola, S., Campolongo, F. and Ratto, M. (2004). *Sensitivity Analysis in Practice: A Guide to Assessing Scientific Models*, Wiley, New York.

Saltelli, A., Tarantola, S. and Chan, K. (2012). A quantitative model-independent method for global sensitivity analysis of model output, *Technometrics* **41**(1): 39–56.

Saura, J.R. (2021). Using data sciences in digital marketing: Framework, methods, and performance metrics, *Journal of Innovation & Knowledge* **6**(2): 92–102.

Shi, D., Yeung, D.S. and Gao, J. (2005). Sensitivity analysis applied to the construction of radial basis function networks, *Neural Networks* **18**(7): 951–957.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, USA*.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z. (2016). Rethinking the inception architecture for computer vision, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, USA*, pp. 2818–2826.

Tan, M. and Le, Q. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks, *International Conference on Machine Learning, Long Beach, USA*, pp. 6105–6114.

Tzeng, F.-Y. and Ma, K.-L. (2005). Opening the black box-data driven visualization of neural networks, *VIS 05: IEEE Visualization, 2005, Minneapolis, USA*, pp. 383–390.

Xiao, H., Rasul, K. and Vollgraf, R. (2017). Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms, *arXiv:* 1708.07747.

Xie, S., Girshick, R., Dollar, P., Tu, Z. and He, K. (2017). Aggregated residual transformations for deep neural networks, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, USA*, pp. 5987–5995.

Yamashita, R., Nishio, M., Do, R.K.G. and Togashi, K. (2018). Convolutional neural networks: An overview and application in radiology, *Insights into Imaging* **9**(4): 611–629.

Zurada, J.M., Malinowski, A. and Usui, S. (1997). Perturbation method for deleting redundant inputs of perceptron networks, *Neurocomputing* **14**(2): 177–193.

Ernest Jeczmionek is a PhD candidate specializing in CNN pruning using sensitivity analysis. He holds an MSc degree in computer science, with the focus on the parallelization of the complete gradient clustering algorithm. In addition to his academic pursuits, Ernest serves as a senior deep learning engineer. His extensive industry experience includes the development of autonomous drones and conducting quantitative risk analysis for credit default swaps (CDSs).

Piotr A. Kowalski is an associate professor at the Faculty of Physics and Applied Computer Science, AGH University of Krakow, and at the Systems Research Institute of the Polish Academy of Sciences (PAS). He received his MSc in teleinformatics (with honours) and automatic control (with honours) from the Cracow University of Technology in 2003 and his PhD in data analysis from the PAS in 2009. In 2018, he obtained his DSc degree from the Systems Research Institute of the PAS. His research interests are in the area of information technology and are focused on intelligent methods (neural networks, fuzzy systems and nature-inspired algorithms) with applications to complex systems and knowledge discovery.

Received: 5 May 2023
Revised: 11 July 2023
Accepted: 9 August 2023