# Pruning and resizing deep neural networks for FPGA implementation in trigger systems at collider experiments

**D. Mascione[1,2,3], M. Cristoforetti[2,3], A. Di Luca[2,3], F. M. Follega[1,3], R. Iuppa[1,3] and A. Saccardo[1]**

[1]Università degli Studi di Trento, Via Sommarive 14, 38123 Trento, Italy
[2]Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy
[3]TIFPA, Via Sommarive 14, 38123 Trento, Italy

E-mail: daniela.mascione@unitn.it

**Abstract.** Deep Learning algorithms are a powerful tool for performing a fast and accurate selection of interesting data in collider experiments. In particular, they could be successfully employed at the trigger level, taking advantage of high-performance hardware such as Field Programmable Gate Arrays. However, this may require resizing Deep Neural Networks to fit available resources. To this end, we have developed a new pruning technique. The proposed method is simple and versatile and is well-suited to different types of neural networks. It is effective in reducing the overall size of the network by eliminating unnecessary nodes, and respects the final size constraints determined by the user.

## 1. Introduction

One of the major challenges in collider experiments is the generation of a significant amount of data, which needs to be handled carefully. Frequently, limitations in data storage capacity result in the recording of only a part of all available data [1]. To determine which events should be saved for later analysis, a process known as trigger is employed. Trigger systems make use of parallelized architecture to perform data selection and can be divided into various levels, each utilizing distinct hardware. Each selection level takes as input the previous level's output and performs an additional selection to improve the accuracy of the result. The first level of trigger systems primarily rely on specialized electronic circuits, including custom Application Specific Integrated Circuits (ASICs), Programmable Logic Devices (PLD), Field Programmable Gate Arrays (FPGAs), and discrete logic components like Random-Access Memories (RAM). However, with the significant advancements in FPGA technology in terms of speed and capacity, many trigger systems are transitioning towards complete implementation in FPGAs [2].

Preventing the loss of crucial information in the trigger pipeline is a big concern, fueled by the enormous quantity of data generated. Deep Learning algorithms, which have shown to be incredibly helpful and efficient when dealing with big volumes of data [3], can be of use in this regard. In an effort to take fast and accurate decisions, the High Energy Physics community has undertaken a number of initiatives to investigate the potential use of Deep Learning algorithms in the selection phases, particularly in the first level of the trigger chain. It is now relatively easy

to implement Deep Neural Networks (DNNs) onto the FPGAs that are utilized in the initial stages of the trigger [4]. However, successful DNNs can require vast amounts of computation and memory and are therefore not always compatible with the programmable resources available on FPGAs, such as the number of logic units and memory slots. Some modifications are usually required to adapt DNNs for implementation on FPGAs [5].

One approach to modify DNNs is to remove unimportant, redundant, or unnecessary components through a process called network pruning. There are numerous ways to prune DNNs, but it is challenging to find an ideal standard that guarantees optimal performance while adhering to the size constraints set by FPGAs. To address this issue, we propose a new pruning technique developed to remove unnecessary nodes from DNNs while respecting size restrictions. Our technique automatically identifies the most effective nodes of each layer while strictly preserving the final size limit, allowing for the selection of the best-performing network architecture that fits within the available FPGA resources. This method proved to be easy and versatile and can be integrated into various types of existing Deep Learning models.

## 2. Network Pruning

DNNs are computational models made up of many layers of interconnected fundamental computing units called artificial neurons or nodes. Their task is to transform weighted inputs into outputs. By changing the weights of the connections between neurons in the various layers, the network learns to identify patterns in the input data in a process called training. DNNs can be formed by a significant number of layers, nodes, and connections. To reduce the size of a network it is necessary to remove some of those components. Network pruning is a class of techniques used for this purpose that involves the removal of some structural parts, without significantly affecting the network performance.

Several pruning strategies in use concentrate on eliminating single connections. This is a quick and simple method for shrinking a model, but such a pruning technique will result in a sparse network, which may be challenging to train [6]. On the other hand, acting on neurons or layers can be less straightforward but results in a smaller, faster, and more resource-efficient DNN without affecting the network's architecture.

Another important aspect of pruning is deciding when to remove the network components. Pruning can be done during the training phase or after the network has been trained. Most pruning algorithms for neural networks perform the pruning after the training following a similar process: first, the network is trained until it converges, and then, each structural element in the network is given a score, and the network is pruned based on these scores [7]. This way of pruning reduces the accuracy of the network, so it needs to be retrained to recover - a process called fine-tuning. This process of pruning and fine-tuning is often repeated several times, gradually reducing the size of the network. As a result, this pipeline can be time-consuming and may not necessarily be the most convenient pruning strategy.

Based on this and the previous considerations, we looked into an alternative approach to pruning DNNs. Instead of concentrating on individual connections, our technique focuses on removing superfluous nodes. Pruning takes place during the training phase and a subsequent fine-tuning campaign is not required. Moreover, the user can choose the ultimate dimension of the pruned network. This approach involves adding a shadow network on top of the DNN that needs to be resized. The shadow network consists of neurons that have only one connection to each node of the previous layer of the original network and multiple connections to the nodes of the subsequent layer. During training, the calculations performed by the shadow nodes are such that their output will be zeroed when they are connected to unnecessary nodes, acting thus as a filter. Consequently, only a portion of the neurons will be employed for learning, with no information passing through the remaining nodes. The training process is optimized for learning with exactly the number of nodes desired by the user, while the selection of which neurons to

use for learning is automatically determined by the shadow network.

This approach allows for both complete and partial DNN pruning, according to where the shadow network is configured. Our method is also very versatile and showed promising results both on Fully Connected DNNs (FC-DNNs), where every neuron in one layer is connected to every neuron in the next layer, and Convolutional Neural Networks (CNNs), where neurons are a set of filters that are applied to small regions of the input data. This new pruning method allows for high-quality pruning that truly follows the specified overall compression rate. Our method turned out to be very promising and can potentially be used for future DNNs implementations on FPGAs.

## 3. Tests and results

Two different types of networks, FC-DNNs and CNNs, were used to test our pruning method. Although they have been employed in different fields, both network types were utilized for classification purposes.

### *3.1. Pruning FC-DNNs*

For testing our pruning method with FC-DNNs, a network created to address a particular High Energy Physics problem has been deployed. The DNN utilized in testing was developed to discriminate between jets containing $b$-quarks produced by boosted Higgs boson decay in proton-proton collision experiments and the irreducible background produced by QCD multi-jet generation. The FC-DNN consisted of 4 hidden layers with 64 nodes per layer, for 40 input variables. The goal was to distinguish between the Higgs boson decay and the background without taking pile-up effects into consideration. In separate training campaigns, the same initial DNN was pruned changing the number of desired nodes to be used for learning.

To evaluate the accuracy of the pruned models, the background rejection rate as a function of the Higgs tagging efficiency has been examined. For each tagging efficiency value, a higher rate of background rejection denotes better DNN performance. The plot in Figure 1 shows that requiring more active nodes results in better performance, as expected; this suggests that just the required fraction of nodes are employed for learning, while the remaining nodes have not been used in the training.

The plot in Figure 2 displays another interesting result. Once the pruning is done, the shadow network and the unused nodes from the main network can be removed. As a result, a DNN with a more condensed architecture is produced, which can then be trained as a new independent model. Test results have demonstrated that the DNN performance obtained is comparable to that obtained by pruning the original network while training it. This validates the hypothesis that integrating training and pruning into a single activity can be an interesting alternative to the conventional training-pruning-retraining pipeline and verifies that the nodes actually employed for the learning task are the ones selected by our method.

### *3.2. Pruning CNNs*

A simple architecture has been used for preliminary tests to evaluate our pruning strategy with CNNs. The network structure is that of LeNet-5 [8], with two convolutional layers, two pooling layers, and three fully connected layers. The shadow network has been overlaid to the convolutional block and the model has been trained using the popular image classification dataset Fashion-MNIST [9]. As with the FC-DNN test, the same initial CNN was pruned in different training campaigns with a different desired number of filters to be used for training. Table 1 displays the results of the top-1 accuracy obtained on the validation set for different fractions of filters required. The accuracy increases as the percentage of desired filters increases, indicating that the constraint on the number of filters to be used has been satisfied and some filters have actually been pruned.

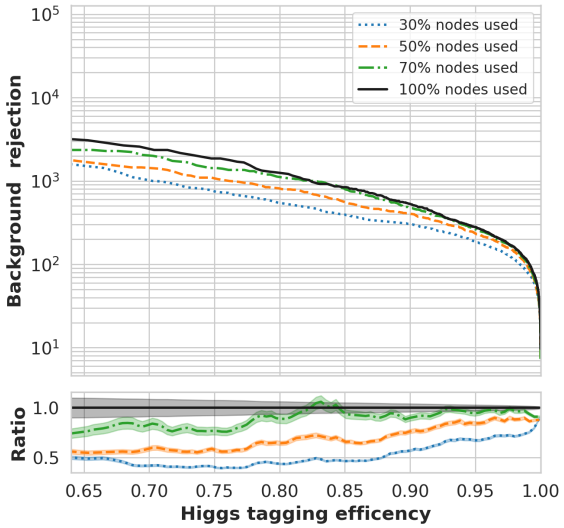


**Figure 1.** Background rejection versus Higgs tagging efficiency for models pruned during training with different percentages of nodes required.

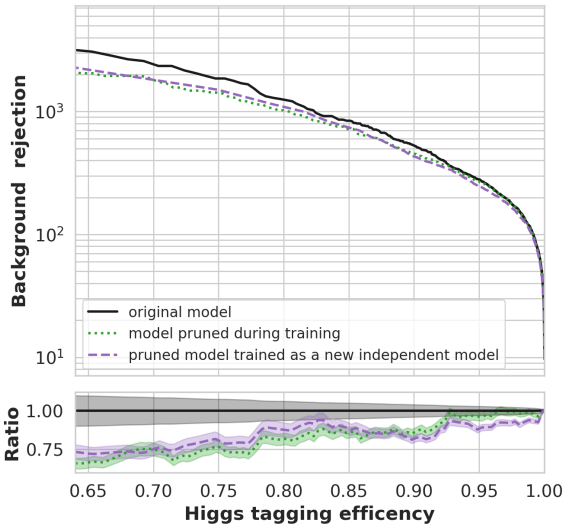


**Figure 2.** Background rejection versus Higgs tagging efficiency for the original model, the model pruned during training, and the pruned model trained as a new independent DNN obtained with the 70% of nodes of the original model.

## 4. Conclusions

Collider experiments produce vast amounts of data that can not always be entirely stored, and it is crucial to take advantage of the implementation of DNNs on FPGAs for performing a fast and accurate data selection at the trigger level. However, DNNs need to be properly optimized before being implemented on FPGAs. In order to downsize DNNs within the limitations imposed by the constrained resources of FPGAs, we developed a novel pruning strategy. With our approach,

**Table 1.** Top-1 accuracy of a LeNet-5 architecture trained on the Fashion-MNIST dataset and pruned during training with different percentages of filters required. All the accuracies are tested on the validation set.

| Percentage of filters required | Top-1 accuracy |
|---|---|
| 100% | 88.71% |
| 70% | 87.98% |
| 30% | 73.18% |

the total number of nodes in the neural network is reduced and the final network dimensions can be determined by the user. Pruning is completed during the training itself, with no need for separate training before or after the pruning. The proposed method has produced interesting results both for FC-DNNs and simple CNNs architectures. Although further investigation with more complex models is required, the preliminary results are encouraging and suggest that our pruning approach can be successfully used for resizing DNNs employed in trigger systems of collider experiments.

**Acknowledgments**
The work described in this paper has been carried out in a joint effort by the members of the *deepPP* initiative of the University of Trento and Fondazione Bruno Kessler. Please visit https://www.deeppp.eu/ for contacts and information about the group's activities.

**References**
[1] Clissa L 2022 Survey of Big Data sizes in 2021 *Preprint* arXiv:2202.07659
[2] Smith W H 2020 Triggering and High-Level Data Selection *Particle Physics Reference Library* vol 2 ed C W Fabjan and H Schopper (Cham: Springer) p 533
[3] Najafabadi M M, Villanustre F, Khoshgoftaar T M, Seliya N, Wald R and Muharemagic E 2015 *J. Big Data* **2** 1.
[4] Duarte J *et al* 2018 *JINST* **13** P07027
[5] Cheng Y, Wang D, Zhou P and Zhang T 2018 *IEEE Signal Processing Magazine* **35** 126
[6] Evci U, Pedregosa F, Gomez A and Elsen E 2020 The Difficulty of Training Sparse Neural Networks *Preprint* arXiv:1906.10732
[7] Blalock D, Gonzalez Ortiz J J, Frankle J and Guttag J 2020 *Proc. Mach. Learn. Syst.* **2** 129
[8] Lecun Y, Bottou L, Bengio Y and Haffner P 1998 *Proc. IEEE* vol 86 no 11 pp 2278-2324
[9] Xiao H, Rasul K and Vollgraf R 2017 Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms *Preprint* arXiv:1708.07747