Model Compression for Resource-Constrained Mobile Robots

Timotheos Souroulla
Ericsson Research AI
Ericsson AB, Sweden
Stockholm, Sweden
timotheos.souroulla@ericsson.com

Alberto Hata
Ericsson Research AI
Ericsson Telecomunicações S/A
Indaiatuba, Brazil
alberto.hata@ericsson.com

Ahmad Terra
Ericsson Research AI
Ericsson AB
Stockholm, Sweden
ahmad.terra@ericsson.com

Özer Özkahraman
Robotics Perception and Learning
KTH, Royal Institute of Technology
Stockholm, Sweden
ozero@kth.se

Rafia Inam
Ericsson Research AI
Ericsson AB
Stockholm, Sweden
rafia.inam@ericsson.com

The number of mobile robots with constrained computing resources that need to execute complex machine learning models has been increasing during the past decade. Commonly, these robots rely on edge infrastructure accessible over wireless communication to execute heavy computational complex tasks. However, the edge might become unavailable and, consequently, oblige the execution of the tasks on the robot. This work focuses on making it possible to execute the tasks on the robots by reducing the complexity and the total number of parameters of pre-trained computer vision models. This is achieved by using model compression techniques such as Pruning and Knowledge Distillation. These compression techniques have strong theoretical and practical foundations, but their combined usage has not been widely explored in the literature. Therefore, this work especially focuses on investigating the effects of combining these two compression techniques. The results of this work reveal that up to 90% of the total number of parameters of a computer vision model can be removed without any considerable reduction in the model's accuracy.

1 Introduction

The amount of scenarios where robots (which are mostly resource-constrained) that need to execute complex machine learning models is increasing rapidly. An example of such a scenario is a Human-Robot Collaboration (HRC) scenario where robots need to avoid any hazardous situations through safety analysis [8, 9]. Commonly, safety analysis involves complex computer vision tasks that require substantial computing power to perform the inferences in real-time.

Many robots have constrained computing resources that might not fulfill the requirement of executing the safety analysis module without delays, which may lead to an increase in the risk level. Therefore, they rely on edge infrastructures to run these complex tasks. In some cases, the communication between the robot and the edge is not possible and in other cases, a less accurate inference result is allowed in low-risk situations (e.g. absence of nearby obstacles). These situations motivate devising solutions to process the robot's sensor data on its embedded processor rather than executing on the edge.

Through the years, different approaches aiming to tackle computer vision and object detection tasks were investigated, leading to a lot of different computer vision models. Most of these algorithms have the major disadvantage of being highly complex, thus they require a lot of processing and computing power to be executed. Consequently, mobile robots with constrained computing resources cannot execute

R.C. Cardoso, A. Ferrando, F. Papacchini,
M. Askarpour, L.A. Dennis (Eds.): Second
Workshop on Agents and Robots for reliable
Engineered Autonomy (AREA'22).
EPTCS 362, 2022, pp. 54–64, doi:10.4204/EPTCS.362.7

them on their processors without any delays. Therefore, an investigation seeking to reduce the original version's complexity and the required computational load will be beneficial.

This work proposes reducing the complexity of the computer vision models through model compression to allow their execution on the robot's embedded computer. Essentially, model compression reduces the complexity of a model by removing network connections that have low contribution in the output generation. In addition, computer vision models can have a plethora of redundant operations that do not need to be processed, such as multiplications by 1 or additions with 0. Two techniques commonly employed in model compression problems are evaluated: Pruning [12], which identifies and removes redundant operations, and Knowledge Distillation [14], which replaces the original model with a smaller, less complex one that mimics the output of the original one. Given these techniques, it is verified whether the compressed models can maintain high levels of accuracy while reducing their total number of parameters. The solution is aimed to be employed in safety-critical HRC scenarios where real-time requirements should be satisfied along with accurate responses of the models.

The main contribution of this work is a model compression solution that combines different strategies to reduce the complexity of computer vision models with low degradation in accuracy. More specifically, Knowledge Distillation and Pruning are combined which showed better performance compared to their stand-alone versions.

The rest of this paper is structured as follows: Section 2 presents related work of model compression techniques; Section 3 presents the methods used for compressing a deep neural network (DNN) model; Section 4 presents the experimental setup; Section 5 provides an analysis and comparisons between the results; Sections 6 and 7 conclude this work with discussions and future works.

2 Related Works

The reduction of the complexity and the computational load of complex computer vision tasks is done for multiple reasons. One of them is the constrained computing resources of mobile robots which prohibits real-time execution of these tasks. As mentioned in [12], the most important reason for compressing a model is the low power and memory budget of a mobile robot as a significant amount of memory is required for storing the network's weights and millions of multiplications must be carried out for a single input. Another reason that makes model compression necessary is the low-latency requirement in time-critical scenarios since mobile robots should act immediately in a hazardous situation.

Several different methods for compressing a model are discussed in the literature with Pruning and Knowledge Distillation being the most prevalent ones. Pruning [3] is a model compression technique that compresses the original model by removing and eliminating redundant operations, and Knowledge Distillation [7] is a model compression technique in which the large complex model is replaced by a smaller and less complex one. Moreover, DNN (Deep Neural Network) Splitting is a model compression method in which the computer vision model is divided into two parts, the "head" and the "tail". The first part is executed on the local processor of a robot, and the second part, which is more complex than the first one is executed on the edge [14]. Additionally, Neural Filter [15] is a filter that is embedded in the early layers of the overall object detection model and is a classifier whose output is binary, indicating whether an image is empty or not. Empty images are discarded and are not processed by the model and as a result, fewer images are processed leading to a reduced amount of total calculations completed.

When it comes to Pruning, there are numerous techniques that one could use that follow different criteria. Minimum weight, activation, and mutual information [16] are three main criteria that are used in Pruning. In the minimum weight criterion, connections that have the lowest weights in a layer are

removed. In the activation criterion, weights are removed based on the assumption that if an activation value (an output feature map) is low, then this feature detector is not significant for the prediction task, and thus, it is removed. In the mutual information criterion though, the decision of whether to keep a connection or not is based on measuring how much information from one connection is already present in another connection of the neural network. The connection is removed if this information is incorporated in other connections.

Knowledge Distillation introduced in [7], is a method in which a simpler model, called the "student" model, is trained to mimic the output of a more complex one, called the "teacher" model. The key assumption is that large "teacher" models are often over-parameterized and can be compressed without any significant performance loss. At the end of the training phase, the "student" model replaces the "teacher" model and as a result, the complexity and the inference time of the computer vision process are reduced.

These two model compression techniques have been investigated in depth in the literature and have strong theoretical and practical foundations, but work on their combinations is limited. The novel part of this work is to investigate their combination and verify if it outperforms the stand-alone versions.

3 Background on the Applied Methods

In this section, the model compression techniques that are applied in the proposed solution are described, beginning with Pruning and followed by Knowledge Distillation.

3.1 Pruning

Deep neural networks are often over-parameterized and may have redundant operations that do not have an effect on the end result. Pruning is a model compression technique that aims to reduce the number of parameters of complex models by identifying and eliminating redundant operations. The first step of this technique is to train the neural network, while the second step is to identify and remove unimportant connections. In this step, connections that have low weights are removed, using one of the many different Pruning techniques that are available. The final step of Pruning is the fine-tuning of the obtained model to avoid losing a significant amount of accuracy.

Among existing Pruning techniques, this work employs random, class-uniform, and class-blind [13] Pruning. In random Pruning, a percentage x is selected by the user, and regardless of their layer and weight value, $x\%$ connections are randomly removed from the whole network. In class-uniform, Pruning is distributed equally among hidden layers, which is achieved by selecting a percentage x and Pruning out the $x\%$ connections with the lowest weights per layer. Whereas in class-blind, given a percentage x, connections with the lowest $x\%$ are removed from the network regardless of the layer they are in. Studies showed a better performance of class-blind compared to the previous two techniques [13]. Each Pruning technique results in a different network as can be seen in Figure 1. Connections that are going to be pruned are marked with red labels.

3.2 Knowledge Distillation

The goal of Knowledge Distillation is to use a less complex model, so-called the student model, to replicate the output represented by a more complex model, so-called the teacher model. This method is based on the assumption that teacher models are often over-parameterized, and can be compressed without causing significant performance loss. By using a simpler model that produces the same output,

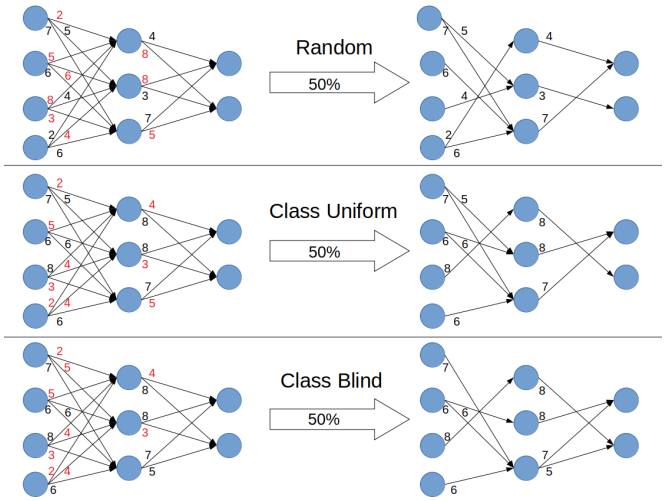


Figure 1: Different Pruning techniques result in different end models, even when using the same initial neural network. Top figure shows random Pruning in which 50% connections are removed randomly. Middle figure shows class uniform where 50% lowest weights are discarded. Bottom figure shows class blind which 50% lowest weights of each layer are eliminated.

the total inference time is reduced. An example of Knowledge Distillation compression can be seen in Figure 2. Mimicking the output of the original model is the key to a successful Knowledge Distillation, and when the student model replicates the output of a teacher model, the knowledge transferred from it can achieve higher performance than the original one [2]. This replication is achieved by training the student model using the output of the teacher model instead of using the labels provided by a dataset. For performing the Knowledge Distillation technique, both models, the teacher and the student must be selected by the user. The selection of these models can be challenging as it can be difficult to find an appropriate student model. Therefore, a trial-and-error process may be necessary to ensure satisfactory results.

4 Experimental Setup

In total, three sets of experiments with the same goal are implemented. This goal is to reduce the total size of the selected models without losing any significant amount of accuracy. The first set of experiment that is implemented is Pruning, with the second one being Knowledge Distillation. The third and last set of experiment is a combination of the first two experiments, which consists of applying Knowledge Distillation on a selected model and then using Pruning on the resulting model. The experiments are reproduced for two different tasks, Object Detection and Semantic Segmentation.

4.1 Datasets

For the Object Detection task, where the output of the computer vision models are bounding boxes of the detected objects, the dataset that is used is CIFAR10 [10].

For the Semantic Segmentation part of the experiments, where the output of the computer vision models are segmentation masks of the detected objects, the datasets that are used are COCO [11] and

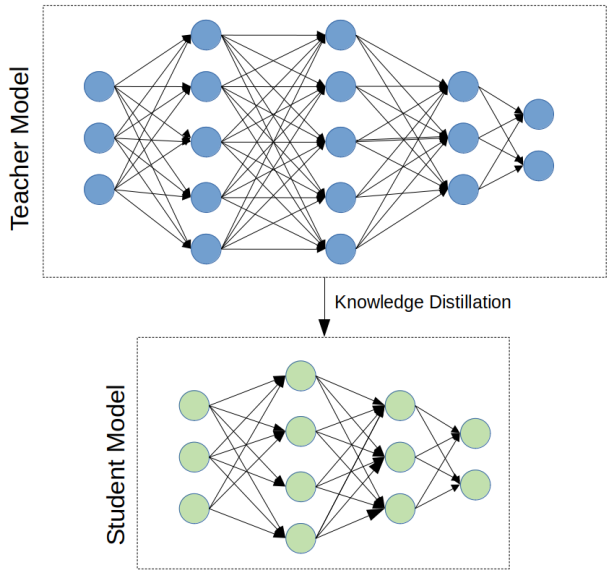


Figure 2: Knowledge Distillation example that begins from a large complex teacher model and results in a smaller, less complex student model.

Pascal-VOC [4].

These datasets were chosen as they are easily accessible through common libraries, such as torchvision [17].

4.2 Models

For the purposes of this work, five models are compressed using two model compression techniques. Three of the models are used for the object detection task, which are LeNet [18] and two models from the ResNet family [6], which are ResNet-18 and ResNet-101. The remaining two models are used for the semantic segmentation task and these models are DeepLabV3 MobileNetV3 [1] and one from the ResNet family, FCN ResNet-101. These models were chosen due to their number of parameters and considerable differences between them. For this work, any model with a similar number of parameters might be used, but these five were chosen due to their easy accessibility and widespread use.

The total number of parameters for each of these models is listed in Table 1.

Table 1: Number of parameters of the models used.

Object Detection		
LeNet	**ResNet-18**	**ResNet-101**
0.357×10^6	11.2×10^6	44.5×10^6

Semantic Segmentation	
FCN ResNet-101	**DeepLab3**
54.3×10^6	11.1×10^6

4.3 Performance Metric

The metric that is considered for determining which model performs best is a trade-off between the drop in the accuracy of the compressed model and the percentage that the network was compressed. The goal is to maximize the compression percentage while minimizing the accuracy drop of the compressed model (for the Object Detection, top-1 accuracy is considered, and for the Semantic Segmentation task, Global Pixel Accuracy is considered [5]).

5 Results and Analysis

In this section, the results of the three experiments are presented and analyzed: Pruning, Knowledge Distillation and the combination of these two techniques.

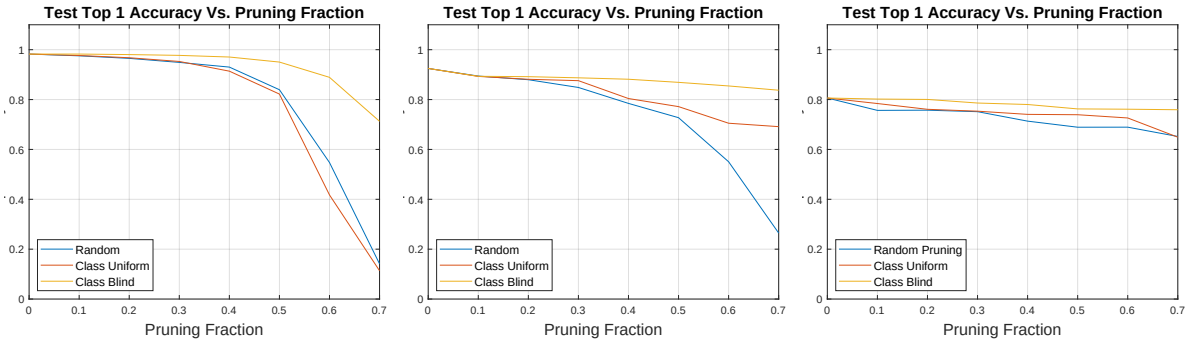
5.1 Pruning

In this experiment, three different Pruning techniques are applied in the different models presented in Table 1, aiming to identify the best performing Pruning technique for both Object Detection and Semantic Segmentation tasks. The three Pruning techniques that are used are random, class-uniform and class-blind Pruning.

The Pruning results for the object detection models can be seen in Figure 3 and Figure 4, which present the top-1 validation accuracy and the number of parameter curves, respectively. Figure 3a and Figure 4a present the results for the LeNet model, while Figure 3b and Figure 4b present the results for ResNet-18 model, and Figure 3c and Figure 4c present the results for the ResNet-101 model. These results show that in all three models and for most of the different Pruning percentages, the technique that performed best was the class-blind technique, as it achieved higher accuracy than the other two techniques. Furthermore, the class-blind technique was more consistent than the other two techniques, as the accuracy of the models did not have a large drop, even for Pruning percentages above 40%. Thus, it can be concluded that the class-blind technique is the best performing Pruning technique for the object detection task and will be used for the upcoming experiments.

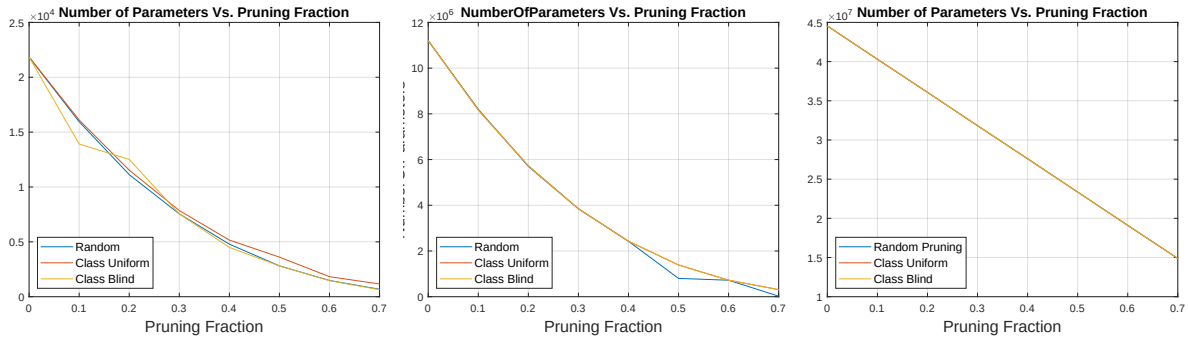
As for the number of parameters, a linear behavior among the different Pruning percentages was expected, as well as the same number of parameters for all three Pruning techniques since the same Pruning percentage was applied. However, as it can be observed from Figure 4, this is not the case. This non-linear behavior takes place as in some cases, all the input connections of a neuron are pruned, and as a result, all the output connections are pruned as well since there is no input. This is the main reason for the difference in the number of parameters between different Pruning techniques and for the non-linear behavior for different Pruning percentages.

The results for the semantic segmentation models can be seen in Figure 5, which presents the Global Pixel Accuracy of the MobileNet model for the different Pruning strategies and percentages. Figure 5a and Figure 5b present the results for the two different datasets used in this experiment. The semantic segmentation results show that there are no significant differences in the accuracy drop between the three Pruning techniques, even for Pruning percentages above 40%. Thus, it can be concluded that any of these techniques could be used for the upcoming experiments, but the decision was to use the class-blind technique in order to stay consistent between object detection and semantic segmentation tasks.



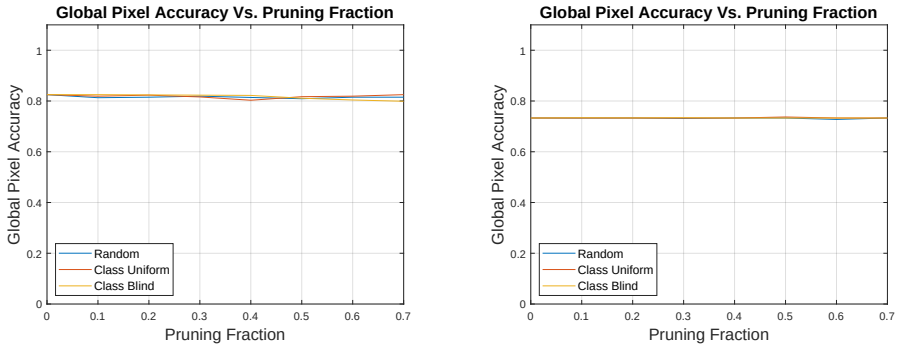
(a) Results for LeNet. (b) Results for ResNet-18. (c) Results for ResNet-101.

Figure 3: Top-1 Accuracy for Object Detection models for different Pruning strategies and percentages.



(a) Results for LeNet. (b) Results for ResNet-18. (c) Results for ResNet-101.

Figure 4: Number of Parameters for Object Detection models for different Pruning strategies and percentages.



(a) Results for COCO Dataset. (b) Results for VOC Dataset.

Figure 5: Global Pixel Accuracy for Semantic Segmentation for different Pruning strategies and percentages.

5.2 Knowledge Distillation

In the Knowledge Distillation experiment, different combinations of models were tested to evaluate the accuracy drop between the teacher and the student models. In total, four different combinations were evaluated, three for the object detection task and one for the semantic segmentation task. At the end of this experiment, the best combinations out of them are selected and used for the upcoming experiment.

For the object detection task, the first evaluated combination had ResNet-18 as the teacher model and LeNet as the student model, while the second one used ResNet-101 as the teacher model and ResNet-18 as the student model. The third combination for this task used ResNet-101 as the teacher model and LeNet as the student model. For the semantic segmentation task, one combination was tested out and that was using FCN ResNet-101 as the teacher model and MobileNet-V3 as the student model, both for COCO and VOC datasets.

In each combination, the accuracy and the number of parameters of the student model were obtained. These values along with the accuracy of the teacher model are presented in Table 2 and Table 3 for object detection and semantic segmentation, respectively. These results show an accuracy higher than 80% for all combinations while significantly reducing the number of parameters. We see that in three combinations (ResNet-101 to ResNet-18, ResNet-101 to LeNet and FCN ResNet-101 to MobileNet-V3), the student model achieved higher accuracy than the teacher model. This behavior is in line with the literature [2] and can happen when the student model has a more suitable DNN architecture for a given task than the teacher model.

	ResNet-18 to LeNet	**ResNet-101 to ResNet-18**	**ResNet-101 to LeNet**
Accuracy of the teacher model(%)	92.48	80.62	80.62
Accuracy of the student model(%)	85.10	86.81	81.47
Number of Parameters	11.2×10^6 to 0.357×10^6	44.5×10^6 to 11.2×10^6	44.5×10^6 to 0.357×10^6

Table 2: Results of the validation accuracy for different combinations of teacher and student models for the object detection task.

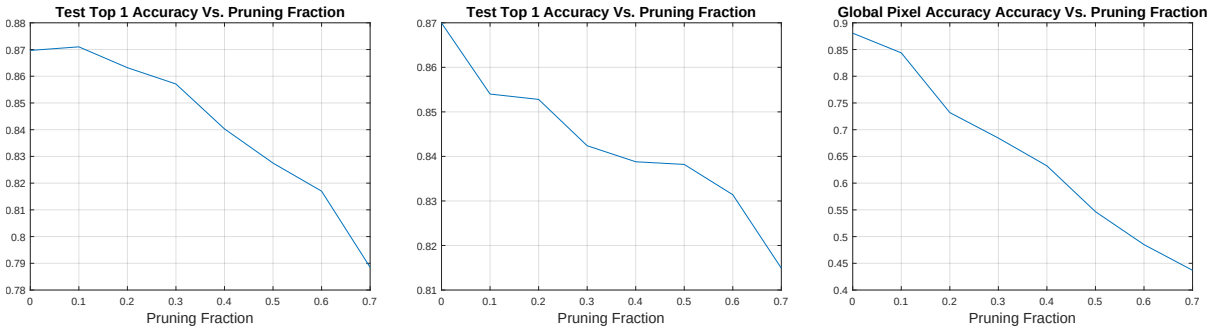
	VOC Dataset	**COCO Dataset**
Accuracy of the teacher model(%)	82.4	82.4
Accuracy of the student model(%)	88.06	90.68
Number of Parameters	54.3×10^6 to 11.1×10^6	54.3×10^6 to 11.1×10^6

Table 3: Results of the validation accuracy for combination of teacher model being FCN ResNet-101 and the student model being MobileNet-V3 for the semantic segmentation task.

Finally, from the Knowledge Distillation experiment, all four combinations achieved accuracy higher than 80%, and the total number of parameters is reduced more than 75%. Nevertheless, the combinations that are going to be used for the upcoming experiment are ResNet-18 to LeNet and ResNet-101 to ResNet-18 for the object detection task, and FCN ResNet-101 to MobileNet-V3 for the semantic segmentation task. These combinations were selected as they achieve total accuracy higher than 85% and they have a sufficient number of parameters left for investigating the effect of Pruning when applied to the student model.

5.3 Combining Knowledge Distillation and Pruning

In this experiment, which is the main contribution of this work, Knowledge Distillation is first applied on a large teacher model, and then Pruning is applied on the resulting student model. Based on the results of the two previous experiments, three student models from Knowledge Distillation were selected and then applied class-blind Pruning. The resulting plots for the pruned student models with the accuracy values against the Pruning fraction are shown in Figure 6.



(a) Results for ResNet-18 model to LeNet.

(b) Results for ResNet-101 to ResNet-18.

(c) Results for FCN ResNet-101 to MobileNet-V3.

Figure 6: Results for the Pruning part for the three resulted student models.

From the results of the object detection task, which can be seen in the first two rows of Table 2 and from figures Figure 6a and Figure 6b, it can be observed that the results of both combinations achieved total accuracy higher than 80% for a compression percentage of more than 90%. An even higher compression percentage can be achieved by applying a bigger Pruning fraction, but it comes with the cost of total accuracy being less than 80%. The best-performing combination is considered to be the second one, which is using ResNet-101 as the teacher model and ResNet-18 as the student model. The reason is that it has a lower accuracy drop than the other combination, with approximately the same ratio in the number of parameters between the student and the teacher model.

The results of the semantic segmentation task can be seen in Table 3 and from figure Figure 6c. From these results, it can be observed that for a Pruning percentage larger than 20%, there is a significant drop in the accuracy of the resulting model. Thus, if the total accuracy of the model needs to be higher than 80%, then the initial model cannot be compressed more than 85%, which is achieved by applying 10% Pruning percentage when Pruning the student model.

6 Discussion

According to the findings of this study, the model compression techniques evaluated were able to retain high levels of accuracy (above 80%) even when the compression percentage was greater than 90%. As a result, it can be argued that the size of a computer vision model, or a DNN in general, can be reduced without sacrificing a significant amount of accuracy, confirming the hypothesis that large computer vision models are frequently over-parameterized. Furthermore, Pruning and Knowledge Distillation model compression techniques were proven to be effective when applied individually to computer vision algorithms and DNNs, and they had even better results when they were combined.

Despite these observations, the proposed solution have limitations. Currently it is not possible to generalize our findings for any computer vision model as each model has its own particularities. Therefore, a manual trial-and-error approach should be performed for each case. Based on the results of this work, a simple recommendation for finding a proper student model in a Knowledge Distillation experiment is to avoid models with a ratio of less than 20% between the number of parameters of the student and the teacher model, implying that the number of parameters should not be reduced by more than 80%.

7 Conclusion and Future Works

The purpose of this work was to reduce the complexity of computer vision models while maintaining high levels of accuracy. The results of this attempt were successful in reducing the complexity and the size of these models while maintaining the accuracy at high levels. In this work, we have shown that the size of a computer vision model can be reduced by 90% while still achieving accuracy higher than 80%. This leads to fewer computations to obtain the result, which means that lower inference time is achieved, and a reduction in computer resources as well. Consequently, real-time processing may be achieved with a compressed version of a model when computations are done locally on the mobile robot's processor. In cases where high precision is required, the original model might be preferred over a compressed version of it.

Due to the scope of this problem, there is room for improvement and future work. A possible future work will be to employ more model compression techniques, such as DNN splitting and neural filtering, and perform more experiments, or even test more combinations for the two techniques that were used in this work. One more possible future work is to evaluate the best-performing model in a real-world scenario and determine if the compressed version of the model produces the desired output without being connected to edge resources. Finally, one last possible future work is to compare a compressed model against an already smaller model, like YOLO, to determine if compressing a model is better than using a smaller one from the beginning.

References

[1] Liang-Chieh Chen, George Papandreou, Florian Schroff & Hartwig Adam (2017): *Rethinking Atrous Convolution for Semantic Image Segmentation*. *CoRR* abs/1706.05587, doi:10.48550/arXiv.1706.05587. arXiv:1706.05587.

[2] Jang Hyun Cho & Bharath Hariharan (2019): *On the Efficacy of Knowledge Distillation*. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4793–4801, doi:10.1109/ICCV.2019.00489.

[3] Xin Dong, Shangyu Chen & Sinno Jialin Pan (2017): *Learning to Prune Deep Neural Networks via Layer-wise Optimal Brain Surgeon*. *CoRR* abs/1705.07565, doi:10.48550/arXiv.1705.07565. arXiv:1705.07565.

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn & A. Zisserman: *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results*. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[5] Eduardo Fernandez-Moral, Renato Martins, Denis Wolf & Patrick Rives (2018): *A New Metric for Evaluating Semantic Segmentation: Leveraging Global and Contour Accuracy*. In: *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1051–1056, doi:10.1109/IVS.2018.8500497.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren & Jian Sun (2016): *Deep Residual Learning for Image Recognition*. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, doi:10.1109/CVPR.2016.90.

[7] Geoffrey Hinton, Oriol Vinyals & Jeff Dean (2015): *Distilling the Knowledge in a Neural Network*. doi:10.48550/ARXIV.1503.02531. Available at `https://arxiv.org/abs/1503.02531`.

[8] Rafia Inam, Elena Fersman, Klaus Raizer, Ricardo Souza, Amadeu Nascimento & Alberto Hata (2018): *Safety for Automated Warehouse exhibiting collaborative robots*. In: *Safety and Reliability–Safe Societies in a Changing World*, CRC Press, pp. 2021–2028, doi:10.1201/9781351174664-254.

[9] Rafia Inam, Klaus Raizer, Alberto Hata, Ricardo Souza, Elena Forsman, Enyu Cao & Shaolei Wang (2018): *Risk Assessment for Human-Robot Collaboration in an automated warehouse scenario*. In: *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, 1, pp. 743–751, doi:10.1109/ETFA.2018.8502466.

[10] Alex Krizhevsky, Geoffrey Hinton et al. (2009): *Learning multiple layers of features from tiny images*.

[11] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár & C. Lawrence Zitnick (2014): *Microsoft COCO: Common Objects in Context*. In David Fleet, Tomas Pajdla, Bernt Schiele & Tinne Tuytelaars, editors: *Computer Vision – ECCV 2014*, Springer International Publishing, Cham, pp. 740–755, doi:10.1007/978-3-319-10602-1_48.

[12] Alberto Marchisio, Muhammad Abdullah Hanif, Faiq Khalid, George Plastiras, Christos Kyrkou, Theocharis Theocharides & Muhammad Shafique (2019): *Deep learning for edge computing: Current trends, cross-layer optimizations, and open research challenges*. In: *2019 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, IEEE, pp. 553–559, doi:10.1109/ISVLSI.2019.00105.

[13] Alberto Marchisio, Muhammad Abdullah Hanif, Maurizio Martina & Muhammad Shafique (2018): *Prunet: Class-blind pruning method for deep neural networks*. In: *2018 International Joint Conference on Neural Networks (IJCNN)*, IEEE, pp. 1–8, doi:10.1109/IJCNN.2018.8489764.

[14] Yoshitomo Matsubara, Davide Callegaro, Sabur Baidya, Marco Levorato & Sameer Singh (2020): *Head Network Distillation: Splitting Distilled Deep Neural Networks for Resource-Constrained Edge Computing Systems*. *IEEE Access* 8, pp. 212177–212193, doi:10.1109/ACCESS.2020.3039714.

[15] Yoshitomo Matsubara & Marco Levorato (2020): *Neural Compression and Filtering for Edge-assisted Real-time Object Detection in Challenged Networks*. *CoRR* abs/2007.15818, doi:10.48550/arXiv.2007.15818. arXiv:2007.15818.

[16] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila & Jan Kautz (2016): *Pruning Convolutional Neural Networks for Resource Efficient Transfer Learning*. *CoRR* abs/1611.06440, doi:10.48550/arXiv.1611.06440. arXiv:1611.06440.

[17] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga & Adam Lerer (2017): *Automatic differentiation in pytorch*.

[18] Naigong Yu, Panna Jiao & Yuling Zheng (2015): *Handwritten digits recognition base on improved LeNet5*. In: *The 27th Chinese Control and Decision Conference (2015 CCDC)*, IEEE, pp. 4871–4875, doi:10.1109/CCDC.2015.7162796.