Beyond Test Accuracy: The Effects of Model Compression on CNNs

Adrian Schwaiger, Kristian Schwienbacher, Karsten Roscher

Fraunhofer IKS
{firstname.lastname}@iks.fraunhofer.de

Abstract

Model compression is widely employed to deploy convolutional neural networks on devices with limited computational resources or power limitations. For high stakes applications, such as autonomous driving, it is, however, important that compression techniques do not impair the safety of the system. In this paper, we therefore investigate the changes introduced by three compression methods – post-training quantization, global unstructured pruning, and the combination of both – that go beyond the test accuracy. To this end, we trained three image classifiers on two datasets and compared them regarding their performance on the class level and regarding their attention to different input regions. Although the deviations in test accuracy were minimal, our results show that the considered compression techniques introduce substantial changes to the models that reflect in the quality of predictions of individual classes and in the salience of input regions. While we did not observe the introduction of systematic errors or biases towards certain classes, these changes can significantly impact the failure modes of CNNs and thus are highly relevant for safety analyses. We therefore conclude that it is important to be aware of the changes caused by model compression and to already consider them in the early stages of the development process.

1 Introduction

Deep Neural Networks (DNNs) enable many complex applications such as autonomous vehicles or automated manufacturing processes. Especially for perception tasks, Convolutional Neural Networks (CNNs) have shown impressive results and have been adopted widely. However, to achieve a high degree of performance, these networks often have millions of parameters that require significant computing power for inference, impeding the deployment on edge or low-power mobile devices (Cheng et al. 2018). One way to approach this problem is to compress the models, e.g., via pruning – i.e. removing parts of the network with a low contribution to the predictions – or quantization – i.e. reducing the number of bits required for each parameter. These methods allow to reduce the memory footprint, increase computational efficiency, and in turn also decrease the power demands, enabling the deployment of DNNs on low-power devices. Compressing models using pruning or quantization techniques can significantly reduce their size without severely impacting the overall performance regarding test accuracy. However, especially for safety-critical applications test accuracy on its own is not sufficient and underlying negative effects, e.g., on the long tail of data distributions have been shown (Hooker et al. 2021). Furthermore, since model compression is often not explicitly addressed in development and assurance frameworks, such as Assurance of Machine Learning for use in Autonomous Systems (AMLAS) (Hawkins et al. 2021), an introduction of additional failure modes by compression techniques might lead to additional efforts required in the development if the effects are considered too late during the process or in the worst case might lead to failures during operations if not considered at all. To this end, in this paper we investigate the effects of model compression when using global unstructured pruning, post-training quantization, and their combination. We therefore aim to provide insights towards the question what and to which extent changes occur on a deeper level and how that could potentially impact efforts towards arguing the safety of the system by making the following contributions:

- We investigate the effects of model compression on the class and sample level regarding their predictive quality over three different models and two datasets
- Additionally, for model pruning we investigate the effects on the attention of the models regarding compared to the initial models by analyzing their saliency maps

2 Related Work

In the following, we present the related work regarding CNNs compression and its relevance towards arguing the safety for ML-based systems.

2.1 Pruning

Model pruning is a common technique for various ML algorithms, such as decision trees (Mingers 1989) and inductive logic programming (Kazmi, Schüller, and Saygin 2017), not only for compression but also to improve generalization capabilities. For neural networks, pruning is not a novel idea (LeCun, Denker, and Solla 1989), but has gained interest in recent years due to the increased popularity of neural

networks and the need to deploy them on computationally-restricted devices (Cheng et al. 2018). Pruning generally can be performed either in a structured (He et al. 2018) or unstructured (Han, Mao, and Dally 2016) manner. The first one removes – based on a norm for scoring the importance of the individual elements – connected groups of parameters, e.g., on a per-channel or per-filter basis. The structured approach therefore not only provides improvements regarding memory usage, but also provides reduced inference times on regular hardware. Compared to structured pruning, unstructured pruning removes individual parameters, allowing to decrease model sizes significantly more while retaining test accuracy. Since only individual parameters are removed, the overall network structure does not change and sparsity is introduced. Therefore, specialized hardware is required to benefit from inference speedups besides the improvements in memory requirements (Luo and Wu 2020). ML frameworks, such as PyTorch or TensorFlow, come with implementations for the most common pruning techniques. Beyond that, research continues in that domain, for instance, AutoPruner (Luo and Wu 2020) improves significantly upon the state-of-the-art by combining pruning and fine-tuning steps, EagleEye (Li et al. 2020) proposes an efficient evaluation strategy to identify the best performing subnetworks as pruning candidates, and with ShrinkBench (Blalock et al. 2020) a benchmarking framework has been proposed to facilitate the comparison of pruning techniques.

2.2 Quantization

Another widespread model compression technique is quantization that aims to reduce the number of bits required to represent the parameters of a DNN. DNNs are usually trained on hardware accelerators, such as GPUs or TPUs, that use floating points, usually 32bit or 16bit, to represent the parameters. A common technique is to quantize the parameters to 8-bit integers, effectively reducing the size by a factor of 4 or 2 respectively and allowing the exploitation of 8-bit optimized computations of mobile CPUs, while having minimal impact on the model performance (Wu et al. 2016). In practice, post-training quantization and quantization-aware training are common. With post-training quantization, the parameters of a model are quantized after the training phase without requiring any fine-tuning. In contrast, quantization-aware training models the parameter quantization during training and is able to achieve even lower bit-widths. As with pruning, both variants have implementations in common ML frameworks. Research in that domain focuses on achieving lower bit-widths, while only minimally impacting the performance of models (Banner, Nahshan, and Soudry 2019; Hubara et al. 2018) or on further simplifying the quantization process, e.g., by eliminating the need for calibration data (Cai et al. 2020).

2.3 Further Model Compression Techniques

Besides pruning and quantization, other model compression techniques have been proposed. For instance, N2N learning (Ashok et al. 2018) removes parts of a network and afterwards shrinks them using a reinforcement learning approach. With Knowledge distillation, one or more networks are trained to serve as teacher models which a smaller student model is trained to mimic (Hinton, Vinyals, and Dean 2015). Approaches based on low-rank factorization such as (Swaminathan et al. 2020) use matrix decomposition to reconstruct linear transformations of a network into counterparts with less redundancy and therefore fewer parameters. Lastly, although not necessarily a compression technique, neural architecture search can be utilized to find efficient architectures as is done, e.g., in MnasNet (Tan et al. 2019) that optimizes towards the real-world inference latency of DNNs.

2.4 Effects of Model Compression on Robustness

Compressed models have been extensively studied regarding their robustness against adversarial attacks. For instance, (Bernhard, Moellic, and Dutertre 2019) concluded that post-training quantization and quantization-aware training slightly improve the robustness of a network against attacks. Similarly, (Duncan et al. 2020) found that quantization can reduce the transferability of adversarial examples by up to 50%. The adversarially trained model compression framework (Gui et al. 2019) incorporates objectives regarding adversarial robustness in the compression process to further improve upon it. Apart from adversarial examples, some research has been conducted regarding other aspects of robustness. For instance, (Ferianc et al. 2021) demonstrated that a uniform quantization scheme does not considerably impact the quality of uncertainty quantification in Bayesian neural networks. (Hooker et al. 2021) studied the effects of model compression beyond test accuracy and found that a small subset of the data is systematically more impacted and that the sensitivity towards distributional shifts correlates significantly with model sparsity.

2.5 Safety Assurance for ML-based Systems

Arguing the safety of ML-based systems is an emerging field and highly relevant to enable the use of ML in safety-critical applications. A promising direction are holistic assurance strategies (Burton, Gauerhof, and Heinzemann 2017) that incorporate an analysis of the operational domain and the system, as well as a sound validation and verification strategy to design confidence arguments that provide evidence towards the safety of the system (Burton et al. 2019). The approach itself is domain agnostic and so far has been applied to, e.g., the automotive (Burton et al. 2021a) and medical (Picardi et al. 2019) domain. While it provides a general framework towards arguing the safety of ML-based systems and frameworks such as AMLAS (Hawkins et al. 2021) provide additional guidance, further research regarding the design of safe ML algorithms and effective testing methods is required to provide sufficient evidence for the assurance case.

3 Evaluation

In this section, we discuss our results and observed findings regarding the changes beyond test accuracy introduced when compressing image classifiers with pruning or quantization techniques.

3.1 Design of Experiments

To analyze the influence model architecture, we considered three different networks. A ResNet-18 (~11m parameters) (He et al. 2016) for its widespread usage, a SqueezeNet (~750k parameters) (Iandola et al. 2016) for its computational efficiency, and a LeNet-5 (~62k parameters) (Lecun et al. 1998) for its small size. We trained the models on CIFAR-10 (Krizhevsky 2009) and the German Traffic Sign Recognition Benchmark (GTSRB) (Stallkamp et al. 2011). CIFAR-10 consists of 60,000 32x32px images, equally divided into 10 classes, e.g., *cat, dog, automobile, or ship*. GTSRB contains 51,839 images of 43 different German traffic signs that we rescaled to 32x32px. The distribution of the traffic signs thereby is imbalanced, with the most frequent sign, *Speed limit (50 km/h)* occurring more than 10 times as often as the least frequent one, *Dangerous curve to the left*. The class imbalance within GTSRB allows us to study if any negative biases towards the underrepresented classes are introduced by the model compression techniques.

We trained each model by minimizing the negative log-likelihood using Adam as an optimizer. To prevent overfitting, we stopped the training after the loss did not decrease for 40 epochs. To improve the base accuracy on CIFAR-10, we transformed each image at each epoch by randomly flipping it horizontally and by randomly cropping it back to 32x32px after adding a 4px padding each side.

To compress the models, we used the implementations for pruning and quantization provided by the ML framework PyTorch. We choose global unstructured pruning, using the L1 norm to score the parameters of the model, whereby the ones scored lowest are removed. We applied no subsequent fine-tuning as it yielded the best results in our experiments. Compared to structured pruning it is not as applicable to practical applications, as without sparse tensor computations it only affects the memory requirements of the model. However, unstructured pruning is widely considered in academia (LeCun, Denker, and Solla 1989; Renda, Frankle, and Carbin 2019) and with improvements in sparse tensor support on embedded hardware might become the predominant method for practical applications in the future. After the training phase, we pruned each model with the target to maximize the amount of dropped connections while maintaining a comparable level of accuracy to the original model.

For quantization, we chose a non-intrusive post-training approach with per-channel bit allocation, as it gave the best results in our experiments. We chose to quantize the weights of all models once to 8bit and once to 4bit. The first case enables the utilization of integer-based hardware accelerators, while the second one would require specialized hardware to gain additional benefits, apart from increased memory efficiency, compared to the 8-bit variant. The activation precision was kept at 8bit for both cases, as values below that severely impacted the accuracy of the models. Finally, we also combined both compression approaches by quantizing the pruned models with 8-bit precision for weights and activations. Table 1 lists all models and their compressed variants, stating their test accuracy and memory footprint. We do not provide a measure of the inference time as it greatly depends on the execution platform, e.g., if it can exploit the sparseness of the pruned models or if it is optimized towards floating point or integer computations.

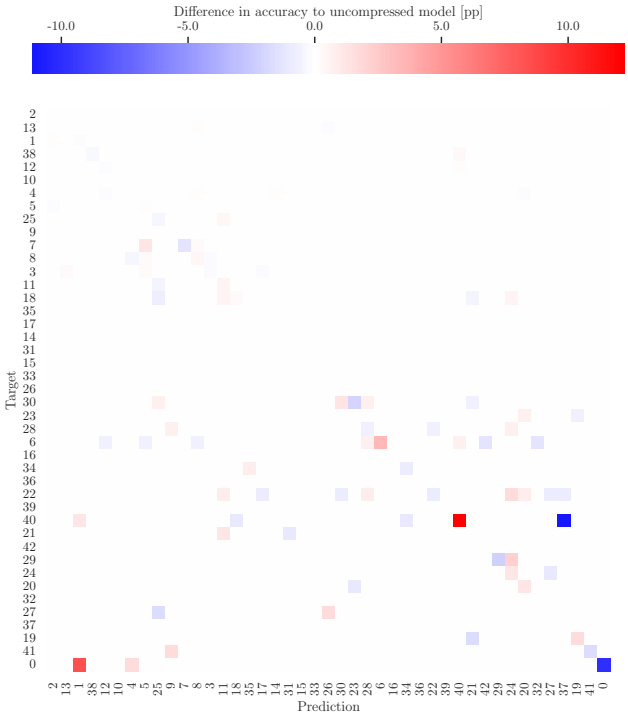


Figure 1: Difference (in percentage points) between the confusion matrices for the uncompressed and pruned ResNet-18 trained on GTSRB. Targets and predictions are ordered by the frequency of the respective class with class 2 being the most frequent and class 0 being the most infrequent one.

For the evaluation, we additionally generated saliency maps by computing the gradients for each input pixel regarding the target class and normalizing them to the range $[0; 1]$ following (Simonyan, Vedaldi, and Zisserman 2014). Since PyTorch does not support gradient calculation for quantized tensors, we only generated saliency maps for the original and pruned variants of the models.

3.2 Results and Discussion

Table 1 shows the test accuracies of all configurations. Most configurations show only a slight drop in accuracy of less than 1 percentage point (pp), with the exception of some networks quantized with 4-bit weight precision. These are not considered further in the following sections as their substantial drop in accuracy already implies significant changes.

3.3 Changes at the Class Level

Pruning The accuracy on the entire test dataset did not reduce significantly after applying pruning for most configurations as Table 1 shows. However, we observe significant changes at the class level for many configurations, especially for GTSRB. For instance, Figure 1 shows the difference between the confusion matrices for the original and pruned ResNet-18 on GTSRB. While the test accuracy

Architecture	Dataset	Percentage Pruned	Uncompressed Accuracy	Pruning Accuracy Difference	4-bit Quant. Accuracy Difference	8-bit Quant. Accuracy Difference	Pruning + 8-bit Quantization Accuracy Difference
LeNet	GTSRB	54.8%	92.3%	-0.4pp	-0.4pp	-0.2pp	-0.7pp
	CIFAR-10	39.8%	74.8%	-0.5pp	**-7.9pp**	-0.1pp	-0.6pp
SqueezeNet	GTSRB	49.4%	93.0%	-0.8pp	**-2.2pp**	-0.2pp	-0.8pp
	CIFAR-10	49.4%	84.5%	-0.4pp	**-4.0pp**	0pp	-0.4pp
ResNet-18	GTSRB	67.4%	95.4%	0pp	-0.2pp	-0.1pp	-0.1pp
	CIFAR-10	72.4%	86.5%	0pp	-0.9pp	-0.1pp	-0.2pp

Table 1: Accuracies on the test dataset for each model and its compressed variants. The column *Percentage Pruned* states how much of the respective network was removed and only applies to the *Pruning* and *Pruning + Quantization (8bit)* variants. Significant deviations in accuracy of more than $1pp$ from the uncompressed model are indicated in bold.

stayed the same, the accuracies and confusions of a few individual classes change significantly. As an example, class 0 (*Speed limit 20km/h*) is confused ~$8pp$ more often with class 1 (*Speed limit 30km/h*) but on the other hand, class 40 (*Roundabout mandatory*) is mistaken ~$11pp$ less often as class 37 (*Go straight or left*). Furthermore, class 40 is predicted more accurately by ~$11pp$ whereas the accuracy of class 0 drops by ~$10pp$. Many more classes have changes in the accuracy or confusion in the range of up to $4pp$ that – depending on the concrete application – might also be relevant. Upon closer inspection, we find that for class 40 the correct predictions of the uncompressed model are a proper subset for the ones of the pruned network. For the remaining samples that were only predicted correctly by the pruned model, we find that the confidence is between $18pp$ and $24pp$ higher for the pruned model. This means, that for these particular samples there is a significant difference in how much support each of the networks generates for them and the increase in the correct predictions for this class by the pruned model is not just based on slight differences. Figure 2 summarizes the difference in confidence for the predicted class between the uncompressed and pruned ResNet-18. While the vast majority of predictions show a similar ($\leq 2.5pp$) confidence, for some samples the confidence changes significantly, up to $27.5pp$. Although this only affects a small subset of all samples and the overall number of samples that are predicted differently is small with 0.6%, it is something to be aware of since it might have been caused by the introduction of additional failure modes. Depending at which stage of the system development model compression is considered this might have several implications. In the worst case, if model compression is performed immediately prior to the deployment without an extensive verification phase afterwards, these failure modes are not addressed, potentially leading to system failures during operations. But even in cases, where it is considered before the model verification it can significantly impact the development. As additional failure modes must be met with proper mitigation measures – e.g., in the form of safety monitors or considerations regarding the operational domain –, the development process can be prolonged if model compression is not considered as integral part of the system development.

For GTSRB, the overall effects regarding pruning are similar but more pronounced for LeNet and SqueezeNet compared to the ResNet-18. Here, for some classes the change in confusion or accuracy is a bit more noticeable with up to $15pp$ and the proportion of samples that are predicted differently is higher with 3.5% and 3% respectively. Also the mean difference in the confidence for each sample is significantly increased, even to the extent where a small fraction of samples for one variant generates full support for the target class and for the other one none, as Figure 3 highlights. The increase in the observed effects is likely due to the smaller initial sizes of LeNet and SqueezeNet compared to the ResNet-18. Although 72% of the ResNet-18's connections were pruned, it still has more than 7 times (SqueezeNet) and 89 times (LeNet) the number of parameters, potentially still containing redundant features.

One important finding on the imbalanced GTSRB is that pruning did not introduce significant biases against the infrequent classes for any of the networks. This also is evident when considering the diagonal in Figure 1, where no correlation between the change in accuracy and the frequency of the class is present. It is to mention, that the significant drop in accuracy for the most infrequent class 0 is only present for ResNet-18, for the other networks it is not present.

On CIFAR-10, the overall effects of pruning are similar to GTSRB but significantly less pronounced as Figure 4 shows. For SqueezeNet and LeNet the change in class-wise accuracies does not exceed $2.5pp$ or $4pp$ respectively. However, for these two networks it is to note that the overall number of samples that are predicted differently by the uncompressed and pruned variant is increased with 4.2% and 7.2% respectively. Referring to Figure 4 this effect can be explained as a result of previously wrong predictions that after applying pruning to the network are still predicted incorrectly but towards another class. Here, it is to highlight that for the classes *airplane* and *horse* this effect is the most prominent, with the first one being overall predicted less and the latter one being predicted more often by the pruned network, therefore introducing a slight bias respectively against or towards these classes. Overall, the increase in the intra-class

Figure 2: Difference (in percentage points) between the confidence in the target class for the uncompressed and pruned ResNet-18 trained on GTSRB. The upper plot shows the difference where the predictions of the both networks differ, the lower one where they are equal. In parenthesis we state the number of samples underlying the respective diagram, e.g., the $n = 77$ in the upper plot shows that for 77 samples the uncompressed and pruned model yielded a different classification result.

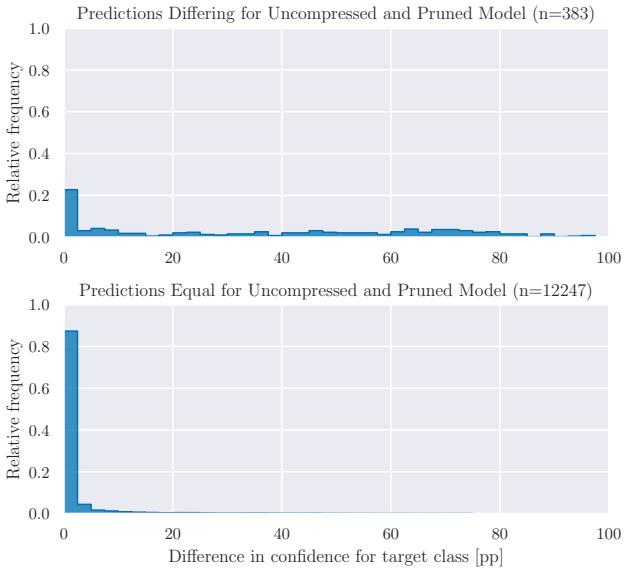


Figure 3: Difference between the confidence in the target class for the uncompressed and pruned SqueezeNet trained on GTSRB. The upper plot shows the difference where the predictions of the both networks differ, the lower one where they are equal.

confusion compared to GTSRB likely can be attributed to the different complexity of the tasks. While GTSRB has only very limited inter-class variance – a *Speed limit (20km/h)* sign always has the same shape and surface, the differences in the images stem from different lighting, viewing angles, etc. – for CIFAR-10 samples from the same class can vary greatly, increasing the likelihood of confusion.

Lastly, considering ResNet-18 on CIFAR-10, virtually no difference is observable between the uncompressed and pruned network. Only two samples are predicted differently and neither the uncompressed nor the pruned variant arrive at the correct prediction. Additionally, the confidence difference regarding the predicted class between both variants in all cases is $\leq 2.5pp$. The likely reason for this similarity is that although 72.4% of the network have been pruned, the pruned network still contains enough redundancy to mimic the initial model and further pruning would be required to elicit any effects. In turn this also highlights that if pruning is performed conservatively and not to the absolute limit, i.e. until even slight changes in the overall accuracy are noticeable, a compressed variant might be achievable that virtually mimics the initial network.

Quantization Referring to Table 1, 8-bit quantization shows very limited impact on the overall accuracy while 4-bit quantization (with 8-bit activation precision) in half of the experiments shows a significant drop in accuracy, a common observation regarding the low precision in combination with the static post-training quantization scheme (Banner, Nahshan, and Soudry 2019). For GTSRB, we again observe changes on the class level introduced by the quantization – but to a slightly lesser extent compared to pruning, although in most cases also reducing the test accuracy lesser – as Figure 5 depicts. The same observation can be made for SqueezeNet and LeNet, where the maximum extent of the change in confusion or accuracy is up to $6pp$ or $12pp$ respectively. Comparing the differences in the confusion matrices for pruned and quantized networks, it is also evident that classes are not necessarily affected in the same way by both compression methods. This hints towards the finding that not only samples that are challenging for the uncompressed model are affected but that the compression techniques can potentially affect any sample. Regarding CIFAR-10, we again report the overall similar but significantly reduced effects compared to GTSRB, as we already observed for pruning, with the exception that the ResNet-18 also is slightly affected with changes in confusion and accuracy, up to $0.6pp$.

Comparing 4-bit (with 8-bit activation precision) quantization to 8-bit quantization for the configurations without a significant drop in accuracy, we find that the observed effects are the same but more pronounced for 4-bit quantization. As already observed when comparing pruning and 8-bit quantization, 4-bit quantization and 8-bit quantization show no consistent patterns regarding the impacted samples, further supporting the hypothesis that any sample or class can be affected.

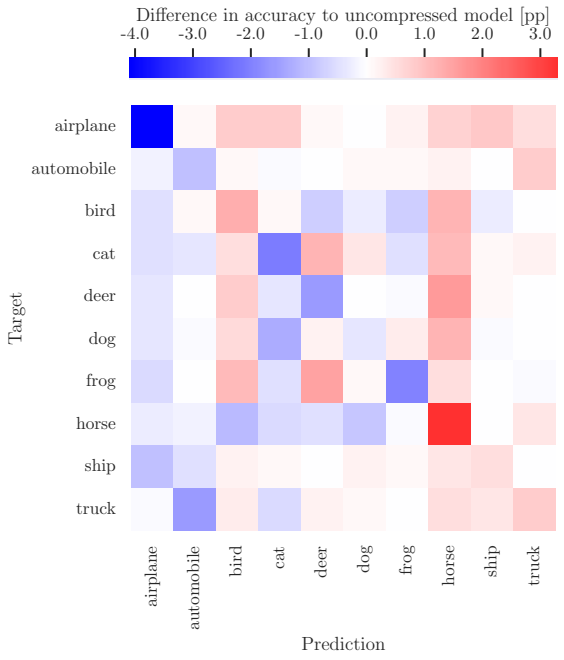


Figure 4: Difference between the confusion matrices for the uncompressed and pruned LeNet trained on CIFAR-10.

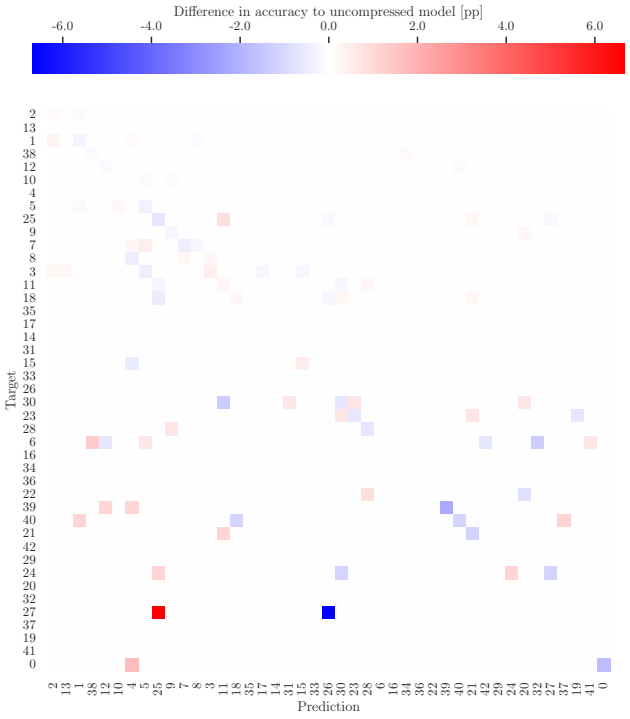


Figure 5: Difference between the confusion matrices for the uncompressed and 8-bit quantized ResNet-18 trained on GTSRB.

Combined Pruning and Quantization Lastly, we combine pruning and quantization, by applying 8-bit quantization to the pruned models. As Table 1 shows, this has a slightly higher impact on the overall drop in accuracy than if the compression techniques would be applied individually, which is to be expected. Overall, the same general effects are observable as if pruning or quantization are applied individually as Figure 6 shows. Regarding the effect on individual classes, patterns present in both compression techniques are combined, sometimes amplifying, other times canceling out the effects. Potentially this could lead to drastic effects, however, in our experiments we did not observe any. Also, it is noticeable that generally the number of samples where the uncompressed and the compressed model disagree is slightly higher than for any of the single compression variants. Generally speaking, the combination of both compression techniques, however, shows no peculiarities and does not introduce significant additional effects.

3.4 Differences in the Relevance of Input Regions

In addition to the quantitative analysis performed in the previous section, we also qualitatively investigated the changes introduced by model compressing. For this, we generated saliency maps that highlight the salient input regions for a model's decision regarding the target class. Since statically quantized models in PyTorch don't support gradient calculation, we only analyze the changes between uncompressed and pruned model variants. In order to keep the number of images to analyze manageable, for each configuration we selected the 20 samples where the biggest difference in the saliency maps was present. To compare two saliency maps, we first performed a 3x3 average pooling with stride 3 over the saliency maps – to reduce the sensitivity towards pixel-level changes in the attention, putting a stronger emphasis on higher-level features – and afterwards computed the Mean Absolute Deviation (MAD) between the reduced saliency maps of the uncompressed and pruned model.

Figure 7 shows five selected saliency maps that summarize the observed findings. Overall, we did not observe any systematic changes between any model and its pruned variant. For example, in some instances, the pruned model focuses better on the foreground, giving the correct prediction, while the original model focuses on the background, classifying incorrectly, as Figure 7a representatively shows. However, this is not a consistent behavior, and the opposite effect can be observed as well, e.g., in Figure 7b, where the pruned model puts too much attention on the sky, classifying the image as an airplane. Furthermore, even for samples where both networks predict the same class, we can observe significant changes in the salience of different input regions, i.e, the models weight features differently or even rely on different ones. While in the previous sections we found virtually no differences between the uncompressed and pruned ResNet-18 on CIFAR-10, regarding their attention we could find noticeable differences as Figures 7d and 7e show. This effect is also not limited to our selected samples, as Figure 8 shows. With an average MAD of 7.7% between the saliency maps of the uncompressed and pruned ResNet-18, it highlights that although the effects of model compression might

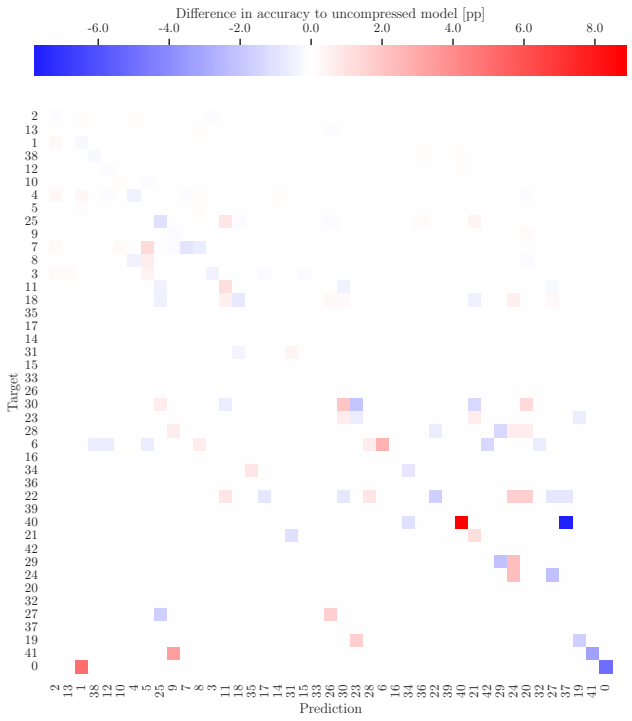
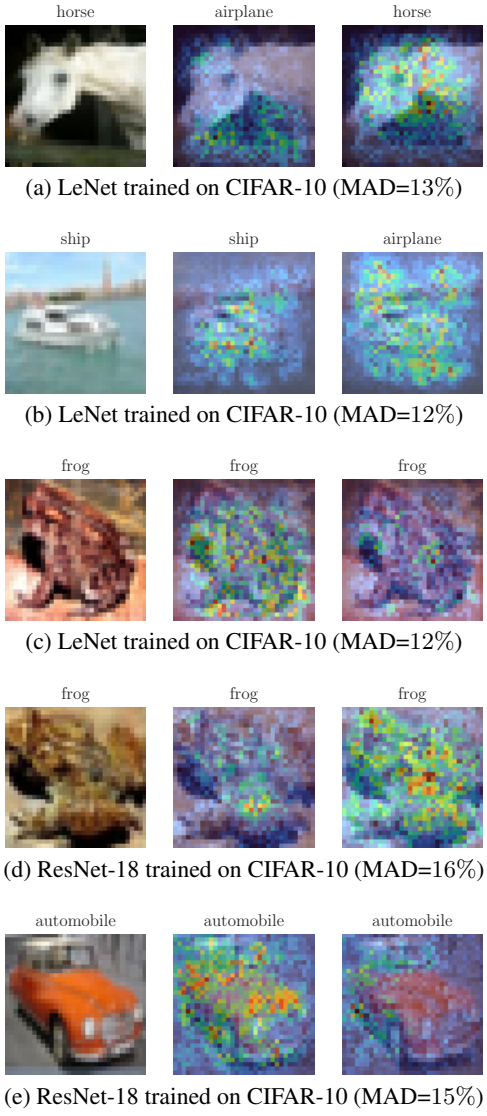


Figure 6: Difference between the confusion matrices for the uncompressed and the pruned + 8-bit quantized ResNet-18 trained on GTSRB.

not be noticeable at the level of dataset or even class-wise accuracy, it is definitely important to consider them in safety analyses, as it might, for example, introduce additional failures in corner cases where a model bases its decision on the wrong features.

4 Conclusions and Future Work

In this paper, we investigated changes in the predictions of networks compressed with either post-training quantization, global unstructured pruning, or a combination of both. While the deviations from the test accuracy of the uncompressed model were minimal, we observed that the compression techniques still caused significant changes in the predictions. For one thing, we found that the accuracy of individual classes can change greatly – in our experiments up to $15pp$ – and that the confusion between classes can vary to the same extent. For another thing, our investigation showed that also the confidence regarding the target class can change significantly, with extreme cases were the uncompressed model has zero confidence in the target class while the compressed variant has full confidence, and vice versa. Lastly, our comparison of saliency maps for uncompressed and pruned models revealed the presence of significant differences, hinting towards the two variants relying on or weighting features differently. It is to mention, however, that we did not observe the introduction of systematic errors, e.g., in the form of biases against infrequent classes. Nonetheless, based on the effects we observed, we strongly suggest to view model



(a) LeNet trained on CIFAR-10 (MAD=13%)

(b) LeNet trained on CIFAR-10 (MAD=12%)

(c) LeNet trained on CIFAR-10 (MAD=12%)

(d) ResNet-18 trained on CIFAR-10 (MAD=16%)

(e) ResNet-18 trained on CIFAR-10 (MAD=15%)

Figure 7: Comparison of saliency maps for the original and pruned variants of the LeNet and ResNet-18 trained on CIFAR-10. On the left is the original image with the target class annotated. Following that are the saliency maps for the original (middle) and pruned (right) model each annotated with the prediction of the respective model.

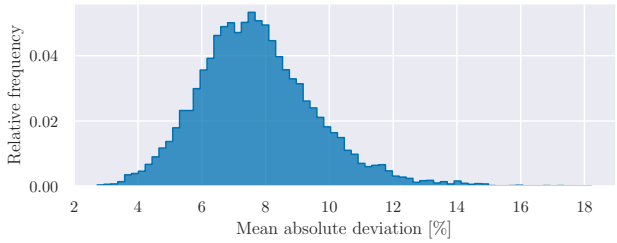


Figure 8: Distribution of the mean absolute deviations between the saliency maps of the uncompressed and pruned ResNet-18 trained on CIFAR-10.

compression as integral part of any ML development cycle and to consider it in early development stages. Model compression can cause substantial changes in the predictions of a network and with that bears the potential to introduce additional failure modes. These must be addressed in the system development and the earlier they are known, the better mitigation measures can be integrated in the system, overall facilitating the development process.

Regarding future work, we suggest to expand our experiments also to other model architectures, datasets, and tasks and to investigate other compression techniques, e.g., quantization-aware training, structured pruning, or knowledge distillation, as these are also highly relevant in practice. Furthermore, we deem it as highly important to further develop methods for systematically and rigorously analyzing machine learning systems that go beyond averaging metrics, as these hide many peculiarities that bear the potential for failures. Lastly, we deem it equally as important to continue research into the direction of continuous safety assurance (Burton et al. 2021b) in order to consider safety as integral part of the development of ML-based systems, addressing issues such as potentially negative effects due to model compression early on.

References

Ashok, A.; Rhinehart, N.; Beainy, F.; and Kitani, K. M. 2018. N2N Learning: Network to Network Compression via Policy Gradient Reinforcement Learning. In *Proc. ICLR*.

Banner, R.; Nahshan, Y.; and Soudry, D. 2019. Post Training 4-Bit Quantization of Convolutional Networks for Rapid-Deployment. In *Proc. NeurIPS*, 714, 7950–7958. Red Hook, NY, USA: Curran Associates Inc.

Bernhard, R.; Moellic, P.-A.; and Dutertre, J.-M. 2019. Impact of Low-Bitwidth Quantization on the Adversarial Robustness for Embedded Neural Networks. In *2019 International Conference on Cyberworlds (CW)*, 308–315.

Blalock, D. W.; Ortiz, J. J. G.; Frankle, J.; and Guttag, J. V. 2020. What Is the State of Neural Network Pruning? In *Proc. MLSys*. mlsys.org.

Burton, S.; Gauerhof, L.; and Heinzemann, C. 2017. Making the Case for Safety of Machine Learning in Highly Automated Driving. In *Computer Safety, Reliability, and Security*, LNCS, 5–16. Cham: Springer International Publishing.

Burton, S.; Gauerhof, L.; Sethy, B. B.; Habli, I.; and Hawkins, R. 2019. Confidence Arguments for Evidence of Performance in Machine Learning for Highly Automated Driving Functions. In *Computer Safety, Reliability, and Security*, LNCS, 365–377. Cham: Springer International Publishing.

Burton, S.; Kurzidem, I.; Schwaiger, A.; Schleiß, P.; Unterreiner, M.; Graeber, T.; and Becker, P. 2021a. Safety Assurance of Machine Learning for Chassis Control Functions. In *Computer Safety, Reliability, and Security*, LNCS. Cham: Springer International Publishing.

Burton, S.; McDermid, J. A.; Garnett, P.; and Weaver, R. 2021b. Safety, Complexity, and Automated Driving: Holistic Perspectives on Safety Assurance. *Computer*, 54(8): 22–32.

Cai, Y.; Yao, Z.; Dong, Z.; Gholami, A.; Mahoney, M. W.; and Keutzer, K. 2020. ZeroQ: A Novel Zero Shot Quantization Framework. In *Proc. CVPR*, 13169–13178.

Cheng, Y.; Wang, D.; Zhou, P.; and Zhang, T. 2018. Model Compression and Acceleration for Deep Neural Networks: The Principles, Progress, and Challenges. *IEEE Signal Process. Mag.*, 35(1): 126–136.

Duncan, K.; Komendantskaya, E.; Stewart, R.; and Lones, M. 2020. Relative Robustness of Quantized Neural Networks Against Adversarial Attacks. In *Proc. IJCNN*, 1–8.

Ferianc, M.; Maji, P.; Mattina, M.; and Rodrigues, M. 2021. On the Effects of Quantisation on Model Uncertainty in Bayesian Neural Networks. *arXiv:2102.11062 [cs, stat]*.

Gui, S.; Wang, H. N.; Yang, H.; Yu, C.; Wang, Z.; and Liu, J. 2019. Model Compression with Adversarial Robustness: A Unified Optimization Framework. In *Proc. NeurIPS*, volume 32. Curran Associates, Inc.

Han, S.; Mao, H.; and Dally, W. J. 2016. Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding. In *Proc. ICLR*.

Hawkins, R.; Paterson, C.; Picardi, C.; Jia, Y.; Calinescu, R.; and Habli, I. 2021. Guidance on the Assurance of Machine Learning in Autonomous Systems (AMLAS). *CoRR*, abs/2102.01564.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *Proc. CVPR*, 770–778.

He, Y.; Kang, G.; Dong, X.; Fu, Y.; and Yang, Y. 2018. *Soft Filter Pruning for Accelerating Deep Convolutional Neural Networks*.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the Knowledge in a Neural Network. *arXiv:1503.02531 [cs, stat]*.

Hooker, S.; Courville, A.; Clark, G.; Dauphin, Y.; and Frome, A. 2021. What Do Compressed Deep Neural Networks Forget? *arXiv:1911.05248 [cs, stat]*.

Hubara, I.; Courbariaux, M.; Soudry, D.; El-Yaniv, R.; and Bengio, Y. 2018. Quantized Neural Networks: Training Neural Networks with Low Precision Weights and Activations. *Journal of Machine Learning Research*, 18(187): 1–30.

Iandola, F. N.; Moskewicz, M. W.; Ashraf, K.; Han, S.; Dally, W. J.; and Keutzer, K. 2016. SqueezeNet: AlexNet-Level Accuracy with 50x Fewer Parameters and <1MB Model Size. *CoRR*, abs/1602.07360.

Kazmi, M.; Schüller, P.; and Saygin, Y. 2017. Improving Scalability of Inductive Logic Programming via Pruning and Best-Effort Optimisation. *Expert Systems With Applications*, 87: 291–303.

Krizhevsky, A. 2009. Learning Multiple Layers of Features from Tiny Images. 60.

Lecun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-Based Learning Applied to Document Recognition. *Proc. IEEE*, 86(11): 2278–2324.

LeCun, Y.; Denker, J. S.; and Solla, S. A. 1989. Optimal Brain Damage. In Touretzky, D. S., ed., *Proc. NIPS*, 598–605. Morgan Kaufmann.

Li, B.; Wu, B.; Su, J.; and Wang, G. 2020. EagleEye: Fast Sub-Net Evaluation for Efficient Neural Network Pruning. In *Proc. ECCV*, LNCS, 639–654. Cham: Springer International Publishing.

Luo, J.-H.; and Wu, J. 2020. AutoPruner: An End-to-End Trainable Filter Pruning Method for Efficient Deep Model Inference. *Pattern Recognition*, 107: 107461.

Mingers, J. 1989. An Empirical Comparison of Pruning Methods for Decision Tree Induction. *Machine Learning*, 4(2): 227–243.

Picardi, C.; Hawkins, R.; Paterson, C.; and Habli, I. 2019. A Pattern for Arguing the Assurance of Machine Learning in Medical Diagnosis Systems. In *Computer Safety, Reliability, and Security*, LNCS, 165–179. Cham: Springer International Publishing.

Renda, A.; Frankle, J.; and Carbin, M. 2019. Comparing Rewinding and Fine-Tuning in Neural Network Pruning. In *ICLR 2019*.

Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2014. Deep inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. In *Proc. ICLR*.

Stallkamp, J.; Schlipsing, M.; Salmen, J.; and Igel, C. 2011. The German Traffic Sign Recognition Benchmark: A Multi-Class Classification Competition. In *Proc. IJCNN*, 1453–1460.

Swaminathan, S.; Garg, D.; Kannan, R.; and Andres, F. 2020. Sparse Low Rank Factorization for Deep Neural Network Compression. *Neurocomputing*, 398: 185–196.

Tan, M.; Chen, B.; Pang, R.; Vasudevan, V.; Sandler, M.; Howard, A.; and Le, Q. V. 2019. MnasNet: Platform-Aware Neural Architecture Search for Mobile. In *Proc. CVPR*, 2820–2828.

Wu, J.; Leng, C.; Wang, Y.; Hu, Q.; and Cheng, J. 2016. Quantized Convolutional Neural Networks for Mobile Devices. In *Proc. CVPR*, 4820–4828.

5 Acknowledgments

This work was funded by the Bavarian Ministry for Economic Affairs, Regional Development and Energy as part of a project to support the thematic development of the Institute for Cognitive Systems.