Critical Testing and Evaluation of DNN Pruning Frameworks

Puja Trivedi, Skyler Granatir, Gurpreet Singh

***Abstract*—State of the art neural network pruning methods are able to substantially reduce the size of convolutional neural networks with low loss in accuracy [1]. Structured pruning methods also report significant speed-up in inference time [2]. These methods invariably make an independence assumption over the representations learned by the networks when designing saliency criteria. Yet, it is well known that neural networks are over-parameterized and learn redundant features [3]. Therefore, our work challenges the independence assumption and introduces a novel, principled regularizer that seeks to decorrelate representations while the network is training. Our results demonstrate that this regularizer leads to better performance when coupled with state-of-the-art pruning methods. We are able to reduce model size by more than 90%, increase accuracy by 1% and decrease inference computations by 90%. We compare our proposed method against 5 existing baselines, with respect to two datasets and four architectures. Furthermore, we perform adversarial attacks on the pruned networks to better understand the sparsity vs. robustness trade-off.**

I. Introduction

With the aim of achieving efficient deployment of Deep Neural Networks (DNNs) on embedded platforms, network pruning has been extensively studied recently. The goal of these methods is to assign importance to DNN parameters such that unimportant parameters may be removed, thereby minimizing resource requirements, without affecting network performance. Existing work invariably assumes that DNN features are independent of each other and that reducing parameter count or FLOPs translates to reducing inference time. This assumption is made to simplify the computing parameter saliency, as entanglement with other parameters is ignored, and eases computational burden.

Recent work challenges both these assumptions: Denil et al. show that a network's weights can be determined using less than 5% of its parameters, indicating high redundancy or dependency between parameters [4]. Chen et al. show that reduction in parameters or floating operations does not translate to direct reduction in inference time [5].

Motivated by these works, for this project, we make the following contributions:

- We show that ignoring neuron correlation harms performance when estimating neuron importance.
- We propose a novel, principled regularizer that seeks to orthogonalize filters, by doing so, we bring the trained network closer to assumptions made existing pruning methods.
- We demonstrate that our regularizer achieves state of the art performance across a variety of networks.
- We analyze reduction in FLOPs for existing pruning methods.
- We perform adversarial attacks on pruned networks to probe the relationships between sparsity, accuracy and robustness.

In section II, we summarize related work and briefly discuss the baselines we compare against. In section III, we demonstrate why the independence assumption is false and derive the orthogonality regularizer. In section IV, we outline our experiment framework and evaluation methodology. In section V, we discuss the performance of the regularizer against baselines. In section VI, we discuss the behavior of pruned networks under adversarial attacks. In section VII, we discuss the implications and extensions of this work.

II. Related Work

Pruning techniques use different saliency metrics to assign "importance" to parameters, where importance is a proxy how much the presence of a given parameter contributes to accuracy of the network. Parameters with the lowest importance scores are iteratively removed. Pruning can be *unstructured*, where individual weights are masked, or *structured*, where units (i.e. filters in a CNN) are removed. Structured pruning maintains dense matrices so existing hardware can efficiently perform computations [6]. Furthermore, techniques can be *local*, where pruning is conducted independently at each layer to achieve a desired sparsity, or *global*, where the lowest ranked parameters are pruned irrespective of the layers where they reside. Global pruning can be seen as a form of neural architecture search. We do not consider methods that are only applicable to fully connected layers as these methods are not competitive with modern architectures that are moving away from such layers. We briefly describe the global pruning baselines we implemented:

- Global Sparse Momentum [7] (GSM) : On the fly pruning and global compression. Assigns each parameters to different parts based on how much influence they have on objective function and then updates them using different (passive and active update) rules.
- NetSlim [8] NetSlim prunes full channels of a network to make a thin, compact network. An example of structured pruning, we attempt to remove channels of the network such that we also minimize the sparsity induced penalty. Upon pruning a given percentage of channels, the network is fine tuned to obtain similar accuracy.
- Data Driven Pruning [9] This is a global, unstructured method of pruning. This method observes which neurons have zero activation during each data sample. We prune the neurons that most commonly output zero. (Percentage of zero output).
- RateDistortion Theory [10] Used distortion theory to calculate lower bound on compression. Then improves the objective function by using quantization. It then minimizes $(w - w')I_w(w - w')$.
- Taylor First Order Molchanov [11] (TFO) Unstructured global method. This method calculates the squared difference between initial error and error retrieved when setting a particular parameter to zero. We prune the parameters that have the lowest difference. To efficiently compute this error, we do a First-Order Taylor expansion.
- Fisher Pruning [12] We consider the increase in error when we remove a certain parameter, and estimate this with a second order approximation that reduces beyond a Hessian. Greedily parameters one-by-one. Unstructured.
- Structured Bayesian Pruning [13] (SBP): Uses a Bayesian dropout-like layer to learn a mask over convolutional and fully connected layers. The SBP-layers introduce additional terms (with corresponding hyperparameters) to the loss.

Our method performs global pruning as these methods empirically have better performance [14].

III. Proposed Method

A. Correlations in CNNs

We see that the pruning methods discussed in the preceding assume that learned representations are independent and importance of a set of neurons is simply their sum. However, several works have demonstrated that the extracted representations within a layer are highly redundant and correlated ***refs correlation/MI papers***, and only *inter-layer* correlations are relatively small [15], [16], [17]. These works clearly show that the underlying independence assumptions in most pruning frameworks is not valid. To demonstrate the impact of redundancies and correlation of features on estimating the importance of a set of filters, we use the global importance framework based on change in loss as a function of neurons' removal. Molchanov et. al introduces this as an oracle for measuring neuron importance, and several recent methods also use these formulation to motivate their own metrics.

Consider the following method for defining the importance of a neuron using weight filter $\mathbf{w_i}$ as a function of its influence on the loss function:

$$I(h_i) = \left(L(\Theta) - L(\Theta, \mathbf{w_i} = 0)\right)^2 \tag{1}$$

where Θ denotes the set of parameters that the DNN is comprised of. Using Taylor's expansion about

$\mathbf{w_i} = \mathbf{0}$ and ignoring terms above the first order (see section II), we have:

$$I(\mathbf{w_i}) = \left(\mathbf{w_i}^T \nabla_{\mathbf{w_i}} L\right)^2 \tag{2}$$

It is worth mentioning that squaring the difference in loss to derive the above metric significantly improves the correlation of the estimated importance with respect to the ground truth [12] [11]. In essence, by squaring the difference, one ensures the final metric will focus on better matching the original network's output, while a mere difference would have focused on simply matching the performance of the pruned network to that of the original network [10].

Equation 2 denotes the impact of an individual neuron on the network's overall loss function. When one prunes a set of neurons, then the net importance of that set is approximated using the arithmetic sum of the individual importances of neurons comprising that set. For example, if we are pruning a set of neurons $[\mathbf{W}]$, comprising of N neurons with parameters $\mathbf{w_1}$, $\mathbf{w_2}$, $\ldots \mathbf{w_N}$, then the net importance of the set will be $I([\mathbf{W}]) = \sum_{i=1}^{N} \left(\mathbf{w_i}^T \nabla_{\mathbf{w_i}} L\right)^2$. However, if one were to use a first order Taylor's series expansion to calculate the importance of the total set, instead of an individual parameter, then the net importance will be formulated as:

$$\begin{aligned} I(\mathbf{w_i}) &= \left(\sum_{i=1}^{N} \mathbf{w_i}^T \nabla_{\mathbf{w_i}} L\right)^2 \\ &= \sum_{i=1}^{N} \left(\mathbf{w_i}^T \nabla_{\mathbf{w_i}} L\right)^2 \\ &+ 2\sum_{i=1}^{N} \sum_{j=i+1}^{N} \left(\mathbf{w_i}^T \nabla_{\mathbf{w_i}} L\right) \left(\mathbf{w_j}^T \nabla_{\mathbf{w_j}} L\right) \end{aligned} \tag{3}$$

Prior work has ignored the term $2\sum_{i=1}^{N} \sum_{j=i+1}^{N} \left(\mathbf{w_i}^T \nabla_{\mathbf{w_i}} L\right) \left(\mathbf{w_j}^T \nabla_{\mathbf{w_j}} L\right)$ by assuming that the neurons are independent and uncorrelated. If one does not make the uncorrelated assumption, then the problem of searching for a set of N neurons, amongst all possible N-neuron sets, such that the removal of that set results in least change in loss, will have a computational complexity increasing exponentially with N. For the following, we call this term the "correlation term".

B. Orthogonality Regularization

In the preceding section, we see i) that the independence between neurons assumption results in an incorrect estimate of neuron importance and ii) determining the exact metric is computationally prohibitive. However, if the correlation term were to not have an impact on the importance estimate of an N-neuron set, then one could very well approximate the set's importance as the arithmetic sum of the importances of the neurons contained within it. Motivated by this, we try to formulate an objective that minimizes the correlation term by pushing it towards zero. To this end, note that $\mathbf{w_i}^T \nabla_{\mathbf{w_i}} L$ is a scalar and can be reformulated as $\nabla_{\mathbf{w_i}} L^T \mathbf{w_i}$. Using this, the correlation term can be re-written as:

$$\begin{aligned} \left(\mathbf{w_i}^T \nabla_{\mathbf{w_i}} L\right) \left(\mathbf{w_j}^T \nabla_{\mathbf{w_j}} L\right) &= \left(\nabla_{\mathbf{w_i}} L^T \mathbf{w_i}\right) \left(\mathbf{w_j}^T \nabla_{\mathbf{w_j}} L\right) \\ &= \nabla_{\mathbf{w_i}} L^T \left(\mathbf{w_i} \mathbf{w_j}^T\right) \nabla_{\mathbf{w_j}} L \end{aligned} \tag{4}$$

This implies that if the outer product of each $(i,j)^{th}$ pair of filters goes to zero, then the correlation term is rendered zero and one gets an exact, unbiased estimate of the importance of a set of neurons. However, since only the filters belonging to the same layer will have the same dimensions, therefore we can only focus on the outer product of those filters. Nonetheless, as has been shown before, the correlation of activations produced by inter-layer neurons, suggesting the remainder can be ignored.

In order to minimize the outer product for filters at a layer, we formulate an easy to optimize regularization objective. Specifically, for a network with L layers, say the l^{th} layer has a weight matrix denoted by $\mathbf{W}(l) \in \mathbb{R}^{hwc_{in} \times c_{out}}$. In order to minimize the pairwise outer product, note that:

$$\begin{aligned} &\left\| \sum_i \sum_j \mathbf{w_i}(l) \mathbf{w_j}(l)^T \right\|_F \\ &= \left\| \left(\sum_i \mathbf{w_i}(l)\right) \left(\sum_i \mathbf{w_i}(l)^T\right) \right\|_F \\ &= \left\| \left(\sum_i \mathbf{w_i}(l)^T\right) \left(\sum_i \mathbf{w_i}(l)\right) \right\|_F \\ &= \left\| \sum_i \sum_j \mathbf{w_i}(l)^T \mathbf{w_j}(l) \right\|_F \end{aligned}$$

$$\leq \sum_i \sum_j \left\| \mathbf{w_i}(l)^T \mathbf{w_j}(l) \right\|_F$$
$$= \left\| \mathbf{W}(l)^T \mathbf{W}(l) \right\|_F$$

where we used the fact $\left\|\mathbf{A}\mathbf{A}^T\right\|_F = \left\|\mathbf{A}^T\mathbf{A}\right\|_F$ and the inequality followed from the triangle inequality for norms. Note that minimizing the upper bound implies minimizing the inner product between weight filters of a layer–i.e., we need to make the weight filters orthogonal. However, in the above derivation, we did not include the constraint $j \neq i$ for ease of understanding. We will therefore have to subtract the self-inner product (or the norm) of each filter. Overall, this leads to the following objective:

$$\mathcal{L}_{\text{ortho}} = \sum_{l=1}^{L} \left\| \mathbf{W}(l)^T \mathbf{W}(l) - \mathbf{I} \right\|_1 \quad (5)$$

This objective enforces unit norm on weight filters; however, any necessary scaling can be easily shifted to the batchnorm layers' scale parameter. Also note that we replaced the Frobenius norm with an $\ell 1$ norm[1] because optimization was significantly faster for it. Replacing the norms is justified because what our objective necessitates is orthogonality of filters, which can be achieved via $\ell 1$ norm as well.

The final learning objective will involve both the classification loss (i.e., cross entropy) and the orthogonality regularizer to preserve performance and induce the required structure in weight filters that leads to better pruning results:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cross entropy}} + \mathcal{L}_{\text{ortho}} \quad (6)$$

1) Importance with and without Orthogonality: To demonstrate the importance of the correlation term, we compute the correlation between the ground-truth and importance of our own method and [12] using AlexNet pretrained on CIFAR100. To compute the ground truth, we sample 500 neurons across the entire network, and measure the change in loss. While increasing the number of samplings improves the correlation for both methods, we see that our method is better correlated with the ground truth. See 1

2) Relation with Dynamical Isometry: Saxe et al. [18] showed that initialization of linear neural networks with orthogonal filters ensures that the norm of an error vector is preserved. This desirable property helps improve training dynamics by allowing better error signal propagation–i.e., avoidance of vanishing or exploding gradients. Recently, Lee at al. showed that if one prunes neurons with low sensitivity, where a neuron's sensitivity is defined similarly to Equation 2, then one can prune neurons at initialization itself and still achieve similar performance as pruning the pre-trained network [19]. However, they show that not all initialization strategies allow this–in fact, if the weight filters of the layers of a network are orthogonal, then the network is more amenable to pruning at initialization because the post-pruning filters are bound to be closer to being orthogonal if orthogonal initialization was used. They term this property "Approximate Dynamical Isometry". Interestingly, we have derived the same conclusion in this paper from a completely different perspective, showing that pruning networks with orthogonal filters at any stage of the training will result in better estimation of more important neurons, which as a repercussion will ensure better training dynamics and generalization. Also note that the techniques proposed in SNIP are single-shot–i.e., the desired number of neurons are removed in a single iteration. In contrast, we prune network iteratively. Single-shot pruning based on current round's parameter values can result in aggressive pruning of overly parameterized layers (e.g., fully connected layers) and can even result in complete deletion of a layer, which significantly hurts performance. Further, since we are using only an approximation to the importance of a neuron, aggressive pruning with incorrect estimates in a single-shot manner can result in removal of important neurons.

IV. Experimental Set-up

To ensure rigorous evaluation, we test our proposed framework, using the baselines mentioned in section II, across 2 datasets (CIFAR100, and Tiny-ImageNET) [20] and 4 baseline architectures (VGG-like, AlexNet, MobileNetV1 and ResNet-34). We select these architectures because i) they

[1]The $\ell 1$ norm is defined for vectors, not matrices. To be exact, $\|.\|_1$ should be defined as the $\ell 1$ norm of the vectorized version of a matrix.

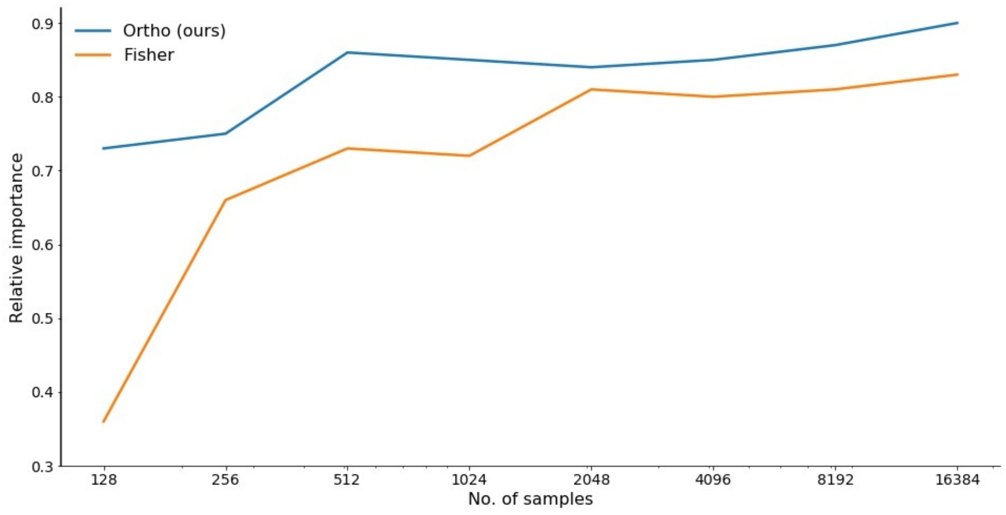


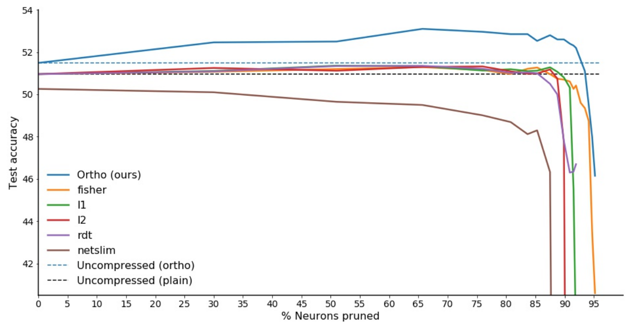
Fig. 1: *RMSE of Estimated vs. Ground Truth Importance* We train VGG-13 on CIFAR-100 with and without the orthogonality regularizer. We then compute the RMSE between ground-truth and estimated importance. We find that the orthogonalized network has less error on 7 out of 10 layers tested. The difference between the correlated and decorrelated RMSE is small on the layers where the correlated networks performs better. We investigate further for our final report.

are standard choices in pruning literature, and ii) they capture a range of difficulty, For example, AlexNet is amenable to high levels of pruning due to its several, large fully-connected layers. ResNet is more difficult to prune because it contains skip-connections. MobileNetV1 is also difficult to prune because it was explicitly designed to be a small network [21]. VGG-like networks, which do not contain any fully connected layers, are used to evaluate method performance on convolutional-reliant networks.

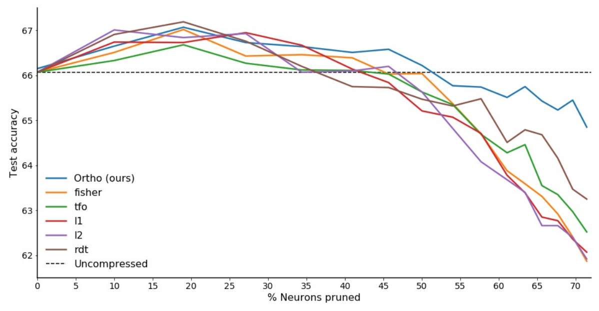
We use global pruning, instead of local pruning, as the former has been empirically shown to achieve better results [14]. To ensure fair comparisons, we run each of the baselines ourselves across standardized architectures. Whenever possible, we re-implement baselines, and ensure our code produces comparable results to the paper. Otherwise, we use available code, but still re-run with our own architectures. We use an Adam Optimizer ($\beta_1 = 0.9, \beta_2 = 0.999$), and learning rate plateau scheduler (drop learning rate after 5 epochs of no improvement), as well as a weight decay regularizer, ($\lambda = 0.0005$) running baselines. The weight decay regularizer is not used with the orthogonality regularizer as there are conflicting objectives. All networks are pruned iteratively: after pruning a fixed number of neurons, the networks is trained until convergence before the next round of pruning. After pruning, we report the compression-vs.accuracy tradeoff, and percentage decrease in GFLOPs.

V. Pruning Results

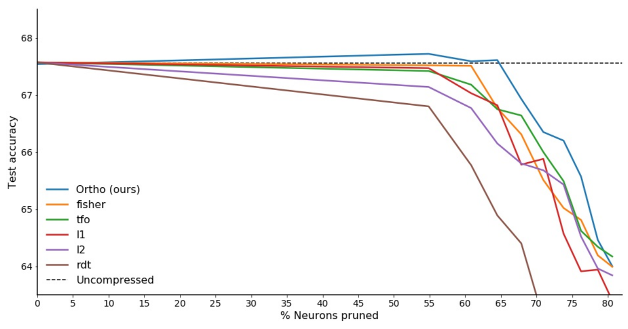
In addition to the five methods (fisher, L1, L2, rate distortion theory, and Taylor First Order) shown in figures 2 and 3, we also tested three additional baselines: Structured Bayesian Pruning [13], DataDriven Pruning [9] and GSM [7]. Given our limited computational resources, we did not conduct our full suite of experiments with these baselines because of their significantly poorer performance relative to the other baselines. SBP was unable to achieve high level of sparsity ($< 40\%$ of neurons removed), while maintaining a tolerable accuracy on AlexNet ($> 3\%$ accuracy drop). While its performance was slightly better on VGG, it required a significant amount of unintuitive hyperparameter



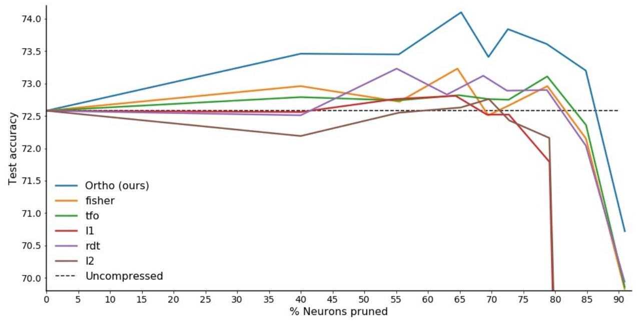
(a) AlexNet – CIFAR100



(b) VGG – CIFAR100



(c) MobileNet – CIFAR100



(d) ResNet34 – CIFAR100

Fig. 2: Sparsity vs. Accuracy on networks trained with CIFAR100.

tuning. DataDriven and GSM faced similar issues, where the accuracy drop relative to performance was clearly inferior to other methods.

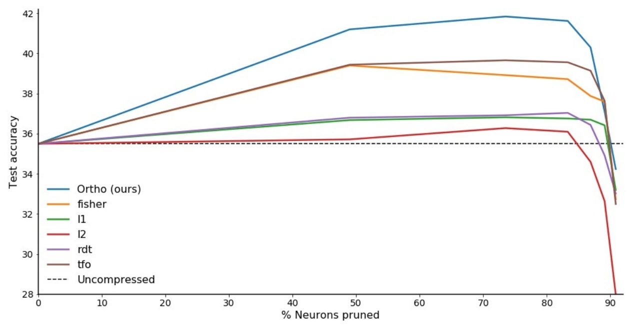
Against the five methods tested, we find that our method performs dominantly: achieving *both* higher sparsity and higher accuracy than the others. For AlexNet-CIFAR100 and ResNet-CIFAR100, the inclusion of the Ortho-Regularizer leads to considerably higher accuracy($> 1\%$) than the pretrained model. There are slight increases with MobileNet-CIFAR100 and VGG-CIFAR100. Using global iterative pruning, we are able to remove more than 95% of neurons in AlexNet-CIFAR100 with $< 10\%$ loss in accuracy. While [12] is able to comparably prune, it has $> 20\%$ decrease in accuracy. With MobileNet, which is difficult to prune because of its reduced size, and ResNet, which is difficult to prune because of dependences induced by skip connections, we are able to prune $> 80\%$ and $> 90\%$ of neurons, respectively, with a tolerable decrease in accuracy.

With AlexNet-TinyImageNet and ResNet34-ImageNet, pruning, regardless of method, leads to increased accuracy. Ortho-Regularizer is able to maintain higher accuracy at higher sparsity levels for networks trained with TinyImageNet as well.

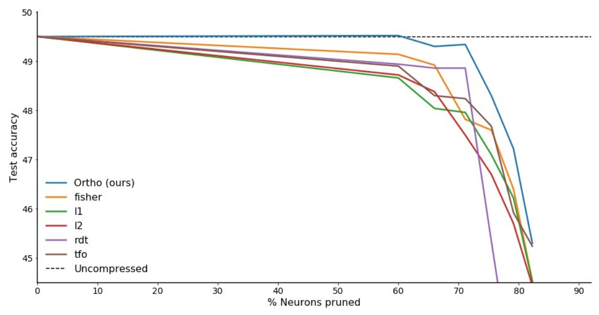
Our project was motivated by deployment on embedded systems, as previous pruning methods are not optimized to reduce inference times. We use total floating point operations to measure how much faster a pruned network would be on an embedded device. In 4, we see that Ortho-pruning is able to achieve high accuracy with high reductions in the number of Flops. This translates to signficant inference speed-up on embedded devices.

VI. Adversarial Attack Results

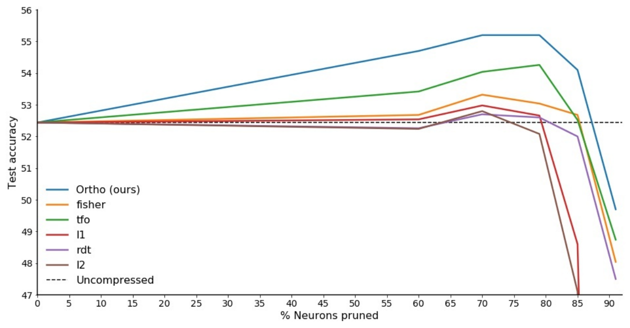
Understanding the trade-off between robustness to adversarial attacks and sparsity is an active research area. To evaluate robustness, we use the checkpoints generated by each method at the same level of sparsity. The corresponding accuracy is reported in



(a) AlexNet – TinyImageNet



(b) VGG – TinyImageNet



(c) ResNet34 – TinyImageNet

Fig. 3: Sparsity vs. Accuracy on networks trained with TinyImageNet.

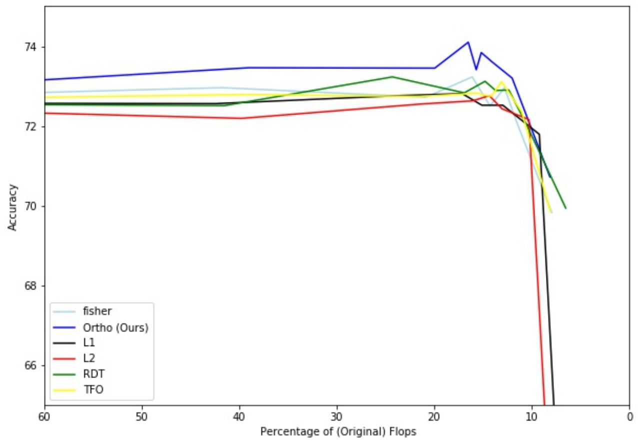
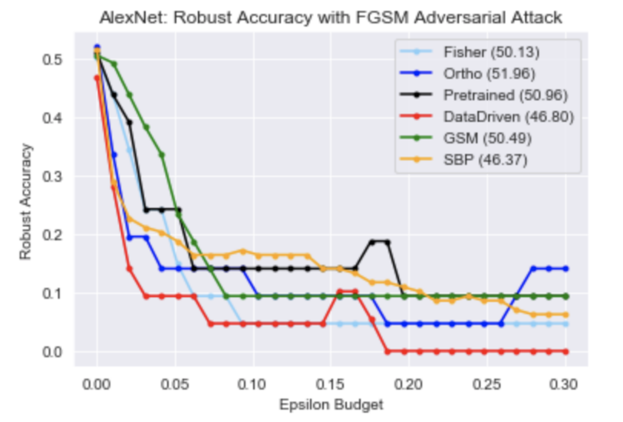
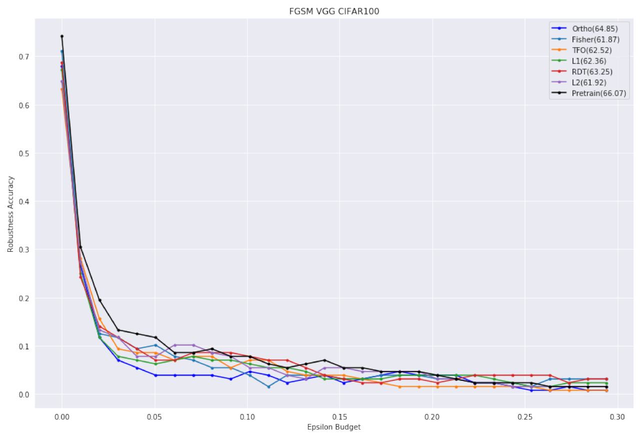


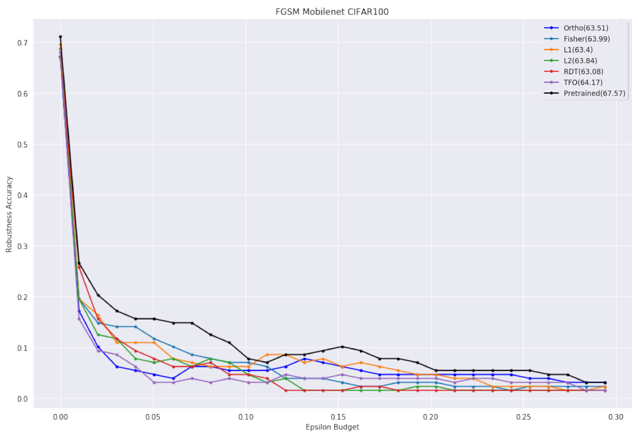
Fig. 4: *Flops vs Accuracy for ResNet* Flops taken as a percentage of unpruned ResNet. Ortho-pruning maintains higher accuracy at high levels of Flops reductions relative to other methods. For brevity, we only present our results on ResNet. Details for other networks can be found in our code repository.



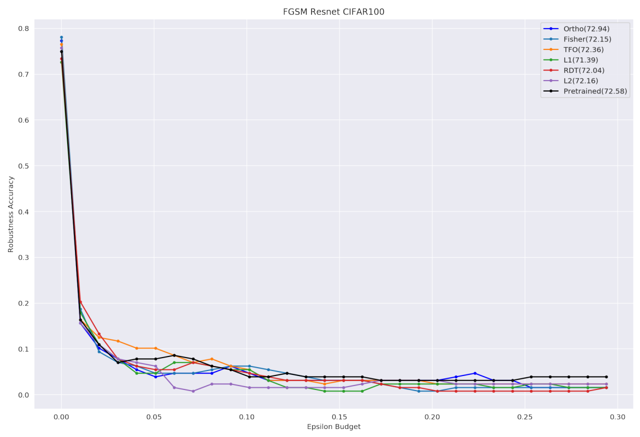
(a) AlexNet – CIFAR100 – FGSM



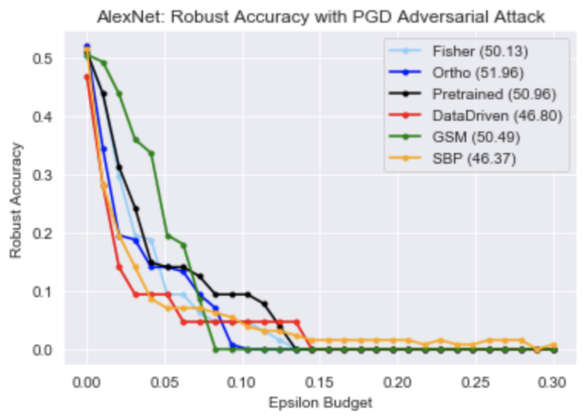
(b) VGG – CIFAR100 – FGSM



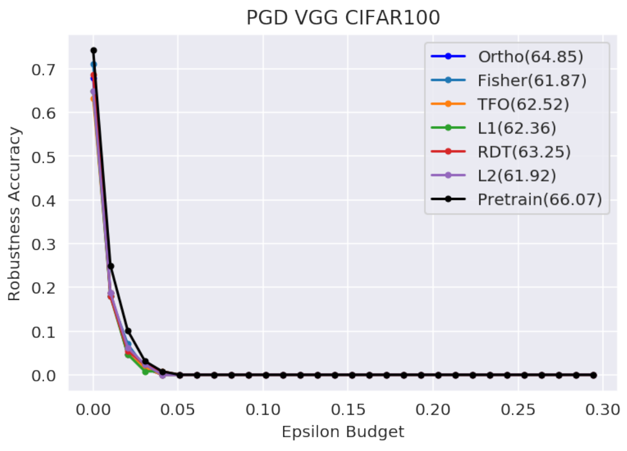
(c) MobileNet – CIFAR100 – FGSM



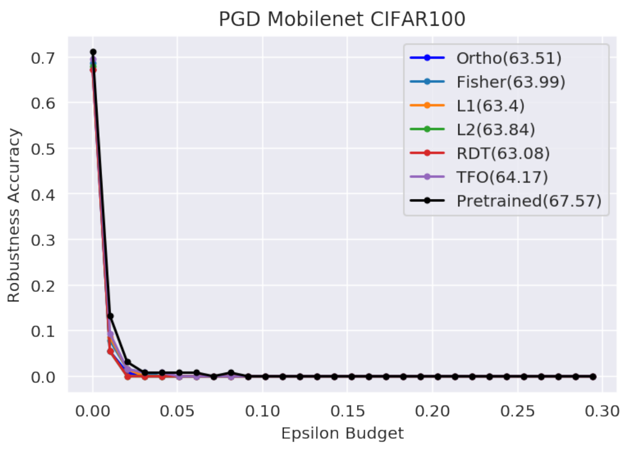
(d) ResNet34 – CIFAR100 – FGSM



(e) AlexNet – CIFAR100 – PGD



(f) VGG – CIFAR100 – PGD



(g) MobileNet – CIFAR100 – PGD

Fig. 5: Adversarial Attacks with networks trained on CIFAR100

the legend. Against FGSM attacks (figure 6), Ortho-regularizer performs on par to existing pruning techniques in AlexNet and ResNet34, but has inferior performance on VGG and MobileNet. Against Projected Gradient attacks, all pruning methods perform similarly. The pretrained network does not seem to perform significantly better under PGD attacks, but the difference is more prounced against FGSM attacks. Recent works suggest that the redundant or unimportant neurons in overparameterized networks help increase robustness. These experiments support this claim.

VII. Conclusion

We propose a novel, principled orthogonality regularizer. We show that the regularizer improves estimates of neuron importance, leading to higher sparsity at higher accuracy agaisnt 6 strong baselines. We also measure FLOPs across pruned networks to evaluate inference time reductions. Additionally, we conduct adversarial attacks on our pruned networks. We find that there is a trade-off between sparsity and robustness even at high levels of accuracy. For future work, we would like to test on an embedded device, such as the Tegra TX 2, as well as explore the connections between the lottery ticket hypothesis and the regularizer.

VIII. Author Contributions

Ekdeep Singh Lubana proposed the orthogonalization regularizer (Section III), and the testing framework. He has since dropped the class, though continues to work with us on the project. All authors contributed equally to the project.

References

[1] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding," *CoRR*, vol. abs/1510.00149, 2015.

[2] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, Jan Kautz, "Pruning Convolutional Neural Networks for Resource Efficient Inference," *In Proc. Int. Conf. on Learning Representations*, 2017.

[3] S. Arora, N. Cohen, and E. Hazan, "On the optimization of deep networks: Implicit acceleration by overparameterization," *CoRR*, vol. abs/1802.06509, 2018. [Online]. Available: http://arxiv.org/abs/1802.06509

[4] M. Denil, B. Shakibi, L. Dinh, M. A. Ranzato, and N. de Freitas, "Predicting parameters in deep learning," in *In Proc. Neural Information Processing Systems*, 2013.

[5] Yu-Hsin Chen, Tien-Ju Yang, Joel Emer, Vivienne Sze, "Understanding the Limitations of Existing Energy-Efficient Design Approaches for Deep Neural Networks," *In Proc. Conf. on Machine Learning and Systems*, 2019.

[6] S. Anwar, K. Hwang, and W. Sung, "Structured pruning of deep convolutional neural networks," *CoRR*, vol. abs/1512.08571, 2015. [Online]. Available: http://arxiv.org/abs/1512.08571

[7] X. Ding, g. ding, X. Zhou, Y. Guo, J. Han, and J. Liu, "Global sparse momentum sgd for pruning very deep neural networks," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 6382–6394. [Online]. Available: http://papers.nips.cc/paper/8867-global-sparse-momentum-sgd-for-pruning-very-deep-neural-networ pdf

[8] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," *CoRR*, vol. abs/1708.06519, 2017.

[9] H. Hu, R. Peng, Y. Tai, and C. Tang, "Network trimming: A data-driven neuron pruning approach towards efficient deep architectures," *CoRR*, vol. abs/1607.03250, 2016. [Online]. Available: http://arxiv.org/abs/1607.03250

[10] W. Gao, Y.-H. Liu, C. Wang, and S. Oh, "Rate distortion for model Compression:From theory to practice," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. Long Beach, California, USA: PMLR, 09–15 Jun 2019, pp. 2102–2111. [Online]. Available: http://proceedings.mlr.press/v97/gao19c.html

[11] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, Jan Kautz, "Importance Estimation for Neural Network Pruning," *In Proc. Int. Conf. on Computer Vision and Pattern Recognition*, 2019.

[12] Lucas Theis, Iryna Korshunova, Alykhan Tejani, and Ferenc Huszár, "Faster Gaze Prediction With Dense Networks and Fisher Pruning," *In Proc. European Conf. on Computer Vision*, 2018.

[13] K. Neklyudov, D. Molchanov, A. Ashukha, and D. Vetrov, "Structured bayesian pruning via log-normal multiplicative noise," 2017.

[14] D. Blalock, J. J. G. Ortiz, J. Frankle, and J. Guttag, "What is the state of neural network pruning?" 2020.

[15] M. Raghu, J. Gilmer, J. Yosinski, and J. Sohl-Dickstein, "Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability," 2017.

[16] A. S. Morcos, M. Raghu, and S. Bengio, "Insights on representational similarity in neural networks with canonical correlation," 2018.

[17] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton, "Similarity of neural network representations revisited," 2019.

[18] A. M. Saxe, J. L. McClelland, and S. Ganguli, "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks," 2013.

[19] N. Lee, T. Ajanthan, S. Gould, and P. H. S. Torr, "A signal propagation perspective for pruning neural networks at initialization," in *International Conference*

on Learning Representations, 2020. [Online]. Available: https://openreview.net/forum?id=HJeTo2VFwH

[20] A. Krizhevsky, V. Nair, and G. Hinton, “Cifar-10 (canadian institute for advanced research).” [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[21] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” *CoRR*, vol. abs/1704.04861, 2017. [Online]. Available: http://arxiv.org/abs/1704.04861