Final report
Knowledge distillation for echocardiogram view classification

Andris Freimanis
gusandrifr@student.gu.se

Moritz Sprenger
gussprmo@student.gu.se

Raouf Bahsoun
gusbahsra@student.gu.se

Yu-Ping Hsu
gushsuyu@student.gu.se

October 23, 2023

Supervisors
Yinan Yu, academic supervisor, yinan@chalmers.se
Charlotte von Numers, industry supervisor, charlotte.vonnumers@astrazeneca.com
Luis Arevalo, industry supervisor, imarevost@gmail.com

Contents

1 Introduction 3

2 Literature Review 4

3 Methods/Prerequisites 5

4 Experimental setup 6

5 Results 6

6 Limitations 8

1 Introduction

In recent years deep neural networks (DNNs) have become the state-of-the-art in many application domains such as computer vision, natural language processing or game playing using reinforcement learning. These advances can partly be attributed to the availability of more specialized computing power enabling larger models with millions or, especially in the case of large language models, billions of parameters. Training or using these models therefore requires the usage of cloud services or large amounts of storage and computing resources.

With the widespread use of mobile and edge devices in every day life and commercial contexts it is natural to leverage the abilities of DNNs on these devices for a multitude of use cases such as language translation or image classification. With some applications having strict latency, privacy or connectivity requirements, preventing the usage of cloud-based services, it becomes infeasible to use such large models because of memory, computational and energy consumption limitations. This makes it critical to develop methods that are aimed at reducing size and/or inference time of models while keeping a similar performance.

This work gives an overview over the three most used compression techniques for DNNs: pruning, quantization and knowledge distillation. Due to the limited scope and length of this work, the presented approaches only represent a small fraction of the published literature and should mainly serve as an introduction to the field, stressing the main concepts for different distillation techniques.

2 Literature Review

A cornerstone of model compression, knowledge distillation, remains a central focus of this exploration. Li et al. recent work on “Curriculum Temperature for Knowledge Distillation” is an innovative contribution to this field [8]. The article presents the concept of curriculum temperature adaptation, which enables dynamic control over the temperature parameter in knowledge distillation. This dynamic adaptability offers student models not only to gather knowledge from teachers but also adapt to evolving data distributions. This innovative approach has helped make knowledge distillation more adaptive, dynamic, and efficient.

In addition, “Conditional Teacher-Student Learning” by Meng et al. has broadened the perspective of knowledge distillation [10]. Their method introduces a dynamic teacher selection mechanism that tailors the choice of teacher models to the characteristics of input data. This adaptation ensures that the student network can choose from a selection of teacher models, each suited to specific input conditions. This approach promises substantial gains in both efficiency and accuracy, potentially setting the stage for customizing knowledge distillation techniques to real-world data scenarios.

“Learning Efficient Object Detection Models with Knowledge Distillation” by Chen et al. provides an overview of knowledge distillation techniques, highlighting its significance in model compression [1]. The article provides an understanding of the principles and methods employed in knowledge distillation, making ground for the approaches presented by Li et al. [8] and Meng et al. [10]. Meanwhile, the article by Hinton et al. “Distilling the Knowledge in a Neural Network” laid the groundwork for knowledge distillation, introducing the concept of teacher-student model learning and its potential to improve model generalization [6].

Further exploring the literature, “A Survey of Quantization Methods for Efficient Neural Network Inference” by Gholami et al. is an important resource in terms of model compression [3]. This survey presents various quantization techniques, highlighting the potential for significant compression with minimal accuracy loss. The survey by Choudhary et al., “A comprehensive survey on model compression and acceleration”, introduces additional importance to these techniques in the context of resource-constrained deployments [2].

3 Methods/Prerequisites

Given the described initial conditions for our project, we decided to focus on offline response-based knowledge distillation techniques and investigate the trade-off between performance and model size/speed. This can be explained by the availability of an already trained and well performing teacher network and the transferability of response-based implementations between different network structures.

Following the conducted literature review we identified three response-based knowledge distillation techniques suitable for the required task. Following [6] the knowledge of a network can be distilled by matching the *logits* and in the case of (multi-class) classification the softmax distribution of the teacher and student networks. The distillation loss L_D can then be expressed as in equation 1, where $L(.)$ is a loss function and $p(.)$ the softmax function and z_t, z_s the logits for teacher and student respectively [4].

$$L_D = L(p(z_t), p(z_s)) \tag{1}$$

Hinton et al. argue that richer information about similarities between samples and thus more knowledge is transferred between teacher and student, when soft targets are used. They adapt the softmax function by using a scaling parameter T, the temperature, to increase the entropy of the resulting probability distribution. Equation 2 shows this adapted softmax function with z_i as the logit for class i and T as the temperature [6].

$$p(z_i, T) = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)} \tag{2}$$

Combining equations 1 and 2 leads to equation 3, the distillation loss with soft targets where L_{KL} is the Kullback Leibler divergence loss.

$$L_D(p(z_t, T), p(z_s, T)) = L_{KL}(p(z_t, T), p(z_s, T)) \tag{3}$$

To make use of the information in the training data, the student is trained with, the distillation loss is usually combined with the student loss L_S, which is the cross-entropy loss between student outputs and the ground truth labels $L_S = L_CE(y, p(z_s, T = 1))$. In combining these two losses, we arrive at the joint loss given in equation 4, where α is a balancing parameter between the two losses and the multiplication with T^2 is done to normalize the magnitude of the gradients [6]. Training with this loss function as an objective is further referred to as vanilla knowledge distillation.

$$L_J = \alpha(L_D * T^2) + (1 - \alpha)L_S \tag{4}$$

Meng et al. observe that it can improve results to only use the distillation loss when the teacher network makes a correct prediction and name this approach conditional teacher student learning [10]. This simplifies the joint loss L_J by only using the distillation loss with soft labels replaced by the one-hot encoded hard labels for samples where the teacher network makes an incorrect prediction. Thereby, the heuristic tuning of hyperparameter α can be avoided.

Both introduced methods so far make use of a static heuristically chosen hyperparameter T for the knowledge distillation. Zheng et al. argue that a dynamic temperature T, would benefit the distillation process inspired by curriculum learning, where the learning difficulty is increased through the training process [9]. The dynamic temperature is modeled as a temperature module that gets learned in an adversarial manner. The temperature module is optimized in the opposite direction of the student, trying to maximize the distillation loss with the update shown in equation 5 [9].

$$\theta_{temp} = \theta_{temp} + \mu \frac{\partial L}{\partial \theta_{temp}} \tag{5}$$

To adopt the curriculum learning and change the difficulty of the task throughout the training, the loss w.r.t. the temperature is scaled by λ, leading to equation 6 [9].

$$\theta_{temp} = \theta_{temp} + \mu \frac{\partial(\lambda L)}{\partial \theta_{temp}} \tag{6}$$

λ increases with each epoch E_n during the training according to either a linear schedule or a cosine schedule given in equation 7, where λ_{min} and λ_{max} are the range for λ and E_{cap} is the epoch at which λ reaches it maximum and stays constant. The linear schedule also adopts the same methodology with E_{cap}.

$$\lambda_n = \lambda_{min} + \frac{1}{2}(\lambda_{max} - \lambda_{min})(1 + cos((1 + \frac{min(E_n, E_{cap})}{E_{cap}})\pi)) \tag{7}$$

Adopting their implementation of a global temperature, T is a single learnable parameter used for each sample in a batch before being updated with its gradient [9]. Following Zheng et al. we set λ_{min} and λ_{max} at 0 and 1 as well as E_{cap} at 10 as our default values.

4 Experimental setup

In accordance with the project goals an ablation study with 4 different student architectures and the 3 described distillation techniques is conducted. As for student architectures we focused on efficient convolutional architectures especially designed for mobile and edge devices. The four considered architectures are *Mobilenet-V3-small*, *Mobilenet-V3-large* [7], Shufflenet [12] and Ghostnet [5]. All network implementations are taken from PyTorch [11]. To confirm with the expected input sizes of these architectures, the greyscale image data is duplicated along the channel axis. Additionally, images are resized to $112px$ sized square images and normalized. Throughout all experiments a batch size of 256 is used and models are trained for 20 epochs with early stopping. For all 4 student architectures experiments are performed with the 3 different knowledge distillation techniques and training from scratch without any distillation. For vanilla knowledge distillation different values for the temperature are investigated. Conditional knowledge distillation is conducted with a temperature of 4. For the curriculum-based knowledge distillation linear and cosine decay are investigated. As performance metrics we report validation accuracy and validation $F1$-score. For size and speed metrics, the inference and train speed are reported as well as the model size in *Megabytes* and the number of trainable parameters. All experiments are conducted with a single *Nvidia A40* GPU.

5 Results

In Figure 1, the validation accuracy of the teacher model is represented by a red dashed line, while the best Knowledge Distillation (KD) model and the model trained from scratch for each of the four student architectures are also plotted. It's interesting to note that the validation accuracies of the KD models are not only closely aligned with that of the teacher model, indicating successful knowledge transfer, but they also marginally yet consistently outperform the models trained from scratch for each student architecture. This suggests that all architectures perform comparably well and that KD provides a slight but consistent advantage. The validation accuracy is very close to the teacher for all student architectures.

In our experiments, each student architecture had a different set of optimal hyperparameters. The GhostNet model achieved its best performance using curriculum learning with a linear decay schedule for temperature and a loss rate of 0.5. For the MobileNet V3 Large and ShuffleNet models, the optimal performance is obtained with a temperature parameter of 2 and a loss rate of 0.5. On the other hand, the MobileNet V3 Small model performs best when curriculum learning is combined with cosine decay schedule for temperature and a loss rate of 0.5.

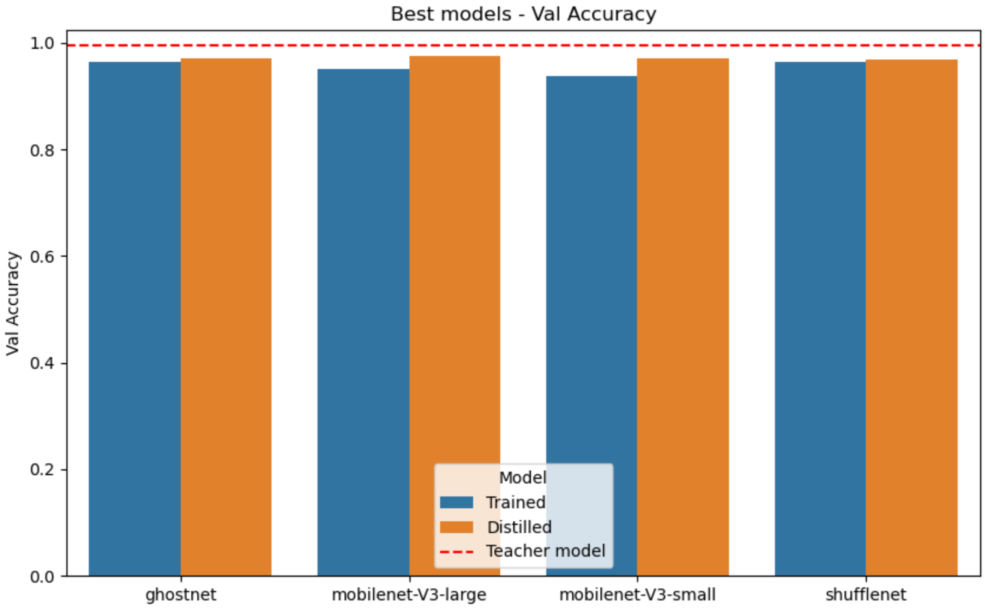

Figure 1: Results for different student architectures

In Figure 2, a scatter plot is presented that illustrates the relative validation accuracy (compared to the teacher model) and inference speedup of each student model, both for models trained using Knowledge Distillation (KD) and those trained from scratch. The thickness of each point represents the memory footprint of the model in MB.

All KD student models achieved similar performance, with relative validation accuracies ranging from 0.975 to 0.98 and inference speedups between 1.35 and 1.4 times. However, the models trained from scratch showed a decrease in accuracy down to 0.95. This reinforces the effectiveness of KD, as it consistently outperforms training from scratch.

Interestingly, there are some variations in the inference speed even between models with the same architecture. This could be attributed to slightly inconsistent allocation of hardware resources and memory index management. Despite these variations, the inference speed is generally close for all architectures.

Among all models, MobileNet V3 Large had the highest relative validation accuracy at 0.98. The two smaller models, MobileNet V3 Small and ShuffleNet, achieved slightly higher inference speedups of 1.4 times.

These results suggest that KD enables student models to achieve comparable performance to their teacher models while benefiting from increased inference speed and reduced memory usage.

Figure 3 provides a focused view on only the KD models, further emphasizing these observations.

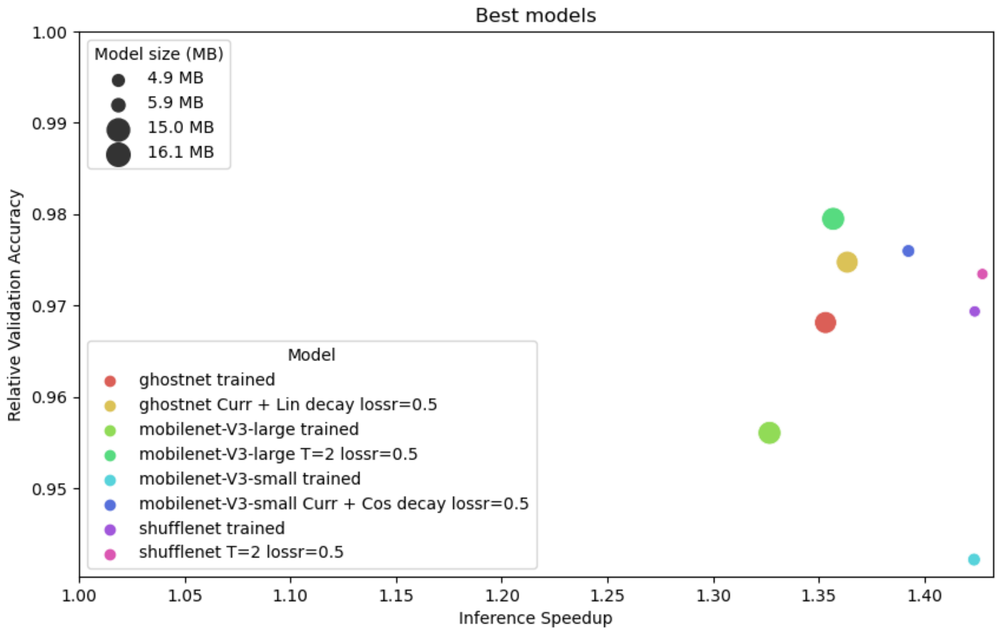

Figure 2: Results for different student architectures

6 Limitations

Due to the limited available time for the project and project specific circumstances multiple limitations have to be considered regarding the achieved results. Because of significant training times and the time constraints set by the project course the search space for different hyperparameter combinations was heavily restricted. With the same reasoning no statistical significance testing was performed and experiments for each setup only performed once. This weighs even more heavy for this project since the student model reach very good results even without knowledge distillation and the different distillation techniques are very close in performance. Therefore, the results can only be interpreted as indications, especially regarding the differences between different knowledge distillation techniques. For a few experiments a batch size of 32 was also tested and achieved slightly better results but due to the increased training time with a reduced batch size, this was not further pursued. The captured data for training and inference speed is very noisy likely because of the shared nature of the computing resource, therefore for speeds the median batch time or the minimum epoch times are reported.

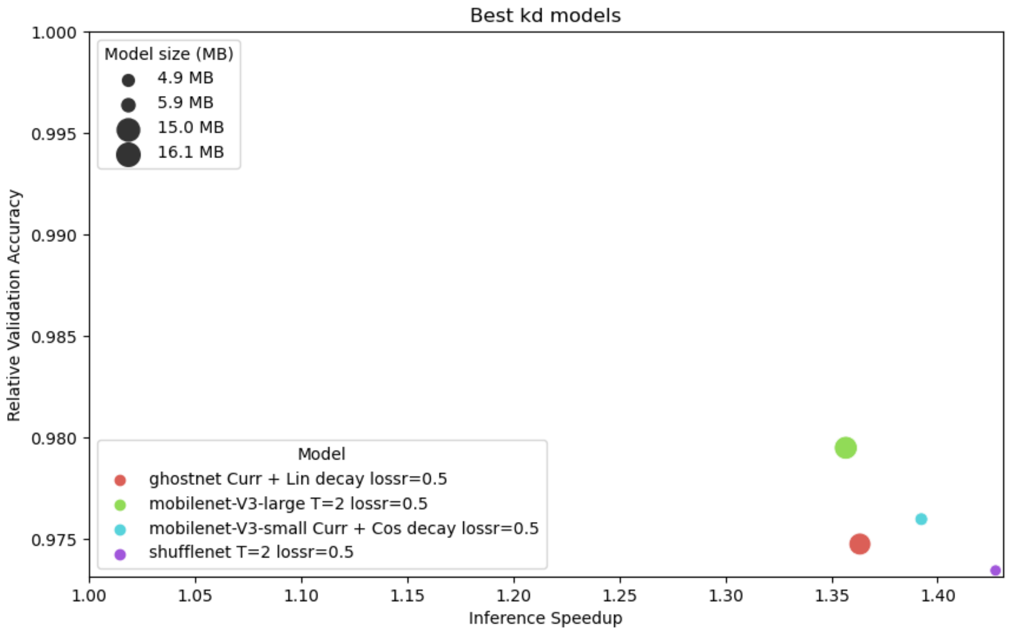

Figure 3: Results for different student architectures

”The computations/data handling/[SIMILAR] were/was enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) at [NAISS AFFILIATED SITE] partially funded by the Swedish Research Council through grant agreement no. 2022-06725.”

Appendix

References

[1] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS’17, pages 742–751. Curran Associates Inc., 2017.

[2] Tejalal Choudhary, Vipul Mishra, Anurag Goswami, and Jagannathan Sarangapani. A comprehensive survey on model compression and acceleration. *Artificial Intelligence Review*, 53(7):5113–5155, 2020.

[3] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. A Survey of Quantization Methods for Efficient Neural Network Inference, 2021.

[4] Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. Knowledge Distillation: A Survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

[5] Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. Ghostnet: More features from cheap operations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1580–1589, 2020.

[6] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the Knowledge in a Neural Network, 2015.

[7] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019.

[8] Zheng Li, Xiang Li, Lingfeng Yang, Borui Zhao, Renjie Song, Lei Luo, Jun Li, and Jian Yang. Curriculum temperature for knowledge distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 1504–1512, 2023.

[9] Zheng Li, Xiang Li, Lingfeng Yang, Borui Zhao, Renjie Song, Lei Luo, Jun Li, and Jian Yang. Curriculum temperature for knowledge distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 1504–1512, 2023.

[10] Zhong Meng, Jinyu Li, Yong Zhao, and Yifan Gong. Conditional teacher-student learning. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6445–6449. IEEE, 2019.

[11] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[12] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6848–6856, 2018.