Neural Criticality: Validation of Convolutional Neural Networks

Václav Diviš[1]
Marek Hrúz
[1]Supported by ARRK Engineering and the University of West Bohemia.
Faculty of Applied Sciences, University of West Bohemia
Pilsen, CZE, 306 14
divisvaclav@gmail.com

Abstract

The black-box behavior of Convolutional Neural Networks is one of the biggest obstacles to the development of a standardized validation process. Methods for analyzing and validating neural networks currently rely on approaches and metrics provided by the scientific community without considering functional safety requirements. However, automotive norms, such as ISO26262 and ISO/PAS21448, do require a comprehensive knowledge of the system and of the working environment in which the network will be deployed. In order to gain such a knowledge and mitigate the natural uncertainty of probabilistic models, we focused on investigating the influence of filter weights on the classification confidence in Single Point Of Failure fashion. We laid the theoretical foundation of a method called the Neurons' Criticality Analysis. This method, as described in this article, helps evaluate the criticality of the tested network and choose related plausibility mechanism.

1 Introduction and motivation

The need to understand and rely on the inference processes of Convolution Neural Networks (CNNs) grows in importance since probabilistic models are being integrated in autonomous vehicles (Tesla 2019),(Mobileye 2020), where the SW development follows functional safety standards and on which lives may depend.

The transparency, evaluation criteria and types of explanations of the achieved results face low interpretability (Belle and Papantonis 2020) due to the increasing complexity of the models used.

Furthermore, as recently shown by various adversary attack examples (Liu et al. 2016),(Brown et al. 2017),(Eykholt et al. 2018), even a small perturbation in the input image can cause a major change to the algorithm's decision. In addition, the current leading norms (ISO26262:2018, ISO/PAS21448:2019) do not define validation process nor metrics related to the probabilistic model (BMW 2019). Finally, to our knowledge. there isn't any method similar to the Software Criticality Analysis (SWCA) or MC/DC module test in the CNN field (Salay and Czarnecki 2018) which can analyze potential Single Point Of Failure (SPOF). All of this motivated us to define a methodology and understandable metrics according to which State-Of-The-Art (SOTA) CNNs can be analyzed and the achieved results validated.

In this work we introduced a new metric called criticality, which can be either assigned to neurons (in CNN's terminology, those are referred to as ”filters' weight”) or to layers. We verified the importance and plausibility of the proposed criticality as a possible safety metric by conducting two experiments. We designed and implemented a method called Neurons' Criticality Analysis (NCA) and tested it on four image classification models.

The results of our experiment are extensively discussed at the end of this work, where we also summarized the pros and cons of the presented metric and methodology and highlighted the use-cases we intend to investigate in the future.

2 Previous work

As mentioned in (Belle and Papantonis 2020), data-driven techniques struggle to be robust against domain shift, data corruption and input space perturbation. The robustness can be influenced by three aspects: architecture, training dataset and optimization algorithm. It is not among the goals of this paper to give a comprehensive overview of all three topics, but to highlight the most influential milestones which inspired our experiment.

2.1 Classification via CNN

As a first step, we looked at the straightforward VGG16 architecture (Simonyan and Zisserman 2014), where BatchNorm layer, 3×3 CONV with stride and padding equal to 1 and 2×2 MAX POOL with stride 2 are several times repeated. As a second step, we reviewed the ResNet (He et al. 2016a) architecture, which differentiates from the previous models mainly by having residual blocks (He et al. 2015) and using 1x1 convolutional operation. We tested ResNet50V2 (He et al. 2016b), which is the successor of the original ResNet with enhanced residual connections for a smoother information propagation. The aim of residual connections is to create additional information flow and to learn additive residual function. As explained in (Lin, Chen, and Yan 2014) and further investigated in (Szegedy et al. 2014), the concept known as ”Network In Network” (NIN) allows reducing the filter's dimensionality and increasing the models' non-linearity. With the combination of computationally more expensive 3×3 and 5×5 convolution and parallel branches the extraction of features from different scales can be achieved simultaneously. This kind of blocks are often called ”projection layer” (Li et al. 2018).

Since building a model in ResNet fashion comes at computational cost, a family of mobileNets emerged. The standard convolution operation which was used up to this point was, in case of MobileNet v1 (Howard et al. 2017), replaced by depthwise separable convolution. It was shown that standard convolution can be split into depthwise and pointwise convolutions, which decreases the number of operations by a square of the spatial dimension of the used filter kernel. Further improvements were carried out by Sandle et al. in MobileNet v2 (Sandler et al. 2018), where residual connections between the cells and the expand/projection layers were added.

A research done by Google discovered a new method of scaling the CNN. The goal of this optimization search was to find scaling coefficients (network width, depth and resolution) with respect to the accuracy and amount of operations. EfficientNet (Tan and Le 2019) was designed to demonstrate the effectiveness of this scaling method and achieved SOTA accuracy on the ImageNet dataset in 2019.

The need to understand how the decision process is made lead us to several papers (Simonyan, Vedaldi, and Zisserman 2013),(Zeiler and Fergus 2014),(Bach et al. 2015),(Shrikumar, Greenside, and Kundaje 2017),(Sundararajan, Taly, and Yan 2017), where different visualization methods are described. A comprehensive overview of the methods and their goals can be found in (Rafegas et al. 2019), where Rafegas et al. also presented a novel method of quantifying the neurons’ selectivity to color and class.

2.2 Safety related issues

The aleatoric and epistemic uncertainty (Kendall and Gal 2017) of probabilistic models are currently under in-depth study. The source of the aleatoric uncertainty is brought by the randomness contained within the training-set, whereas the epistemic uncertainty is caused by the lack of the model’s knowledge. Bayesian machine learning (Neal 2012) approach allows propagating the intermediate covariances to the final layer and quantifying the hypothesis uncertainty (Graves 2011), (Shridhar, Laumann, and Liwicki 2019). Such an approach requires time-consuming training, and, for the moment, models do not achieve the expected accuracy. An additional one-shot approach uses Monte Carlo dropout during inference in order to sample a subset of networks, build statistics and calculate the thereof resulting uncertainty (Gal and Ghahramani 2016). This improves the demands on the inference time to reasonable limits and can therefore be applicable in automotive.

Many papers additionally address the problem of data-driven ML algorithms and how to incorporate safety mechanism in order to monitor the prediction uncertainty (Cheng 2020), (Lakshminarayanan, Pritzel, and Blundell 2017), (DeVries and Taylor 2018). Several uncertainty estimation methods were lately evaluated by Henne et al. (Henne et al. 2020) with respect to functional safety.

Our approach is driven by ISO/PAS21448 SOTIF (ISO2019 2019), ISO2626 (ISO 2011) norms, which lack validation-unambiguity, and by the conclusion that only 40% of the current automotive verification/validation methods can be transferred to ML application (Salay and Czarnecki 2018). We mostly considered the SOTIF norm, which is an extension of the well-known ISO2626 norm and provides a guidance (recommended activities) on applicable design, verification and validation measures in order for the product to be norm-compliant. The goal of the recommended activities is to maximize the area of known safe scenarios and minimize the unknown or unsafe areas by applying technical measures.

2.3 Adversary attacks

The original idea of adversary attack is to introduce a small perturbation to an input image so that the original class doesn’t have the highest confidence and the adversary noise stays unrecognized to the human perception system.

One of the first attacks used the Fast Sign Gradient Descent (FSGD) method (Goodfellow, Shlens, and Szegedy 2015), which calculates the gradient of a model’s loss function with respect to the input image and ground-truth label and either adds or subtracts a small portion of it, depending whether the gradient was positive or negative. Additional papers proposed a general and large perturbation attack algorithm of physical objects considering spatial constraints and physical limits on imperceptibility, (Brown et al. 2017), (Eykholt et al. 2018). Ian Goodfellow summarized additional weaknesses of the classification task in (Goodfellow 2018).

The defense mechanism started to be deeply investigated in the work of Lie et al. (Liu et al. 2016), which shows a comprehensive experiment of different ResNets Architectures trying to resist non-target adversarial images and states that ResNet-152 has a 0% resistivity. The explanation to that phenomenon is still in the open research area (Brown et al. 2017), but one of the latest works (Song et al. 2018) shows promising results and mentions defending methods, showing that the robustness against different attacks can improve.

3 Analysis methodology

In this section we defined the metric and methodology related to manipulating and analyzing the CNN’s decision process. We took the inspiration for the Neurons’ Criticality Analysis (NCA) method from the Software Criticality Analysis (SWCA). The SWCA is a method which divides modules of any action chain between critical (the SPOF of which could have fatal consequences) and non-critical. In case of an automotive SW component, the SWCA is carried out by analyzing the signal flow from the actuator to the sensor, while heuristically justifying the signal’s non-criticality. In order to investigate if the decision of any CNN can be significantly influenced by the SPOF of the filter connections, the idea of an approach similar to SWCA was explored. From now on we will refer to filter connection as a neuron, since the principle can be generally applied to FC, CONV and Depth-wise CONV layer,

3.1 Criticality metric

Firstly, we denote the analyzed convolutional neural network as N, which consists of a set of layers L and contains weights W and biases b. Secondly, we introduce the criticality metric according to Equation 1 and the evaluation algorithm 1 which calculates the criticality for a given input image x_i, belonging to class i, drawn from a test set $\mathcal{X}$.

$$f_{cr} = \begin{cases} \hat{y}_i - \hat{y}_{mi}, & \text{if } f_m(x_i) : (\hat{y}_i - \hat{y}_{mi}) \geq \tau \\ \frac{1}{1-\hat{y}_{mj}}, & \text{if } f_m(x_i) : \hat{y}_{mi} \leq \hat{y}_{mj} \text{ and } \hat{y}_{mj} < 0.5 \\ 2, & \text{if } f_m(x_i) : \hat{y}_{mi} \leq \hat{y}_{mj} \text{ and } \hat{y}_{mj} \geq 0.5 \\ 0, & \text{otherwise,} \end{cases} \tag{1}$$

where f_{cr} returns a criticality with domain $[0, 2]$ for a given CNN which is masked, $f_m(x_i) \subset N$. The masking of a CNN is carried out by setting neurons' weights to zero. In case of convolution, all the values of a filter are set to zero. A different kind of error modeling would lead to extensive permutation and was therefore not further investigated.

The term $\hat{y}_{mi}$ denotes the masked network's prediction confidence of ground-truth class i, whereas the predicted confidence value $\hat{y}_{mj}$ belongs to another class j. In the first case of f_{cr}, the difference between the non-masked predicted confidence $\hat{y}_i$ and the masked one $\hat{y}_{mi}$ is taken as metric, by considering the parametrizable "criticality" τ with domain $[0, 1]$.

In the second case of f_{cr}, the network missed the ground-truth class and predicted a different one. Since this misclassification can have severe consequences, we define the criticality measure as the proportion of 1 and difference between maximum likelihood and predicted confidence $\hat{y}_{mj}$. The denominator will always result in a number greater than 1, consequently ensuring the distinguishability of neurons which have class-changing ability. Experiments in the early phase showed that criticality can reach multi-digit number and therefore we decided in the third case of f_{cr} to clip its maximum to 2, so that the results remain tractable. For all other cases we set the criticality to 0. This covers the cases where the network predicted the right class with negligible deterioration of the confidence ($< \tau$).

It can occur that by masking a neuron the decision likelihood of the correct class will increase, which will result in a negative criticality. In this case, we refer to this neuron as anti-critical to the related class i and calculate its criticality according to Equation 2. However, it should be noted that the anti-criticality will be computed only in case $\tau = 0$ and it doesn't exclude the criticality of the neuron for a different class j.

$$f_{anti\_cr} = \hat{y}_i - \hat{y}_{mi}, \text{if } f_m(x_i) : (\hat{y}_i - \hat{y}_{mi}) < 0 \tag{2}$$

3.2 Analysis algorithm

We define the task of NCA as the analysis of the neurons' contribution to the classification hypothesis which can be seen as equivalent to the Single Point Of Failure analysis. If all neurons are active, the resulting hypothesis is strong h_{str}, whereas in case a certain amount of neurons have been excluded from the decision, the hypothesis is considered weakened h_{weak}. The neuron's criticality observation of the weakened hypothesis has to be done for every image and class within a test set. The algorithm is described in Algorithm 1.

Algorithm 1: NCA algorithm

```
Input: Criticality threshold $\tau$
Output: Neural criticality statistics for $\mathcal{X}_{\rangle}$
Data: Let $\mathcal{X}$ be a testing set, $i$ a tested class, $N$ the
      analyzed CNN, $k$ the number of filters in a
      layer $L$ and $f_{cr}$ is the criticality function
for image $x_i \in \mathcal{X}$ do
    $\hat{y}_i = calculate\_conf(N, x_i)$
    $cls_i = predict(N, x_i)$
    for every $L$ in $N$ do
        for every $k$ in layer $L$ do
            $mask\_neuron(k)$
            $\hat{y}_{mi} = calculate\_conf(N, x_i)$
            $cls_{mi} = predict(N, x_i)$
            $criticality = f_{cr}(\hat{y}_i, \hat{y}_{mi}, cls_i, cls_{mi})$
```

4 Experiments

The motivation behind testing different network architectures was to see the influence of models' chronological improvements on the decision stability, such as residual connections, depthwise convolution and scaling. We therefore evaluated VGG16, Resnet50V2, MobileNetV2 and EfficientNetB0, all pre-trained Keras models on ImageNet (Deng et al. 2009). We chose two classes, "street sign" and "mountain bike", in order to evaluate the criticality. For each class, 150 samples were taken. All samples had ground-truth confidence higher than 0.8 so that we ensured that kernels' responses would be highly excitated. Adversary samples were generated by non-target FSGD method until either achieving a confidence greater than 0.5 or ending after 20 iterations. For all tests we set the criticality threshold τ to 0.0, which allows, as described in Section 3.1, the algorithm to measure and visualize the criticality of all neurons and distinguish between critical and anti-critical ones. In practice, the threshold should be justifiable via hazard and risk assessment and will be presumably higher than 0.0.

4.1 Neural criticality

As a first step, we gathered statistics of every neuron as described in Algorithm 1 with $\tau = 0.0$. As a second step, we normalized the list of hypotheses over the number of layers' neurons and highlighted in red the layers for which masking at least one neuron caused a drop of confidence by 0.5 and more or lead to misclassification of the predicted class. It is noticeable that in Figures 3 and 4, especially the first projection layers have very high criticality. This confirms the sparsity theory of projection layers (Szegedy et al. 2014, 2016), which states that projection layers are helpful in terms of higher space feature extraction, whereas our experiment shows that they cause an increase of criticality.

VGG16 (Figure 1) and ResNet50v2 (Figure 2) have on the other hand an average criticality spread over all layers.

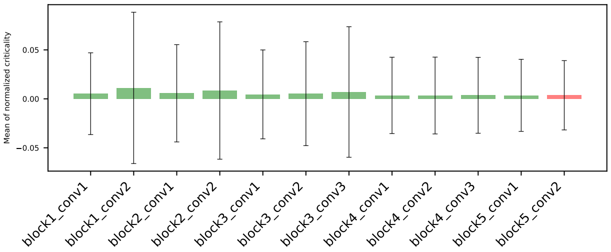


Figure 1: Results of NCA on VGG16 showed that straight architecture, without any projection and residual layers, has the lowest criticality.

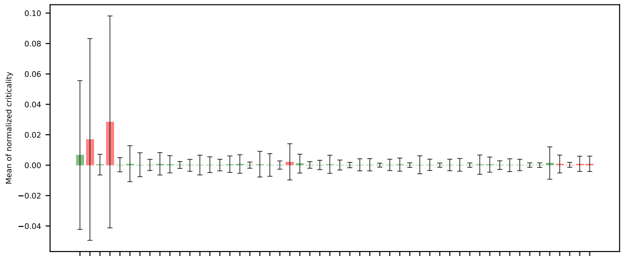


Figure 2: The neural criticality of ResNet50v2 peaked in the early projection layers, but remained overall very small.

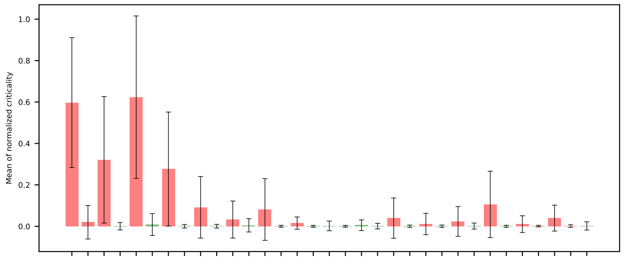


Figure 3: Test on MobileNetv2 architecture showed a higher instability caused by projection layers. The criticality of several neurons exceeds 1.0, which means that masking just one neuron can cause misclassification.

The experiments' results shown in this work are only related to the "mountain bike" class. The results for the fairly simple "traffic sign" class backed up the intuition about simple features being predominantly filtered in early layers, since their criticality raised significantly. We advise the reader to visit our GitHub, where additional figures and stored statistics for both classes can be found.

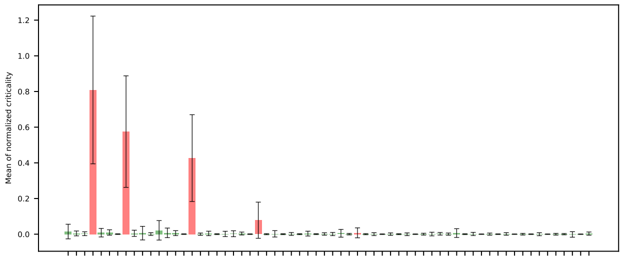


Figure 4: This figure shows that EfficientNetB0 contains many neurons which criticality is exceeds 1.0. Because the EfficientNetB0 architecture contains, in its early stage, many projection layers with only few neurons, we removed the overlapping x-axis labels.

4.2 Network stability

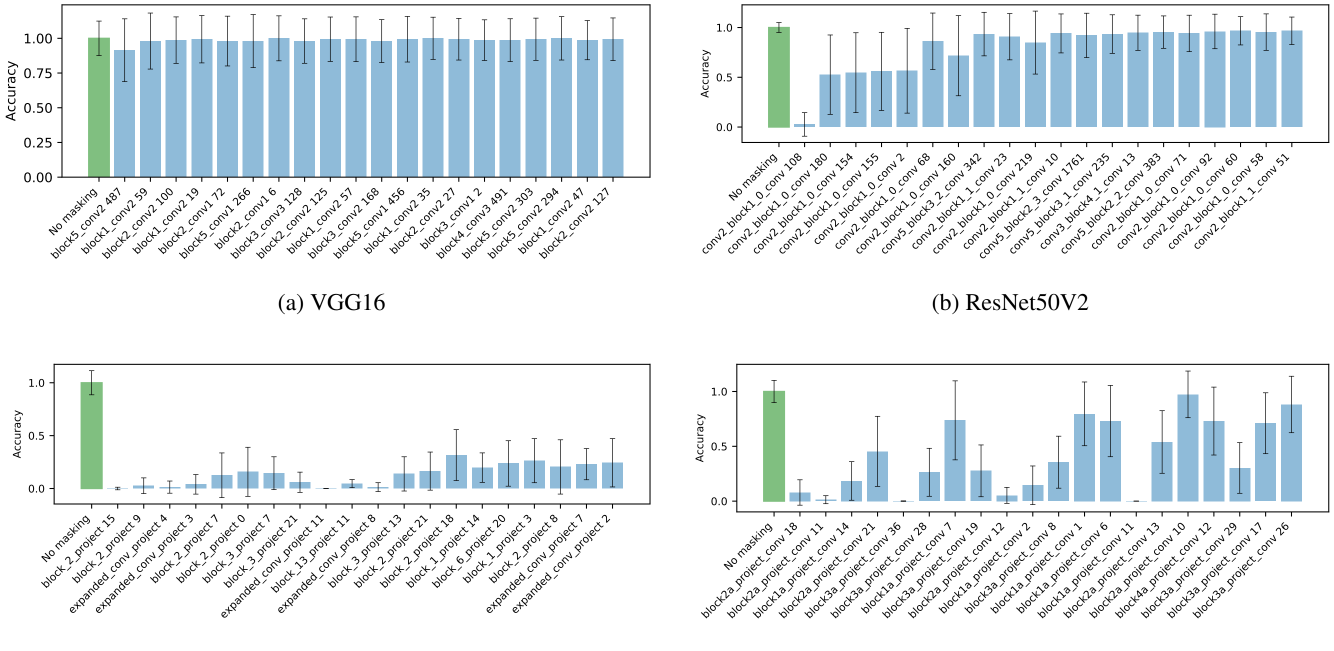
To further evaluate the beneficial effects of neural criticality, we conducted a stability experiment with n most critical neurons (derived from NCA results) on original and adversary datasets. We gradually masked the n most critical neurons and calculated the mean and standard deviation of the model's accuracy on the aforementioned test set. The intuition behind this test was that the mean accuracy increases with respect to the criticality of lower neurons, and hence proves our analysis' reliability. In our text we refer to this approach as Network Stability Analysis.

As can be seen in Figure 5, gradually masking the 20 most critical neurons has a major influence on the accuracy only in case of MobileNetv2 and EfficientNetB0. VGG16 and ResNet50V2, on the other hand, show a high accuracy stability. Figure 6 shows the raising tendency of MobileNets' accuracy with respect to lower neurons criticality, reaching a mean accuracy of 0.8 approximately at the 50th most critical neuron.

Results of the accuracy on adversary dataset didn't confirm the hypothesis that critical neurons are the only neurons allowing malicious adversary attacks. As can be seen in Figure 9, masking the critical neuron generally doesn't improve the accuracy. On the other hand, several neurons lifted the ground-truth class accuracy. The awareness of such neurons could lead to on-the-fly diagnoses, where masking a combination of specific neurons (e.g. only for x^{th} frame, which would be excluded from the classification or detection task) would uncover irregularities in inference process, e.g. adversary attack. It has to be mentioned that only critical neurons from projection layers (MobileNetV2 and EfficientNetB0) have such an ability, but they have to be chosen with respect to the mean and standard deviation of the calculated accuracy. Other models sensitivity to adversary noise are plotted in Figure 8.

5 Conclusion

At the beginning of this work in Chapter 2 we pointed out the current functional safety issues and open research area related to convolutional neural networks. In Section 1 we described our motivation related to autonomous driving and



(a) VGG16

(b) ResNet50V2

(c) MobileNetv2

(d) EfficientNetB0

Figure 5: Results of all models' accuracy stability related to the 20 most critical neurons, evaluated on a normal dataset. The models' accuracy without masking can be found on the first position, marked in green. Going to the right, the criticality of the neurons decreases and hence the mean accuracy increases.

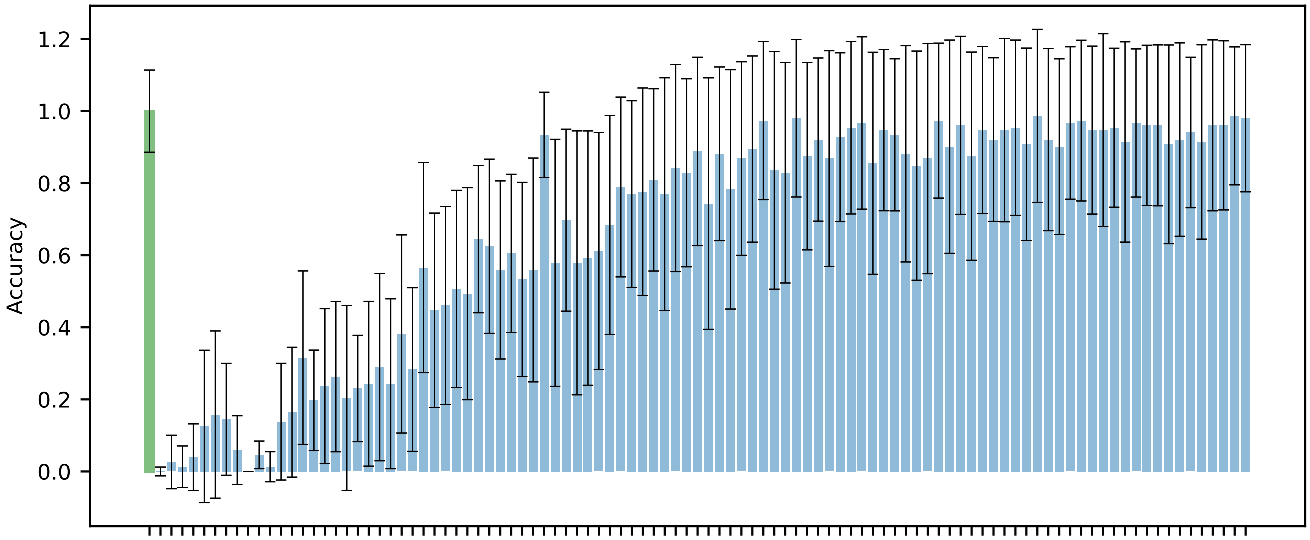


Figure 6: Accuracy stability on normal dataset (for class "mountain bike"), showing a gradual increase of accuracy with respect to decreasing neurons' criticality in case of MobileNetv2 model.

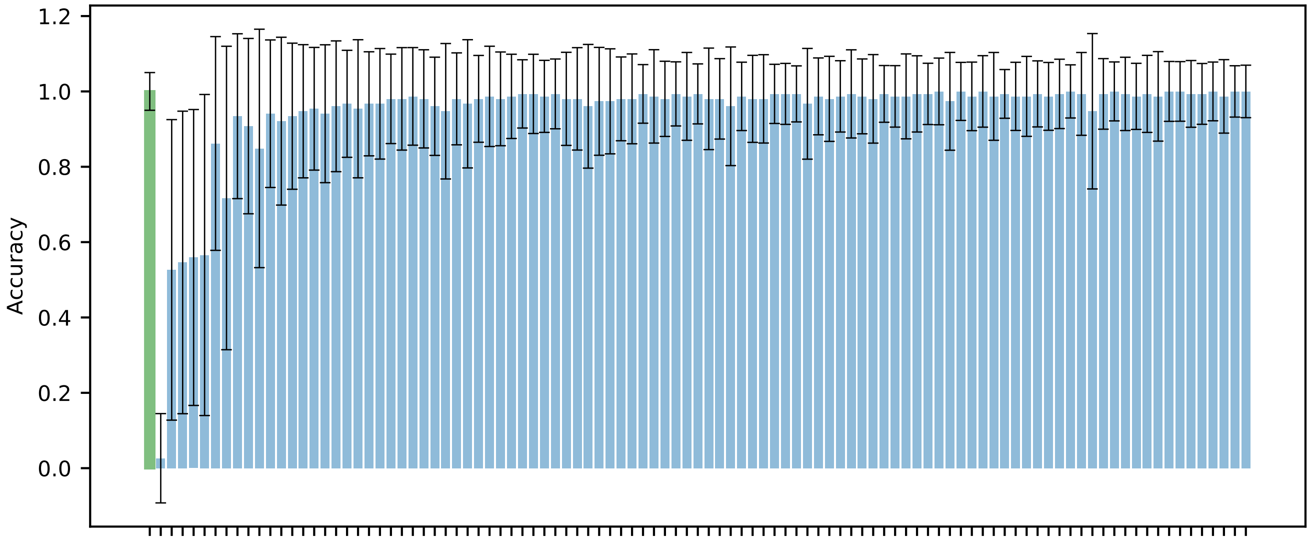
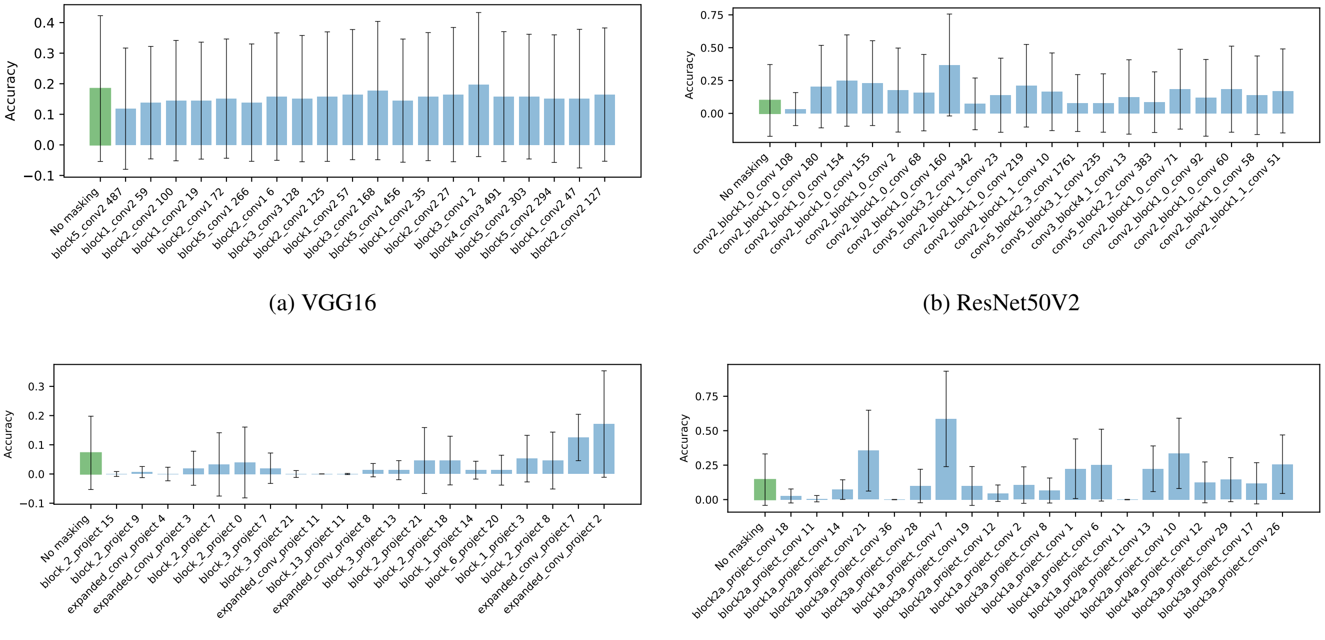


Figure 7: Compared to MobileNetv2 or EfficientNetB0, the ResNet50V2 architecture has a higher accuracy stability, which was explained by missing projection layers in Chapter 4.1. Deeper investigation showed that for the ”moutain bike” class, all of the few critical neurons can be found in layer block1_0_conv_c0.



(a) VGG16

(b) ResNet50V2

(c) MobileNetv2

(d) EfficientNetB0

Figure 8: Results of accuracy stability related to the 20 most critical neurons for all models, evaluated on adversary dataset. It is obvious that different results of generating adversary attacks was achieved since initial models accuracy differs. The VGG16 model shows minimal accuracy fluctuation, whereas more modern models contains neurons with higher sensitivity to adversary noise.

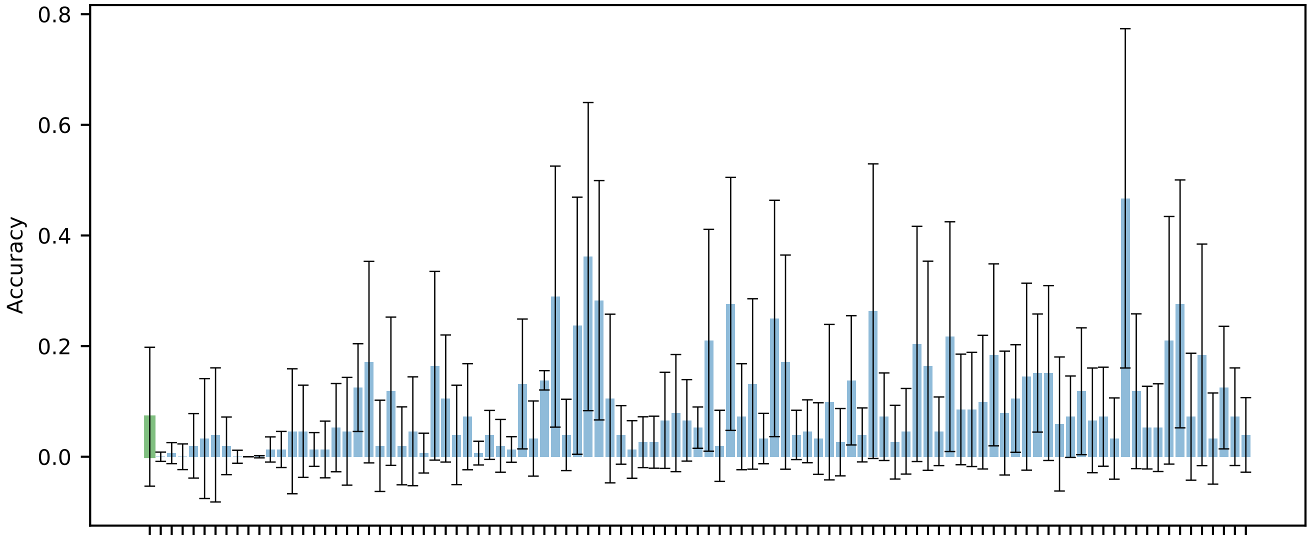


Figure 9: MobileNetV2's accuracy stability of the 100 most critical neurons, taken from the analysis of a normal dataset and evaluated on an adversary dataset. Neurons with increased accuracy could be further used for diagnosis purposes, but have to be chosen with respect to both mean and standard deviation of the resulting accuracy. In such a diagnostic case, masking multiple neurons could be desirable and would lead to higher diagnoses accuracy.

missing validation process. Further, as outlined in Section 3, we introduced a new metric and auxiliary analysis method which we implemented and verified on four classification CNNs in Chapter 4.

We dedicated a great part of our work to introducing and testing an innovative method, the Neurons' Criticality Analysis. The outcome of this analysis was a comprehensive report diagram depicting the criticality of each layer and each neuron of evaluated model. The domain of criticality is $[-1, 2]$. We discussed that masking neurons with negative criticality can also have a positive influence on the model's decision confidence. We called this behavior ”anti-critical”. The inter-class anti-critical neurons could hypothetically be removed from the decision process. This idea led us to the conclusion that the correlation between the neurons removed during the pruning process and the anti-critical neurons discovered via NCA should be further investigated.

We claimed that using spatial aggregation via projection layers may on the one hand improve the high dimensional feature representation(Szegedy et al. 2016), but on the other hand creates very critical dense connections, especially in the shallow layers, as we pointed out in Section 4.1. From functional safety point of view this isn't necessarily negative, since the plausibility function could be applied to only a concentrated area of neurons. In addition, some critical neurons showed the ability to increase mean accuracy on adversary dataset, which could be used in order to discover adversary attacks and irregularities during inference. We hypothesize that an equilibrium between the position of the first projection layer, number of critical neurons and models' accuracy should be further investigated.

As aforementioned, the purpose of NCA is to identify all critical neurons. With further measures, the mean and standard deviation of the criticality should be decreased and the flawless calculation of the neuron should be ensured. Concretely this can be achieved by several approaches, such as:

- fine-tuning of the model with deterministic dropout and loss which will incorporate the layers criticality
- plausibility check of the critical neurons or layers or redundant computational branch results
- storage of the neurons' weights and biases in two places in RAM and comparing them
- introduction of deconvolutional layers in order to compute and evaluate the original inputs over critical connections

Our method can also be used for Out-of-Distribution detection, where instead of randomly sampling sub-networks predictions, as it is done by MC dropout, deterministic dropout would be based on several highly critical neurons for every class. Such an approach would decrease the computational demand and arguably increase the reliability and transparency of such a network. In order to encourage additional experiments and deeper explorations, we published our code and supplement results on GitLab [1].

6 Acknowledgment

The work has been supported by the grant of the University of West Bohemia, project No. SGS-2019-027.

[1] https://gitlab.com/divisvaclav/cnn_eval_tool/-/tree/wo_gui_branch

References

Bach, S.; Binder, A.; Montavon, G.; Klauschen, F.; Müller, K.-R.; and Samek, W. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* 10(7): e0130140.

Belle, V.; and Papantonis, I. 2020. Principles and Practice of Explainable Machine Learning.

BMW, D. e. a. 2019. Safety First for Automated Driving (SaFAD) Safety First for Automated Driving (SaFAD). https://www.daimler.com/innovation/case/autonomous/safety-first-for-automated-driving-2.html. Accessed: 2020-06-08.

Brown, T. B.; Mané, D.; Roy, A.; Abadi, M.; and Gilmer, J. 2017. Adversarial patch. *arXiv preprint arXiv:1712.09665* .

Cheng, C.-H. 2020. Safety-Aware Hardening of 3D Object Detection Neural Network Systems. *arXiv preprint arXiv:2003.11242* .

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 248–255. IEEE.

DeVries, T.; and Taylor, G. W. 2018. Learning confidence for out-of-distribution detection in neural networks. *arXiv preprint arXiv:1802.04865* .

Eykholt, K.; Evtimov, I.; Fernandes, E.; Li, B.; Rahmati, A.; Xiao, C.; Prakash, A.; Kohno, T.; and Song, D. 2018. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1625–1634.

Gal, Y.; and Ghahramani, Z. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, 1050–1059.

Goodfellow, I. 2018. Defense Against the Dark Arts: An overview of adversarial example security research and future research directions. *arXiv preprint arXiv:1806.04169* .

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and Harnessing Adversarial Examples.

Graves, A. 2011. Practical variational inference for neural networks. In *Advances in neural information processing systems*, 2348–2356.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep Residual Learning for Image Recognition.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016a. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016b. Identity mappings in deep residual networks. In *European conference on computer vision*, 630–645. Springer.

Henne, M.; Schwaiger, A.; Roscher, K.; and Weiss, G. 2020. Benchmarking Uncertainty Estimation Methods for Deep Learning With Safety-Related Metrics. In *SafeAI@ AAAI*, 83–90.

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* .

ISO. 2011. Road vehicles – Functional safety.

ISO2019. 2019. Road vehicles — Safety of the intended functionality. RFC 1654, RFC Editor. URL http://www.rfc-editor.org/rfc/rfc1654.txt.

Kendall, A.; and Gal, Y. 2017. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, 5574–5584.

Lakshminarayanan, B.; Pritzel, A.; and Blundell, C. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, 6402–6413.

Li, Z.; Peng, C.; Yu, G.; Zhang, X.; Deng, Y.; and Sun, J. 2018. DetNet: A Backbone network for Object Detection.

Lin, M.; Chen, Q.; and Yan, S. 2014. Network In Network.

Liu, Y.; Chen, X.; Liu, C.; and Song, D. 2016. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770* .

Mobileye. 2020. Advanced Technologies - Mobileye Future of Mobility - Advanced Technologies. https://www.mobileye.com/. Accessed: 2020-11-13.

Neal, R. M. 2012. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media.

Rafegas, I.; Vanrell, M.; Alexandre, L. A.; and Arias, G. 2019. Understanding trained CNNs by indexing neuron selectivity. *Pattern Recognition Letters* .

Salay, R.; and Czarnecki, K. 2018. Using machine learning safely in automotive software: An assessment and adaption of software process requirements in ISO 26262. *arXiv preprint arXiv:1808.01614* .

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4510–4520.

Shridhar, K.; Laumann, F.; and Liwicki, M. 2019. A comprehensive guide to bayesian convolutional neural network with variational inference. *arXiv preprint arXiv:1901.02731* .

Shrikumar, A.; Greenside, P.; and Kundaje, A. 2017. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 3145–3153. JMLR. org.

Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* .

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* .

Song, C.; He, K.; Wang, L.; and Hopcroft, J. E. 2018. Improving the Generalization of Adversarial Training with Domain Adaptation. *arXiv preprint arXiv:1810.00740* .

Sundararajan, M.; Taly, A.; and Yan, Q. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 3319–3328. JMLR. org.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2014. Going Deeper with Convolutions.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2818–2826.

Tan, M.; and Le, Q. V. 2019. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *arXiv preprint arXiv:1905.11946* .

Tesla. 2019. Autopilot and Full Self-Driving Capability Autopilot and Full Self-Driving Capability. https://www.tesla.com/support/autopilot. Accessed: 2020-06-08.

Zeiler, M. D.; and Fergus, R. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*, 818–833. Springer.