NEURAL NETWORK(s) PRUNING : ONE AND ENSEMBLES

SUMANTH MANDURU, smanduru@gmu.edu
DILEEP KUMAR LATCHIREDDI, dlatchir@gmu.edu

DNNs and E-DNNs have achieved impressive results in real-world applications, but their deployment on resource-constrained devices is challenging due to high computational costs and memory requirements. To address this issue, model compression techniques are being used to accelerate DNNs while preserving their performance. Our empirical experiments have shown that DropNet, an iterative pruning approach, is robust across a range of scenarios. We have demonstrated that DropNet can remove up to 70-80% of filters in larger networks like ResNet and VGG, with no significant accuracy degradation. Furthermore, the pruned network maintains its strong performance even after weights and biases are reinitialized. Our proposed algorithm combines DropNet with a Hierarchical Ensemble pruning method that reduces the computational costs and memory requirements of E-DNNs. HQ selects a subset of high-performing E-DNNs from a set of M models, with the goal of achieving better performance than the original set while reducing the size to S. Our experiments indicate that pruned DNNs can form better ensembles than their unpruned counterparts. By combining pruned models, we were able to achieve better accuracy while also reducing computational costs and memory requirements. This algorithm is specifically tailored for deployment on resource-constrained devices.

CCS Concepts: • **Neural Network Simplification** → **Model Pruning**; **Model Size Reduction**; • **Ensembles of Deep Neural Networks** → **Hierarchical Pruning**.

Additional Key Words and Phrases: ResNet, VGG19, Deep Convolutional Neural Networks, Pruning the Layer Nodes and Weights, Model Compression, Fully Connected Neural Networks

1 INTRODUCTION

Deep Neural Networks (DNNs) have demonstrated extraordinary effectiveness in a variety of domains, but training them demands enormous computer resources, time, and memory owing to their millions or billions of parameters. As a result, they are difficult to train and can only be used on high-performance computer equipment requiring high computational resources. To overcome these challenges, researchers are exploring methods to reduce the complexity of DNNs. The smaller network learns to emulate the bigger network's behavior, decreasing complexity while retaining performance. DropNet - Iterative pruning is one such technique that involves removing the least important nodes or filters from the network based on their post-activation values which employs both the information about the input weights and the input data to make an informed decision about which nodes/filters to drop. Hierarchical Ensemble Pruning (HQ) is another strategy for picking a smaller number of deep models from a larger collection of models by grouping comparable models together and selecting the best representative model from each group. These strategies drastically reduce the complexity of DNNs and E-DNNs, resulting in significant savings in time, power, and computational resources. As a result, these models become more accessible to a wider range of devices and applications. We should expect even more efficient and effective ways to develop as research in this sector progresses, significantly enhancing the capabilities of deep learning models.

2 PROBLEM DEFINITION

The utilization of deep networks with millions or even billions of parameters is critical to the success of Deep Neural Networks (DNNs) and Ensembled Deep Neural Networks (E-DNNs) in real-world applications. The emergence of powerful GPUs that enable efficient processing has aided this breakthrough. However, we intend to address the model compression challenge, specifically reducing the size and number of parameters in these networks. Our problem is divided into the following sub-problems:

- DropNet - Iterative Pruning
- Hierarchical Ensemble Pruning (HQ)

Our objective is to compress Deep Convolutional Neural Network architectures in order to minimize their size and parameters while preserving equivalent accuracy to the original model. In milestone 2, we have compared Model A (Fully Connected) and Model B (2 layered CNNs). Now, We have considered another two types of model architectures.

- Model C - Deep convolutional Neural Networks ($\geq$ 4 CNN Layers)
- Bigger architecture models like ResNet20[1], VGG11, VGG13, VGG16, and VGG19[2]

2.1 Extension to Milestone 2 - Novel Approach

In milestone 2, we successfully replicated the results and methodology of DropNet - Iterative Pruning and HQ Algorithms. In this next phase, we aim to merge these two techniques to create an ensemble of pruned models. Specifically, we plan to use the pruned models of ResNet and various VGG variations like VGG11, VGG16 and VGG19 generated by Iterative Pruning as input for the Hierarchical Ensemble Pruning Algorithm. This approach will allow us to group similar models together and select the most representative model from each group, resulting in an ensemble version of pruned models.

- **Goal :** Our goal is to decrease the size of pruned Ensembled Deep Neural Networks (E-DNNs) from a pool of pruned models. By utilizing an ensemble approach, we aim to demonstrate that these pruned E-DNNs can achieve high accuracy and robustness while also being more efficient in terms of time and space requirements for ensemble decision-making.
- **Motivation :** The motivation behind this novel approach of combining Iterative Pruning and Hierarchical Ensemble Pruning Techniques is, the potential to provide significant advantages over using either technique alone. By leveraging the strengths of both methods, we can create an ensemble of pruned models that offer improved performance and reduced complexity, making them more accessible and practical for various applications. This novel approach has the potential to advance the field of deep learning and open up new avenues for research and development.

3 METHODOLOGY

In this section, we will be discussing the DropNet [3] and Hierarchical Ensemble (HQ) [4] algorithms. DropNet is an iterative algorithm that identifies and removes filters based on their lowest average post-activation value across all training samples. This can be done either in a layer-wise or global manner. On the other hand, Hierarchical Ensemble Pruning is a technique used to identify a smaller set of deep models from a given set of models using focal diversity techniques.

3.1 DropNet - Iterative Pruning

We start with a dense feed-forward neural network $f(x; n)$, which has an initial configuration of weights and biases $\theta = \theta_0$ and filters n. We train the network until it reaches a minimum validation loss l and achieves a test accuracy of a. Next, we introduce a binary mask m, which has the same size as the nodes/filters and consists of elements from the set 0,1. Initially, the mask is set to all 1s, indicating the presence of all filters. We then train the function $f(x; m \odot n)$, where $m \odot n$ represents element-wise multiplication between m and n, resulting in an updated initialization of $f(x; m \odot n)$. We continue training until f attains the lowest validation loss $l^{'}$ with a corresponding test accuracy of $a^{'}$ while using the mask.

3.2 Hierarchical Ensemble Pruning

The proposed approach combines three ensemble selection techniques: focal diversity metrics, hierarchical pruning, and diversity consensus voting-based focal diversity pruning. Firstly, a deep neural network ensemble is trained with various backbones, and then focal diversity metrics are used to prune the ensemble. This measures the diversity of models in the ensemble and ensures they are different enough to generalize well to unseen data. Hierarchical pruning further reduces the ensemble size by partitioning it into smaller subsets based on diversity and performance, creating a hierarchical structure. The diversity consensus voting-based focal diversity pruning step removes low-performing models with negative correlations to other models in the ensemble. This approach significantly reduces the search space for high-performance deep ensembles and eliminates ensembles lacking diversity. The final ensemble size can be set to limit time and space complexity.
Input : Given a pool of DNNs with initial state θ_0, initial filters n, epochs j, mask m and a pruning metric. Pruning metrics are listed in Table 1.
Output : Ensembles of Pruned DNNs Models
Hyper-parameters : pruning fraction p $\in$ (0, 1]
Algorithm : The algorithm for the proposed Iterative Pruning followed by Ensemble Pruning approach is presented in Algorithm 1.
Importance Score : To calculate the importance of a filter in CNNs, we use Expected Absolute Value. Filters in CNNs consist of constituent nodes, and to determine which filter to drop, we must evaluate the importance of its constituent nodes. To calculate the importance score of a filter $f_i, i \in \mathbb{Z}^+$ among n filters in a network, we find the average of the expected absolute values of all constituent nodes, $E(f_i) = \frac{1}{r} \sum_{j=1}^{r} |E(a_i)|$, where $E(f_i)$ represents the expected absolute value of the filter f_i across all training samples $(x_1, x_2, ..., x_t)$.

3.3 Investigative Questions

The question we want to investigate is whether ensembles of pruned models can achieve better performance, and if we can apply the HQ Algorithm on the pool of pruned models to further reduce computational costs and memory requirements. By pruning multiple models and combining them into an ensemble, we can potentially achieve better performance than a single non-pruned model. By applying the HQ Algorithm on a pool of pruned models, we can potentially further reduce the size of the ensemble without sacrificing performance.

Algorithm 1 DropNet - Iterative Pruning and Hierarchical Ensemble Pruning

Require: Ensembles of pruned models from a pool of $f(x; m \odot n)$ sub-networks, $M^{'}$
Ensure: $a^{'} \approx$ a (similar accuracy) and $\| m \|_0 \ll$ n (fewer parameters), $\forall\ a \in M$
Initialize starting mask m = $\{1\}^{|n|}$
while Validation Accuracy $a^{'} \ll ka$ **do**
1. Reset the network to initial state θ_0
2. Mask the nodes/filters - f(x; m $\odot$ n)
3. Train the model with early stopping for at most j iterations
4. Use the pruning metric to select a proportion p of nodes (or) filters to remove and update m.
end while
A pool of $f(x; m \odot n)$ sub-networks, $M^{'}$ are stored
for m in $M^{'}$ **do**
1. Compute accuracies
2. Calculate focal diversity metrics and prune ensembles using diversity score
end for
while Partition the ensembles recursively **do**
1. Calculate accuracy for ensembles
2. Rank by their focal diversity scores
3. Prune out the $\beta\%$ of the ensembles with large diversity scores
3. Prune all subsequent ensembles that are super-sets of pruned ensembles
end while

Pruning Metric	Global Type	Layer-wise Type	Importance Score
Min	✓	✗	$\mathrm{E}[a_i]$ or $\mathrm{E}[f_i]$
Max	✓	✗	$-\mathrm{E}[a_i]$ or $-\mathrm{E}[f_i]$
Random	✓	✗	0
Min Layer	✗	✓	$\mathrm{E}[a_i]$ or $\mathrm{E}[f_i]$
Max Layer	✗	✓	$-\mathrm{E}[a_i]$ or $-\mathrm{E}[f_i]$
Random Layer	✗	✓	0

Table 1. Significance of Pruning Metrics

Therefore, our investigation would involve pruning multiple models and creating ensembles of both non-pruned and pruned models. We would then compare the performance of the ensembles and determine whether the pruned ensembles outperform the non-pruned ensembles. If so, we would then apply the HQ Algorithm on the pool of pruned models to further reduce the size of the ensemble and compare the performance with the original pruned ensemble.

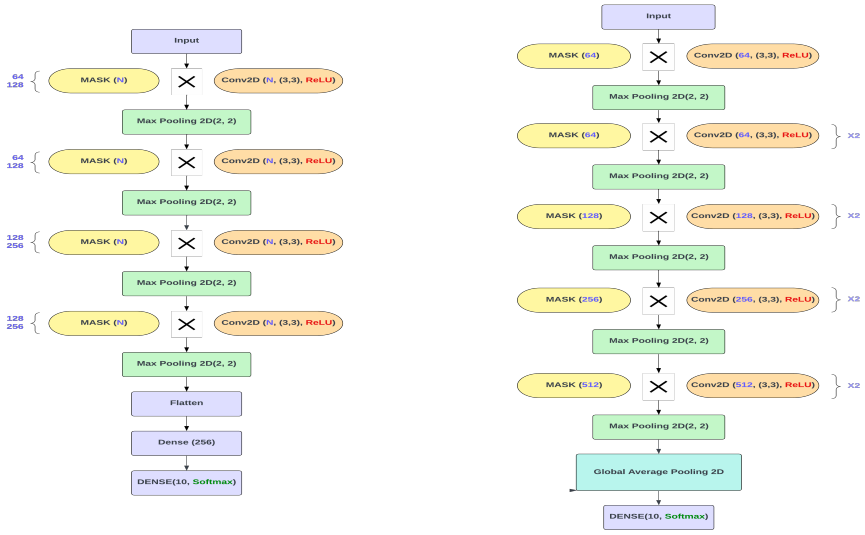

Fig. 1. Model Architectures considered for DropNet - Model C (Left Side) and ResNet (Right Side). The number of nodes/filters tested is shown in blue alongside the CNN layer in the design.

4 EXPERIMENTAL RESULTS

To evaluate the performance of DropNet algorithm and HQ Algorithm, we do empirical tests with Deep Convolutional Neural Networks (DCNNs) like ResNet and variations of VGG. The pruning pipeline is illustrated in Figure 2.

4.1 Dataset Description

- For both algorithms, we have utilized the same dataset. The details of the dataset as follows:
  - **Description :** The CIFAR-10 dataset (Canadian Institute For Advanced Research) is one of the most common datasets to train machine learning and computer vision algorithms. It is a set of images collected in machine learning research. The CIFAR-10 dataset comprises 60,000 color pictures of 32x32 shape divided into ten categories - Airplanes, Vehicles, Birds, Cats, Deer, Dogs, Frogs, Horses, Ships, and Trucks. Each class has 6,000 photos.
  - **Link :** https://www.cs.toronto.edu/~kriz/cifar.html or https://pytorch.org/vision/main/generated/torchvision.datasets.CIFAR10.html

4.2 Experimental Design

- **DropNet - Iterative Pruning**
  - **Train-Validation-Test Split :** The 60K CIFAR-10 dataset is divided into (80%) 54,000 samples for training, (10%) 6,000 samples for validation, and 10,000 samples for testing.
  - **Pre-processing :** The pixel values of the input images are divided by 255 to fall within the range of 0 to 1.

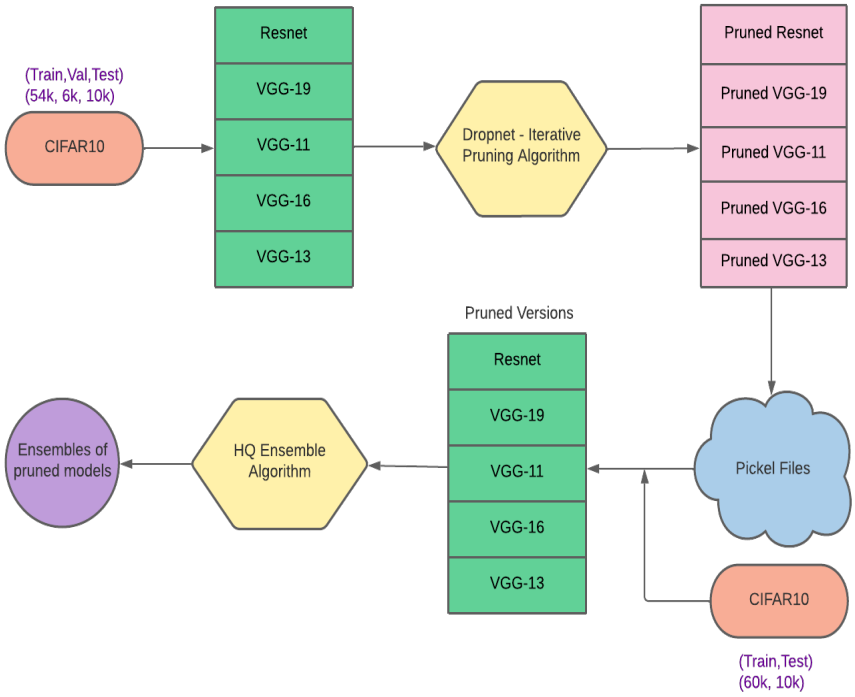

Fig. 2. Flow Chart of the Novel Approach, combining both DropNet - Iterative Pruning and HQ Ensemble Algorithm.

  - **Activation Function :** Except for the last layer, which uses Softmax as its activation function, all layers have ReLU as their activation function.
  - **Optimizer & Loss Function:** With a learning rate of 0.1, Stochastic Gradient Descent (SGD) is used as Optimizer. Cross Entropy is used as the Loss Function.
  - **Batch Size:** All the experiments were conducted with a batch size of 128.
- **Hierarchical Ensemble Pruning (HQ)** We have followed the same experimental setup for the later phase (HQ Algorithm) of the approach.
  - **Train-Validation-Test Split :** 50000 training images and 10000 test images.
  - **Models Used :** ResNet, VGG11, VGG13, VGG16, and VGG19.
  - **Diversity Metrics Used :** Kohavi-Wolpert variance (KW), Binary Disagreement (BD) , Cohen's kappa (CK), Generalized diversity (GD),

4.3 Code

- **DropNet - Iterative Pruning**
  - The source code is available at https://github.com/tanchongmin/DropNet for Model C, ResNet and VGG19 in which the models were built using Keras Framework and Numpy.

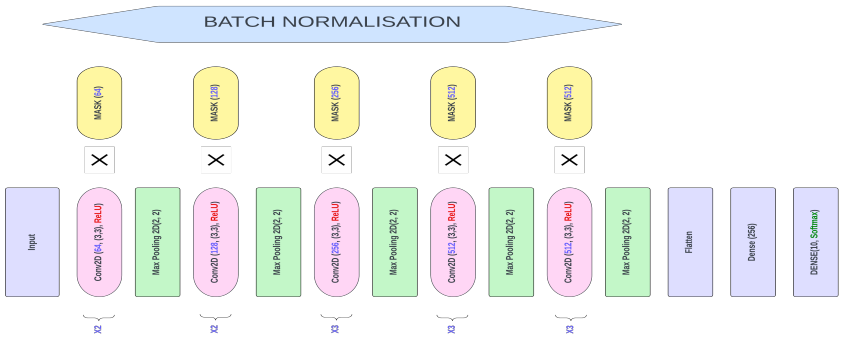

Fig. 3. Model Architecture for VGG16. Variations of VGG (11 and 19) are formed by adding/removing Convolutional layers in the same architecture.

  - In reference to the source code, we have built the models from scratch using PyTorch Framework. There are two set of models - Model C and Large Models - ResNet, VGG - 11, 13, 16, 19 illustrated in Figure 1 and Figure 3.
  - There is no code or experiments conducted on variations of VGG. We have developed from scratch for the variations of VGG11 and VGG16
  - We used the source code functions to visualize the findings and made modifications to our PyTorch model configuration accordingly.
- **Hierarchical Ensemble Pruning**
  - The source code is available at https://github.com/git-disl/HQ-Ensemble.
  - We have developed code to construct ResNet, VGG11, VGG13, VGG16, and VGG19 models and enhanced it to facilitate the execution of all these models and preserve their weights.
  - The source code provides the basic framework, and we have created Python code snippets to implement and execute it.

4.4 Hyperparameter and Optimization

- **DropNet - Iterative Pruning**
  - **Pruning metric :** A pruning metric is a parameter that is used to evaluate the importance of each node/filter in a neural network, and the ones with the lowest importance scores will be dropped during the pruning process. There are two types of pruning metrics: global and layer-wise. Each pruning type has three metrics, namely Minimum, Maximum, and Random. The minimum metric globally drops a fraction p of the nodes/filters whose $E(a_i)$ or $E(f_i)$ is lower, while the maximum metric globally considers negative $E(a_i)$ or negative $E(f_i)$ to prune. A fraction p of nodes is randomly dropped globally under the random metric as a control in the experiment. Similarly, layer-wise pruning metrics prune a fraction p of

nodes/filters layer-wise. The figure 6 (Left Side) illustrates the early stopping on Model C - Conv64x2 - 128x2.

- **Early Stopping - Patience :** During our experiments, we set an early stopping constraint of 100 epochs with a patience of 5. We found that the VGG19 model converged before 50 epochs, except for the minimum layer and random pruning metrics, which converged around 60 epochs. Similarly, the VGG16 model also converged before 50 epochs, with the exception of the random layer metric, which converged at 60 epochs. For ResNet, all pruning metrics except for the random layer converged around 60 epochs, while the random layer converged at 80 epochs. Finally, for the VGG11 model, all metrics converged mostly before 50 epochs.

- **Hierarchical Ensemble Pruning**
  - **Hyperparameter β% :** The value of β determines the proportion of ensembles that are eliminated at each ensemble size S, thereby influencing the extent of pruning during the algorithm. Setting higher values of β leads to more significant pruning. In our experiments, we maintained a constant value of 10 for β.
  - **Focal diversity metrics :** We employ four diversity metrics, namely Cohen's kappa (CK), Generalized diversity (GD), Kohavi-Wolpert variance (KW), and Binary Disagreement (BD), to assess the diversity of a model ensemble. These metrics ensure that the models are dissimilar to one another and can generalize well to unseen data.

Cohen's kappa (CK) is a widely used evaluation metric for classification models. It assesses the agreement between predicted and actual classes, accounting for chance agreement. The scores range from -1 to 1, with higher scores indicating better agreement. Generalized diversity (GD) is a metric that gauges the diversity of predictions within an ensemble. It computes the average pairwise distance between the predictions made by different models in the ensemble. The value of GD ranges between 0 and 1, where 0 indicates identical models, while 1 indicates completely dissimilar models. Higher values of GD denote greater diversity, while lower values indicate greater similarity among the models. Kohavi-Wolpert variance (KW) is a non-parametric test that compares the variances of multiple groups. It measures the difference between the ranks of the data and the expected ranks if all groups were drawn from the same population. A high KW variance suggests that the groups have varying variances, whereas a small KW variance indicates that the groups have similar variances. Binary Disagreement (BD) is a metric that measures the disagreement between binary predictions made by different models. It ranges from 0 to 1, with higher values indicating more disagreement between the models.

4.5 Cross Validation Procedure

- **DropNet - Iterative Pruning**
  - **Training Runs :** To ensure the robustness and reliability of our results, we conducted a trial on Model C. At the trial, we evaluated the accuracy of the model against the proportion of filters surviving at various stages of pruning. To provide a comprehensive and reliable assessment, we calculated the average accuracy across the trial for each pruning metric and fraction value. To illustrate the accuracy performance of our models, we present a figure (Figure 3) showcasing the validation accuracy conducted on Model C, which has a filter sizes of 64x2 - 128x2, 128x2 - 128x2, 128x2 - 256x2. This figure demonstrates the accuracy

p=0.35	remain = 1.0		remain = 0.65		remain = 0.42		remain = 0.27	
	TA (%)	Loss	TA (%)	Loss	TA (%)	Loss	TA (%)	Loss
Min	75.2	1.71	72.0	1.74	65.3	1.81	**67.2**	1.79
Max	75.3	1.71	67.5	1.79	0.10	2.30	0.10	2.30
Random	74.9	1.71	73.3	1.72	63.2	1.83	0.10	2.30
Min Layer	72.2	1.73	69.5	1.76	71.7	1.74	**70.6**	1.75
Max Layer	74.2	1.72	74.1	1.72	0.10	2.30	0.10	2.30
Random Layer	75.6	1.70	74.0	1.72	72.3	1.74	0.10	2.30

Table 2. Test Accuracy and Loss occurred when Percentage of filters to drop when p=0.35 on **Model C: Conv128x2 - Conv256x2**

p=0.5	remain = 1.0		remain = 0.50		remain = 0.25	
	TA (%)	Loss	TA (%)	Loss	TA (%)	Loss
Min	71.8	1.74	71.4	1.74	**66.7**	1.79
Max	73.4	1.72	72.3	1.74	0.10	2.30
Random	74.9	1.71	72.6	1.73	0.10	2.30
Min Layer	74.3	1.72	70.3	1.76	0.10	2.30
Max Layer	74.6	1.71	70.6	1.75	0.10	2.30
Random Layer	74.4	1.72	70.0	1.77	0.10	2.30

Table 3. Test Accuracy and Loss occurred when Percentage of filters to drop, p=0.5 on **Model C: Conv128x2 - Conv256x2**

of the model at different stages of pruning, highlighting the effectiveness of our pruning techniques and their impact on model performance.

- **Percentage of filters to Drop :** We evaluated the performance of the model using different pruning fraction values and a variety of pruning metrics. We conducted experiments on the same Model C variation 128x2 - 256x2. Based on our evaluation, we observed that when the pruning fraction value was set to p=0.35, the global minimum and minimum layer pruning metrics demonstrated excellent performance. Specifically, these metrics performed well until the model had 25% of its nodes pruned, with only a slight drop of less than 10% in accuracy. In our analysis, we found that when the pruning fraction value was set to p=0.5, only the minimum pruning metric showed better performance, with only 25% of the nodes in the model. In contrast, the global maximum and maximum layer pruning metrics required at least 50% of the nodes to perform well, and the global random and random layer pruning metrics needed at least 35% of the nodes to perform at a similar level. We have summarized the results for p=0.35 and p=0.5 at different stages of the experiment in Table 2 and Table 3, respectively.
- **Number of filters :** During our experimentation, we explored different filter size variations for Model C, including 64x2 - 128x2, 128x2 - 128x2, and 128x2 - 256x2. We evaluated the

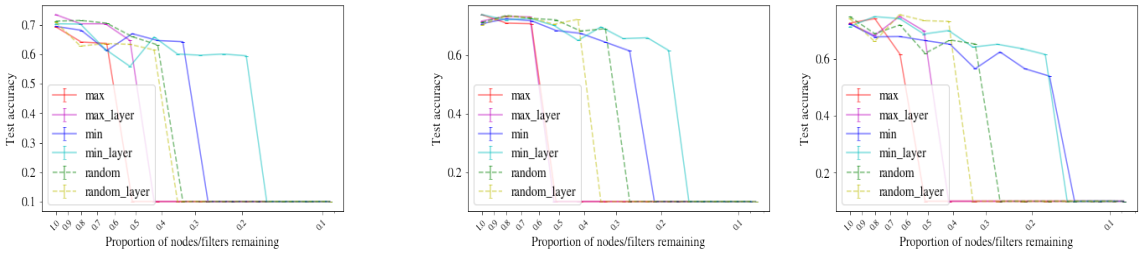

Fig. 4. Test accuracy vs. fraction of nodes remaining for various metrics in Model C: Conv64x2 - Conv128x2, Model C: Conv128x2 - Conv128x2 and Model C: Conv128x2 - Conv256x2 on CIFAR10

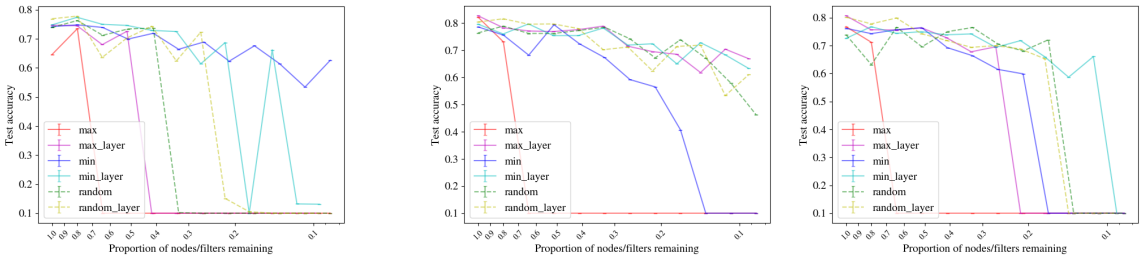

Fig. 5. Test accuracy vs. fraction of nodes remaining for various metrics in ResNet20, VGG16 and VGG19

performance of each variation and visualized the results in figure to compare and contrast their effectiveness.

	0% Pruned		20% Pruned		50% Pruned	
Best Ensemble Team	01234	023	01234	02	01234	0234
Ensemble Accuracy (%)	75.29	75.85	70.939	69.97	65.46	67.5
Accuracy improvement (%)	0	**0.56**	0	**-0.96**	0	**2.04**
Team Size	5	3	5	2	5	4
Cost	100%	**60%**	100%	**40%**	100%	**80%**

Table 4. Results of ensembled pruning on **Global Minimum** pruned models (0: VGG13, 1:VGG11, 2:VGG16 3:VGG19, 4:ResNet)

4.6 Results

- **DropNet - Iterative Pruning**
  - **Model C: Conv64x2 - Conv128x2 :** Figure 5 (Left Side) depicts a plot of test accuracy vs the proportion of filters remaining for various parameters. Minimum Layer performs the best followed by minimum, random, random layer, maximum layer and lastly maximum which consistently performed poor when the proportion of filters is 0.5.

	0% Pruned		20% Pruned		50% Pruned	
Best Ensemble Team	01234	01	01234	01	01234	124
Ensemble Accuracy (%)	75.54	75.68	74.13	76.34	13.55	13.15
Accuracy improvement (%)	0	**0.14**	0	**2.04**	0	**-0.4**
Team Size	5	2	5	2	5	3
Cost	100%	**40%**	100%	**40%**	100%	**60%**

Table 5. Results of ensembled pruning on **Global Maximum** pruned models (0: VGG13, 1:VGG11, 2:VGG16 3:VGG19, 4:ResNet)

- **Model C: Conv128x2 - Conv128x2 :** Figure 5 (Middle) depicts a plot of test accuracy vs proportion of nodes remaining for various parameters. Minimum Layer performs the best followed by minimum, random, random layer. Both maximum layer and maximum which consistently performed poor when the proportion of filters is 0.5.
- **Model C: Conv128x2 - Conv256x2 :** In Figure 5 (Right Side), we can see a plot of test accuracy against the proportion of nodes remaining for different pruning parameters. The results indicate that the Minimum Layer parameter performed the best, followed by the Random and Random Layer parameters. Both Minimum and Maximum Layer parameters performed well even after 80% pruning, with Maximum Layer consistently performing poorly when the proportion of filters is around 60%.
- **Results & Evaluate :** DropNet was successful in reducing the number of filters by at least 60% while maintaining acceptable model accuracy. This demonstrates the effectiveness of DropNet in significantly reducing network complexity without compromising performance. The 60% reduction was successfully replicated in the category of Model C type networks.
- **ResNet20 :** In Figure 5 (Left Side), a ResNet plot shows test accuracy against the fraction of remaining filters for different metrics. The Minimum metric performed good even after removing 90% of the filters. However, the Maximum metric consistently showed poor performance after 40% of the filters were removed.
- **VGG16 :** In Figure 5 (Middle), a VGG16 plot shows the test accuracy versus the fraction of remaining filters for various metrics. The minimum pruning metric outperformed the others, while most of the pruning metrics maintained good performance even after 80% of the filters were removed. However, the maximum pruning metric performed poorly after removing 30% of the filters.
- **VGG19 :** The plot in Figure 5 (Right Side) shows the test accuracy versus the fraction of filters remaining for different pruning metrics. The results are similar to VGG16, with all metrics performing well except for the maximum pruning metric, which deteriorated at 30% of filters removed. The other metrics maintained good performance until 80% of filters were removed.
- **Results & Evaluate :** DropNet was found to be effective in significantly reducing the complexity of ResNet, VGG and its variations while maintaining acceptable model accuracy, with the minimal or minimum layer metrics performing competitively. Using DropNet, the

number of filters can be reduced by 80% or more without significantly affecting model accuracy, demonstrating its efficacy in lowering network complexity. This result was successfully replicated in the category of bigger Convolutional Neural Networks.

	0% Pruned		20% Pruned		50% Pruned	
Best Ensemble Team	01234	0123	01234	0123	01234	12
Ensemble Accuracy (%)	76.34	77.04	74.43	74.78	62.4	65.14
Accuracy improvement (%)	0	**0.7**	0	**0.35**	0	**2.74**
Team Size	5	4	5	4	5	2
Cost	100%	**80%**	100%	**80%**	100%	**40%**

Table 6. Results of ensembled pruning on **Global Random** pruned models (0: VGG13, 1:VGG11, 2:VGG16 3:VGG19, 4:ResNet)

- **Hierarchical Ensemble Pruning** The experiments were conducted on models pruned at 0%, 20%, and 50% levels.
  - **Global Minimum :** The results of our experiments with the Minimum pruning metric are shown in Table 4. We observed that at 0% pruning, we achieved a notable increase in accuracy while using only 60% of the cost compared to the global ensemble. At 20% pruning, we achieved nearly the same accuracy as the global ensemble with only 40% of the cost. Furthermore, at 50% pruning with 20% less cost, we achieved almost 2% higher accuracy. These findings highlight the effectiveness of combining pruning techniques to attain high accuracy with reduced computational cost.
  - **Global Maximum :** Table 5 shows that with maximum pruning and no reduction in the number of nodes, we achieved slightly better accuracy with only 40% of the cost compared to the global ensemble. At 20% pruning, we achieved a significant improvement in accuracy with 60% less cost, and at 50% pruning, we achieved nearly the same accuracy as the global ensemble.
  - **Global Random :** Table 6 shows the results of global Random pruning. At 0% pruning, we achieved 0.7% more accuracy with a 20% reduction in cost. For the 20% pruned models, we obtained 0.35% more accuracy with just 80% of the cost. At 50% pruning, we achieved 2.75% more accuracy than the global ensemble with 60% less cost.
  - **Minimum Layer :** Table 7 summarizes the results of the HQ Ensemble pruning technique. With no pruning and 60% less cost, we achieved a 2.39% increase in accuracy compared to the global ensemble. For 20% pruned models, we obtained 0.22% more accuracy with 20% less cost, and for 50% pruned models, we achieved 0.5% more accuracy with 60% less cost. These results demonstrate the effectiveness of the HQ Ensemble technique in reducing computational cost while maintaining or even improving accuracy.
  - **Maximum Layer :** In Table 8 we can observe that with 0% maximum pruning, we achieved 0.14% more accuracy with 20% less cost, and with 20% pruning we got 2.04% more accuracy with 40% cost, and with 50% pruned models we achieved nearly the same accuracy as the Global ensemble with 40% cost.

- **Random Layer :** Table 9 shows the results of Random layer pruning. At 0% pruning, we achieved 1.6% more accuracy with 20% less cost. At 20% pruning, we achieved 4.32% more accuracy than the global ensemble with 40% less cost. With 50% pruning, we achieved 1.47% more accuracy with just 60% of the cost.
- The authors worked on 10 DNN models in the paper, but due to resource constraints, only 5 were studied in this work. However, the results show that a significant improvement in accuracy can be achieved with a reduction in cost ranging from 20

	0% Layer Pruned		20% Layer Pruned		50% Layer Pruned	
Best Ensemble Team	01234	01	01234	0123	01234	12
Ensemble Accuracy (%)	73.15	75.54	75.67	75.89	68.18	68.68
Accuracy improvement (%)	0	**2.39**	0	**0.22**	0	**0.5**
Team Size	5	2	5	4	5	2
Cost	100%	**40%**	100%	**80%**	100%	**40%**

Table 7. Results of ensembled pruning on **Minimum Layer** pruned models (0: VGG13, 1:VGG11, 2:VGG16 3:VGG19, 4:ResNet)

	0% Layer Pruned		20% Layer Pruned		50% Layer Pruned	
Best Ensemble Team	01234	0123	01234	02	01234	04
Ensemble Accuracy (%)	73.64	73.97	69.89	73.01	59.22	63.8
Accuracy improvement (%)	0	**0.14**	0	**2.04**	0	**-0.4**
Team Size	5	4	5	2	5	2
Cost	100%	**80%**	100%	**40%**	100%	**40%**

Table 8. Results of ensembled pruning on **Maximum Layer** pruned models (0: VGG13, 1:VGG11, 2:VGG16 3:VGG19, 4:ResNet)

	0% Level Pruned		20% Level Pruned		50% Level Pruned	
Best Ensemble Team	01234	0123	01234	012	01234	012
Ensemble Accuracy (%)	75.53	76.59	70.9	75.22	69.3	70.77
Accuracy improvement (%)	0	**1.6**	0	**4.32**	0	**1.47**
Team Size	5	4	5	3	5	3
Cost	100%	**80%**	100%	**60%**	100%	**60%**

Table 9. Results of ensembled pruning on **Random Layer** pruned models (0: VGG13, 1:VGG11, 2:VGG16 3:VGG19, 4:ResNet)

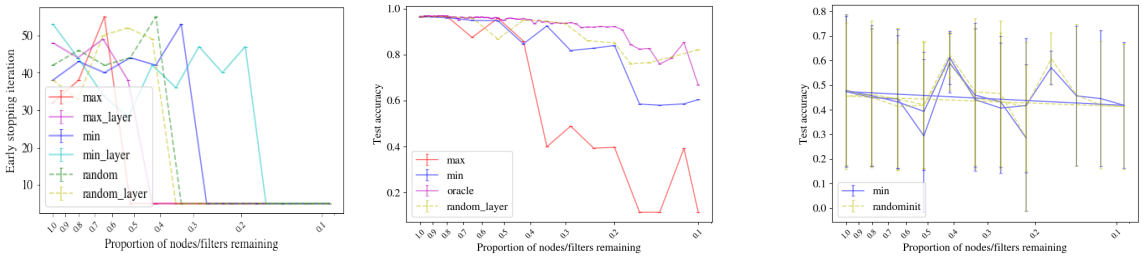

Fig. 6. Early Stopping and Test Accuracy on Oracle, Comparing Random Initialization with Minimum pruning metric on Model C - Conv64x2 - Conv128x2

4.7 Ablation Study and Empirical Analysis

- **Oracle Comparison :** The oracle is defined as the algorithm that greedily eliminates a filter from all remaining node/filters accessible at each iteration of DropNet in order to minimize the total training loss. To offer a fair comparison with the Oracle, nodes/filters are pruned one at a time when utilizing the different metrics. The results show that the lowest measure is competitive even against an oracle that reduces training loss. This demonstrates that the minimal metric is a competitive criteria for dropping nodes/filters. While DropNet runs in Linear time, the oracle operates in polynomial time since it must go over all potential node/filter selections at the conclusion of each iteration.
- **Compare to Random Initialization :** During iterative filter pruning for the Model C, Conv64x2 - Conv128x2, we evaluated the performance of two scenarios: retaining the initial weights and biases θ_0, and random initialization of the pruned architecture. DropNet was not significantly affected by random initialization up to a 70% drop of nodes/filters. The final pruned network architecture was the primary factor determining the performance, while the initial weights and biases had a negligible impact. This could be due to the high learning rate (0.1) used during retraining, which facilitated the network's adaptation to the new architecture.

The results were illustrated in Figure 6 demonstrating Test Accuracy on Oracle (Middle) and Test Accuracy on Compare Random Initialization (Right Side).

4.8 Discussion on Observations

- In our experiments, we were able to achieve improved accuracy of ensembles at different levels of pruning. Notably, we saw significant improvement in the Random Layer approach with 20% pruning, while using only 60% of the cost of the global ensemble. Additionally, we observed that in the Minimum Layer approach, the global ensemble with no pruning had lower accuracy compared to the ensemble model with 20% pruning. Similarly, In Global Maximum, we found that using the HQ algorithm to prune ensembles of models led to an improvement of 1% in accuracy compared to non-pruned ensembles, while still reducing computational costs by 60%. This shows that ensembles of pruned models can achieve better performance.
- The algorithm described in the paper utilizes ReLU activation function, which leads to the presence of inactive nodes that do not activate when their value drops to 0 or below. Inactive

nodes with low expected absolute value, i.e., those that fall below zero most of the time, have minimal impact on the model's classification performance, and hence can be removed.
- Nodes with low expected absolute values are less responsive to changes in inputs during backpropagation and therefore contribute less to the model's output. Removing these nodes has a minimal impact on classification accuracy compared to removing more important nodes, which contribute more to the model's output and are therefore more adaptive to learning from inputs. Further experiments should be conducted to test the applicability of DropNet on other types of neural networks, specifically Recurrent Neural Networks (RNNs) and Reinforcement Learning (RL) agents, in order to evaluate its effectiveness in improving their performance.
- Hierarchical Pruning employs diversity metrics to identify negative correlations among models, which in turn helps in selecting diverse models to improve accuracy.

5 WHO DID WHAT

This study explores two pruning techniques, DropNet: Iterative Pruning and Hierarchical Ensemble Pruning, out of many available options. The motivation behind selecting these two techniques is to experiment with their combination for our Milestone 3 and observe the results.

5.1 Sumanth Manduru - DropNet

- He has worked on the Abstract and Problem Definition in the report.
- He designed and developed the methodology of combining both DropNet and HQ Ensemble techniques. Analysing the results, discussions on Observations
- He implemented DropNet pruning for VGG11, VGG13, VGG16, VGG19 and Resnet models. He developed the models from scratch.
- He converted the code from Keras and Numpy Framework to PyTorch Framework.
- He executed all the models and obtained weights in the form of pickle files for the CIFAR10 dataset.
- Although we only showed our work for 0%, 20%, and 50% pruning he implemented all the models with pruning from 0% to 90%.
- He conducted empirical experiments on Random Initialization and Oracle.

5.2 Dileep Kumar - HQ Ensemble Pruning

- He has worked on the Introduction Section.
- He has worked in the methodology, experimental results, analysis, and coding aspects of HQ Ensemble Pruning.
- He converted all the weight files which we acquired from DropNet to .pt files which include Prediction Labels to Prediction Vectors.
- He worked on HQ-Ensemble pruning for maximum, minimum, and random for both global pruning and layer wise pruning.
- The results for HQ - Ensemble pruning were obtained by him

6 CONCLUSION

In Milestone 2 of our project, we conducted experiments on the MNIST dataset using the DropNet technique and on the CIFAR10 dataset using the Hierarchical Ensemble Pruning (HQ) algorithm. In Milestone 3, we extended our experiments on CIFAR10 to include larger networks such as ResNet, VGG, and their variations. We applied the Iterative Pruning algorithm to these networks to obtain pruned models that maintained high performance while using fewer parameters. We then created a pool of pruned models and applied the Hierarchical Ensemble Pruning algorithm to generate an ensemble of pruned models. Our goal in Milestone 3 is to study the behavior of the pruned models in an ensemble setting and assess their effectiveness and efficiency. Through these experiments, we gained a better understanding of the potential of pruning techniques to improve the performance and efficiency of deep neural networks.

6.1 Future Work

Due to the unavailability of orc hopper clusters, we were not able to conduct all the experiments that we had planned for our novelty approach. However, here are some of the experiments that we believe would provide strong evidence for the proposed method:

- As part of our proposed method, we could have experimented to remove different percentiles of filters (such as p = 0.35 and p = 0.5) from larger networks like ResNet, VGG19, and their variations, as the pruning fraction $p \in (0, 1]$. This would help us to study the behavior of these models with varying levels of pruning and determine the optimal level of pruning for each network architecture.
- Experimenting with different activation functions, such as SoftPlus or other variations of ReLU, is recommended. This is because ReLU as an activation function can result in inactive nodes that do not fire when the node value is 0 or below. By trying different activation functions, we could potentially enhance the performance of our pruned models and gain a better understanding of how activation functions affect the pruning process.
- Experimenting with larger pools of pruned models could have provided a broader scope for this innovative approach. Interpretation of these complex models before pruning and after pruning and understand their structure.

6.2 Source Papers

- **Conference Paper :** Tan, C. M. J., and Motani, M. Dropnet: Reducing neural network complexity via iterative pruning. In International Conference on Machine Learning, pp. 9356–9366. PMLR, 2020.
- **Conference Paper :** Yanzhao Wu and Ling Liu. 2021. Boosting Deep Ensemble Performance with Hierarchical Pruning. (Dec. 2021), 1433–1438

REFERENCES

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. *CoRR* abs/1512.03385 (2015). arXiv:1512.03385 http://arxiv.org/abs/1512.03385

[2] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. (2015).

[3] Chong Min John Tan and Mehul Motani. 2020. DropNet: Reducing Neural Network Complexity via Iterative Pruning. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, 9356–9366. https://proceedings.mlr.press/v119/tan20a.html

[4] Yanzhao Wu and Ling Liu. 2021. Boosting Deep Ensemble Performance with Hierarchical Pruning. (Dec. 2021), 1433–1438.