a. Methodology [20 pts]: Which 3+ pruning methods did you use? Describe each briefly.

1. Magnitude-based Pruning with Percentile-based Global Threshold Determination:
This method sets a global threshold based on the percentile of the absolute values of all weights in the network. Weights below this threshold are pruned (set to zero). After pruning, the original model for a given number of epochs with an optional reduced learning rate is retrained.
2. Magnitude-based Pruning with Standard Deviation-based Threshold Determination:
This method uses a layer-specific threshold determined by the standard deviation of the weights in each layer. The pruning and retraining process is similar to the first method, but the pruning criteria differ, focusing on the variability within each layer.
3. L1-Norm Based Filter Pruning (Implemented as Gradual Pruning with Fine-Tuning):
This method employs L1-norm as a criterion for pruning filters in convolutional and dense layers. The pruning rate increases stepwise over several iterations. In each step, filters with the smallest L1-norm values, falling below a dynamically adjusted percentile threshold, are pruned. Instead of removing the filter, we zero out these filters. After each pruning step, the model undergoes fine-tuning for a set number of epochs.
4. Channel pruning:
This method prunes channels by minimizing the feature map reconstruction error of the next layer and the optimization problem is solved by LASSO regression. As model architecture is not allowed to change, i.e., the input of channels actually could not be pruned, making weight reconstruction not meaningful in this context. Therefore, we tried to replicate the two stages but only performed the first stage and set the weights for selected channels to 0 for simplicity.

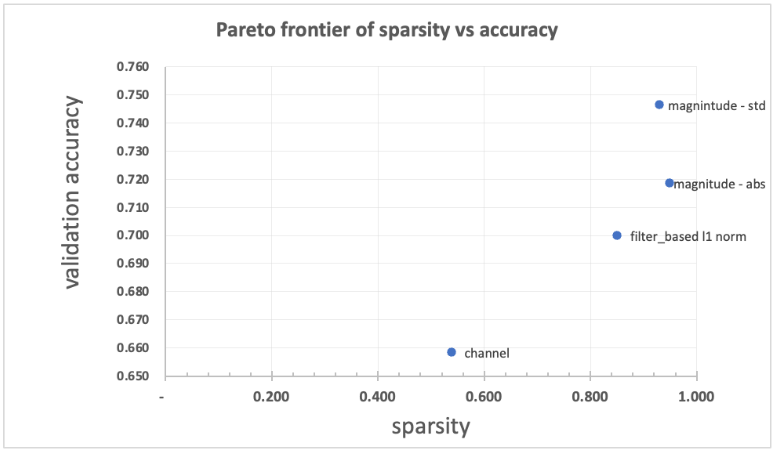
b. Comparison [60 pts]: Compare the performance of the pruning methods: For example, which method led to the best overall model score? Which was easiest to implement? Provide a plot exploring the methods in terms of the Pareto frontier of sparsity vs accuracy.

The application of magnitude-based pruning criteria emerged as the most effective approach, yielding the highest overall model performance, while also proving to be the most straightforward to implement. This method demonstrated success in pruning both Conv2D and dense layers, making it versatile across different layer types.

Notably, the dense layer alone contributed significantly to the model's parameter count, constituting nearly 90% of the total 592,933 parameters. With 524,800 parameters, the dense layer presented a substantial opportunity for reduction through pruning.

While techniques like channel pruning are traditionally associated with convolutional layers in CNNs, the significance of the dense layer in this pre-trained model underscores the importance of considering pruning strategies that address parameters in both Conv2D and dense layers. Channel pruning, tailored for convolutional layers, often involves intricate architectural adjustments, making it less applicable when model architecture remains fixed.

In this context, global pruning methods, such as magnitude-based weight pruning, prove to be particularly effective in achieving a favorable tradeoff between sparsity and accuracy. These methods offer the advantage of simplicity and versatility, especially when the model's architecture is not subject to modification. By focusing on the magnitudes of weights across all layers, including the dense layer, these global pruning techniques efficiently reduce the model's parameter count while preserving its overall performance.



c. Reflection [10 pts]: Did the results match what you expected? What did you learn about model pruning as part of this project?

For magnitude based pruning with standard deviation-based threshold determination, the method aligned with the theoretical expectations outlined in literature, successfully pruning less important weights while maintaining model accuracy. This confirms the effectiveness of using standard deviation as a criterion for identifying less crucial weights. Similar to the first method, magnitude-based pruning with percentile-based global threshold determination achieved a high score, indicating its efficacy in balancing model sparsity and performance.

For L1 norm based filter pruning, The performance was lower than expected. The deviation from the method described in the literature, specifically not removing filters but zeroing them out, impacted the effectiveness. Pruning neurons in the last dense layer for sparsity led to some loss in accuracy.

For channel pruning, the result is as expected. Without actually pruning channels and proper weight reconstruction, simply setting weights to 0 leads to the loss of information and decrease in validation accuracy. Additionally, the major part of hyperparameters are in the dense layer rather than the Conv2D layer. If not combined with magnitude-based weight pruning in the dense layer, this method does not work in this task.

Learning:

While pruning can result in a more efficient model, it may also have an impact on the model's accuracy. After pruning a model, we observe a drop in accuracy, as the pruned model has fewer parameters to capture the complexity of the underlying data. In that case, we might employ techniques to mitigate this loss in accuracy such as fine tuning, Iterative pruning and fine-tuning and applying a smaller learning rate to help the model to make smaller adjustments to the weights and adapt to the changes introduced by pruning.