Emergence of Implicit Filter Sparsity in Convolutional Neural Networks

Dushyant Mehta [1 2] Kwang In Kim [3] Christian Theobalt [1 2]

Abstract

We show implicit *filter level* sparsity manifests in convolutional neural networks (CNNs) which employ Batch Normalization and ReLU activation, and are trained using adaptive gradient descent techniques with L2 regularization or weight decay. Through an extensive empirical study (Mehta et al., 2019) we hypothesize the mechanism behind the sparsification process. We find that the interplay of various phenomena influences the strength of L2 and weight decay regularizers, leading the supposedly non sparsity inducing regularizers to induce filter sparsity. In this workshop article we summarize some of our key findings and experiments, and present additional results on modern network architectures such as ResNet-50.

1. Introduction

In this article we discuss the findings from (Mehta et al., 2019) regarding filter level sparsity which emerges in certain types of feedforward convolutional neural networks. Filter refers to the weights and the nonlinearity associated with a particular feature, acting together as a unit. We use filter and feature interchangeably throughout the document. We particularly focus on presenting evidence for the implicit sparsity, our experimentally backed hypotheses regarding the cause of the sparsity, and discuss the possible role such implicit sparsification plays in the adaptive vs vanilla (m)SGD generalization debate. For implications on neural network speed up, refer to the original paper (Mehta et al., 2019).

In networks which employ Batch Normalization and ReLU activation, after training, certain filters are observed to not activate for any input. Importantly, the sparsity emerges in the presence of regularizers such as L2 and weight decay (WD) which are in general understood to be non sparsity inducing, and the sparsity vanishes when regularization is removed. We experimentally observe the following:

- The sparsity is much higher when using adaptive flavors of SGD vs. (m)SGD. The sparsity exists even with leaky ReLU.
- Adaptive methods see higher sparsity with L2 regularization than with WD. No sparsity emerges in the absence of regularization.
- In addition to the regularizers, the extent of the emergent sparsity is also influenced by hyperparameters seemingly unrelated to regularization. The sparsity decreases with increasing mini-batch size, decreasing network size and increasing task difficulty.
- The primary hypothesis that we put forward is that selective features[1] see a disproportionately higher amount of regularization than non-selective ones. This consistently explains how unrelated parameters such as mini-batch size, network size, and task difficulty indirectly impact sparsity by affecting feature selectivity.
- A secondary hypothesis to explain the higher sparsity observed with adaptive methods is that Adam (and possibly other) adaptive approaches learn more selective features. Though threre is evidence of highly selective features with Adam, this requires further study.
- Synthetic experiments show that the interaction of L2 regularizer with the update equation in adaptive methods causes stronger regularization than WD. This can explain the discrepancy in sparsity between L2 and WD.

Quantifying Feature Sparsity: Feature sparsity can be measured by per-feature activation and by per-feature scale. For sparsity by activation, the absolute activations for each feature are max pooled over the entire feature plane. If the value is less than 10^{-12} over the entire *training* corpus, the feature is inactive. For sparsity by scale, we consider the scale γ of the learned affine transform in the Batch Norm layer. We consider a feature inactive if $|\gamma|$ for the feature is less than 10^{-3}. Explicitly zeroing the features thus marked inactive does not affect the test error, which ensures the validity of our chosen thresholds. The thresholds chosen are purposefully conservative, and comparable levels of sparsity are observed for a higher feature activation threshold of 10^{-4}, and a higher $|\gamma|$ threshold of 10^{-2}.

[1] Max Planck Institute For Informatics, Saarbrücken, Germany [2] Saarland Informatics Campus, Germany [3] Ulsan National Institute of Science and Technology, South Korea.

Identifying and Understanding Deep Learning Phenomena Workshop, 36th International Conference on Machine Learning, 2019.

[1] Feature selectivity is the fraction of training exemplars for which a feature produces max activation less than some threshold.

Table 1. Convolutional filter sparsity in *BasicNet* trained on CIFAR10/100 for different combinations of regularization and gradient descent methods. Shown are the % of non-useful / inactive convolution filters, as measured by activation over training corpus (max act. $< 10^{-12}$) and by the learned BatchNorm scale ($|\gamma| < 10^{-03}$), averaged over 3 runs. The lowest test error per optimizer is highlighted, and sparsity (green) or lack of sparsity (red) for the best and near best configurations indicated via text color. L2: L2 regularization, WD: Weight decay (adjusted with the same scaling schedule as the learning rate schedule).

		CIFAR10			CIFAR100		
		% Sparsity		Test	% Sparsity		Test
	L2	by Act	by γ	Error	by Act	by γ	Error
SGD	1e-03	27	27	21.8	23	23	47.1
	1e-04	0	0	11.8	0	0	37.4
	1e-05	0	0	10.5	0	0	39.0
	0	0	0	11.3	0	0	40.1
Adam	2e-03	88	86	14.7	82	81	42.7
	1e-04	71	70	10.5	47	47	36.6
	1e-05	48	48	10.7	5	5	40.6
	0	3	0	11.0	0	0	40.3
Adadelta	5e-04	82	82	13.6	61	61	39.1
	2e-04	40	40	11.3	3	3	35.4
	1e-04	1	1	10.2	1	1	35.9
Adagrad	2e-02	75	75	11.3	88	88	63.3
	1e-02	65	65	11.2	59	59	37.2
	5e-03	56	56	11.3	24	25	35.9
AMSGrad	1e-02	93	93	20.9	95	95	71.9
	1e-04	51	47	9.9	20	13	35.6
	1e-06	0	0	11.2	0	0	40.2
Adamax	1e-02	75	90	16.4	74	87	51.8
	1e-04	49	50	10.1	10	10	39.3
	1e-06	4	4	11.3	0	0	39.8
RMSProp	1e-02	95	95	26.9	97	97	78.6
	1e-04	72	72	10.4	48	48	36.3
	1e-06	29	29	10.9	0	0	40.6
		CIFAR10			CIFAR100		
		% Sparsity		Test	% Sparsity		Test
	WD	by Act	by γ	Error	by Act	by γ	Error
SGD	1e-03	27	27	21.6	23	23	47.6
	2e-04	0	0	13.3	0	0	39.4
	1e-04	0	0	12.4	0	0	37.7
Adam	5e-04	81	81	18.1	59	59	43.3
	2e-04	60	60	13.4	16	16	37.3
	1e-04	40	40	11.2	3	3	36.2

Table 2. Convolutional filter sparsity for BasicNet with leaky ReLU with different negative slopes, trained on CIFAR-100 with Adam and L2 regularization (1e-4). Average of 3 runs.

Neg. Slope	Train Loss	Val Loss	Val Err.	% Spar. by γ
0.00	0.10	1.98	36.6	46
0.01	0.10	1.99	36.8	41
0.10	0.14	2.01	37.2	43

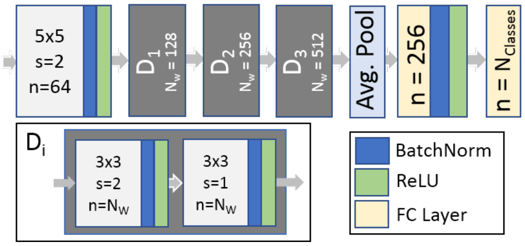

Figure 1. **BasicNet**: Structure of the basic convolution network studied in this paper. We refer to the convolution layers as C1-7.

2. Observing Filter Sparsity

Preliminary Experiments: We use a 7-layer convolutional network with 2 fully connected layers as shown in Figure 1. We refer to this network as *BasicNet* in the rest of the document. For the basic experiments on CIFAR-10/100, we use a variety of gradient descent approaches, a mini-batch size of 40, with a method specific base learning rate for 250 epochs which is scaled down by 10 for an additional 75 epochs. The base learning rates and other hyperparameters are as follows: Adam (1e-3, β_1=0.9, β_2=0.99, ϵ=1e-8), Adadelta (1.0, ρ=0.9, ϵ=1e-6), SGD (0.1, mom.=0.9), Adagrad (1e-2), AMSGrad (1e-3), AdaMax (2e-3), RMSProp (1e-3). We study the effect of varying the amount and type of regularization[2] on the extent of sparsity and test error in Table 1. It shows significant convolutional filter sparsity emerges with adaptive gradient descent methods when combined with L2 regularization. The extent of sparsity is reduced when using Weight Decay instead, and absent entirely in the case of SGD with moderate levels of regularization. Table 2 shows that using leaky ReLU does not prevent sparsification.

Sparsity Manifests Across Network Architectures and Datasets: The emergence of sparsity is not an isolated phenomenon specifc to CIFAR-10/100 and *BasicNet*. We show in tables 3, 4, and 5 that sparsity manifests in VGG-11/16 ((Simonyan & Zisserman, 2014)), and ResNet-50 ((He et al., 2016)) on ImageNet and Tiny-ImageNet. ResNet-50 shows a significantly higher overall filter sparsity than non-residual VGG networks.

Sparsity Increases with Decreasing Mini-Batch Size: We see in Tables 6, 7, 3, 4, and 5 that decreasing the mini-batch size (while maintaining the same number of iterations) leads to increased sparsity across network architectures and datasets.

3. Explaining Filter Sparsity

Feature Selectivity Hypothesis: From Figure 2 the differences between the nature of features learned by Adam and SGD become clearer. For zero mean, unit variance Batch-

[2] Note that L2 regularization and weight decay are distinct. See (Loshchilov & Hutter, 2017) for a detailed discussion.

Table 3. Sparsity by γ on VGG-16, trained on TinyImageNet, and on ImageNet. Also shown are the pre- and post-pruning top-1/top-5 single crop validation errors. Pruning using $|\gamma| < 10^{-3}$ criteria.

	# Conv	Pre-pruning		Post-pruning	
TinyImageNet	Feat. Pruned	top1	top5	top1	top5
L2: 1e-4, B: 20	3016 (71%)	45.1	21.4	45.1	21.4
L2: 1e-4, B: 40	2571 (61%)	46.7	24.4	46.7	24.4
ImageNet					
L2: 1e-4, B: 40	292	29.93	10.41	29.91	10.41

Table 4. Effect of different mini-batch sizes on sparsity (by γ) in VGG-11, trained on ImageNet. Same network structure employed as (Liu et al., 2017). * indicates finetuning after pruning

	# Conv	Pre-pruning		Post-pruning	
	Feat. Pruned	top1	top5	top1	top5
Adam, L2: 1e-4, B: 90	71	30.50	10.65	30.47	10.64
Adam, L2: 1e-4, B: 60	140	31.76	11.53	31.73	11.51
(Liu et al., 2017)	85	29.16		31.38*	-

Table 5. Convolutional filter sparsity for different levels of ResNet-50 on ImageNet, with different batch sizes, using Adam and L2 regularization (1e-4).

Batch	Train	Test	Top 1	Top 5	% Sparsity by γ					
Size	Loss	Loss	Val Err.	Val Err.	conv1	res2	res3	res4	res5	Total
32	1.3	1.1	27.7	9.2	0	0	1	17	46	26
64	1.0	1.0	25.2	7.7	0	0	1	3	42	19

Table 6. BasicNet sparsity variation on CIFAR10/100 trained with Adam and L2 regularization.

		CIFAR 10				CIFAR 100			
	Batch	Train	Test	Test	%Spar.	Train	Test	Test	%Spar.
	Size	Loss	Loss	Err	by γ	Loss	Loss	Err	by γ
L2: 1e-4	20	0.17	0.36	11.1	70	0.69	1.39	35.2	57
	40	0.06	0.43	10.5	70	0.10	1.98	36.6	46
	80	0.02	0.50	10.1	66	0.02	2.21	41.1	35
	160	0.01	0.55	10.6	61	0.01	2.32	44.3	29

Table 7. Convolutional filter sparsity for BasicNet trained on Tiny-ImageNet, with different mini-batch sizes.

Batch	Train	Val	Top 1	Top 5	% Spar.
Size	Loss	Loss	Val Err.	Val Err.	by γ
20	1.05	2.13	47.7	22.8	63
40	0.16	2.96	48.4	24.7	48
120	0.01	2.48	48.8	27.4	26

Norm outputs $\{\hat{x}_i\}_{i=1}^{N}$ of a particular convolutional kernel, where N is the size of the training corpus, due to the use of ReLU, a gradient is only seen for those datapoints for which $\hat{x}_i > -\beta/\gamma$. Both SGD and Adam (L2: 1e-5) learn positive γs for layer C6, however βs are negative for Adam, while for SGD some of the biases are positive. This implies that all features learned for Adam (L2: 1e-5) in this layer activate for $\leq$ half the activations from the training corpus, while SGD has a significant number of features activate for more than half of the training corpus, i.e., Adam learns more selective features in this layer. Features which activate only for a small subset of the training corpus, and consequently see gradient updates from the main objective less frequently, continue to be acted upon by the regularizer. If the regularization is strong enough (Adam with L2: 1e-4 in Fig. 2), or the gradient updates infrequent enough (feature too selective), the feature may be pruned away entirely. The propensity of later layers to learn more selective features with Adam would explain the higher degree of sparsity seen for later layers as compared to SGD. Understanding the reasons for emergence of higher feature selectivity in Adam than SGD, and verifying if other adaptive gradient descent flavours also exhibit higher feature selectivity remains open for future investigation.

Quantifying Feature Selectivity: Similar to feature sparsity by activation, we apply max pooling to a feature's absolute activations over the entire feature plane. For a particular feature, we consider these pooled activations over the entire training corpus to quantify feature selectivity. See the original paper (Mehta et al., 2019) for a detailed discussion. Unlike the selectivity metrics employed in literature (Morcos et al., 2018), ours is class agnostic, and provides preliminary quantitative evidence that Adam (and perhaps other adaptive gradient descent methods) learn more selective features than (m)SGD, which consequently see a higher relative degree of regularization.

Interaction of L2 Regularizer with Adam: Next, we consider the role of the L2 regularizer vs. weight decay. In the original paper we study the behaviour of L2 regularization in the low gradient regime for different optimizers through synthetic experiments and find that coupling of L2 regularization with certain adaptive gradient update equations yields a faster decay than weight decay, or L2 regularization with SGD, even for smaller regularizer values. This is an additional source of regularization disparity between parameters which see frequent updates and those which don't see frequent updates or see lower magnitude gradients. It manifests for certain adaptive gradient descent approaches.

Task 'Difficulty' Dependence: As per the hypothesis developed thus far, as the task becomes more difficult, for a given network capacity, we expect the fraction of features pruned to decrease corresponding to a decrease in selectivity of the learned features (Zhou et al., 2018). Since the task difficulty cannot be cleanly decoupled from the number of classes, we devise a synthetic experiment based on grayscale renderings of 30 object classes from ObjectNet3D (Xiang et al., 2016). We construct 2 identical sets of $\approx 50k$ 64×64 pixel renderings, one with a clean background (BG) and the other with a cluttered BG. We train *BasicNet* with a mini-batch size of 40, and see that as expected there is a much higher sparsity (70%) with the clean BG set than with the more difficult cluttered set (57%). See the original paper (Mehta et al., 2019) for representative images and a list of the object classes selected.

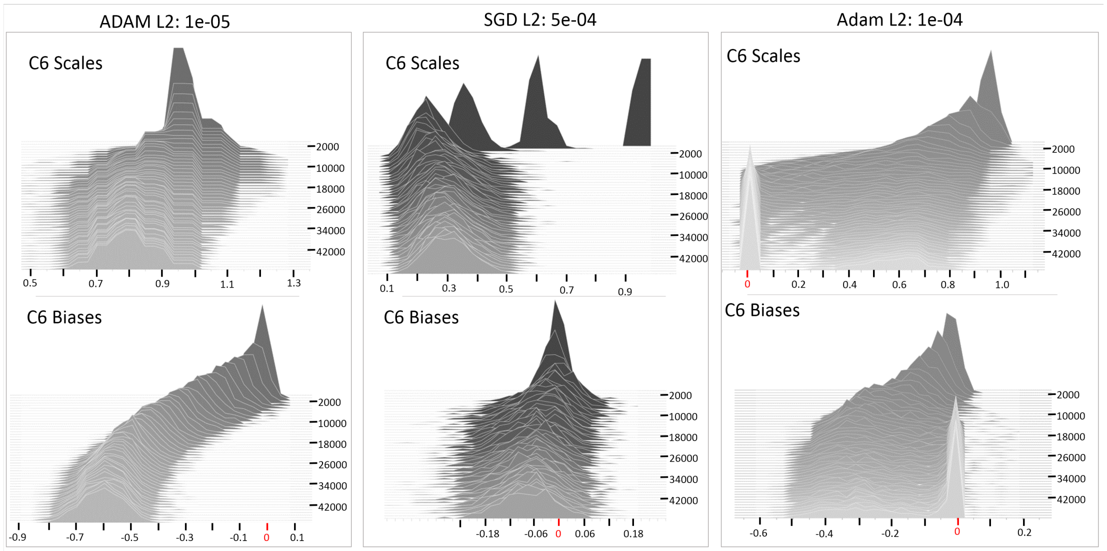

Figure 2. **Emergence of Feature Selectivity with Adam** The evolution of the learned scales (γ, top row) and biases (β, bottom row) for layer C6 of *BasicNet* for Adam and SGD as training progresses. Adam has distinctly negative biases, while SGD sees both positive and negative biases. For positive scale values, as seen for both Adam and SGD, this translates to greater feature selectivity in the case of Adam, which translates to a higher degree of sparsification when stronger regularization is used.

4. Related Work

(Ye et al., 2018; Liu et al., 2017) employ explicit filter sparsification heuristics that make use of the learned scale parameter γ in Batch Norm for enforcing sparsity on the filters. (Ye et al., 2018) argue that BatchNorm makes feature importance less susceptible to scaling reparameterization, and the learned scale parameters (γ) can be used as indicators of feature importance. We thus adopt γ as the criterion for studying implicit feature pruning.

Morcos et al. (Morcos et al., 2018) suggest based on extensive experimental evaluation that good generalization ability is linked to reduced selectivity of learned features. They further suggest that individual selective units do not play a strong role in the overall performance on the task as compared to the less selective ones. They connect the ablation of selective features to the heuristics employed in neural network feature pruning literature which prune features whose removal does not impact the overall accuracy significantly (Molchanov et al., 2017; Li et al., 2017). The findings of Zhou et al. (Zhou et al., 2018) concur regarding the link between emergence of feature selectivity and poor generalization performance. They further show that ablation of class specific features does not influence the overall accuracy significantly, however the specific class may suffer significantly. We show that the emergence of selective features in Adam, and the increased propensity for pruning the said selective features when using L2 regularization may thus be helpful both for better generalization performance and network speedup.

5. Discussion

Our findings would help practitioners and theoreticians be aware that seemingly unrelated hyperparameters can inadvertently affect the underlying network capacity, which interplays with both the test accuracy and generalization gap, and could partially explain the practical performance gap between Adam and SGD. Our work opens up future avenues of theoretical and practical exploration to further validate our hypotheses, and attempt to understand the emergence of feature selectivity in Adam and other adaptive SGD methods.

As for network speed up due to sparsification, the penalization of selective features can be seen as a greedy local search heuristic for filter pruning. While the extent of implicit filter sparsity is significant, it obviously does not match up with some of the more recent explicit sparsification approaches (He et al., 2018; Lin et al., 2017) which utilize more expensive model search and advanced heuristics such as filter redundancy. Future work should reconsider the selective-feature pruning criteria itself, and examine non-selective features as well, which putatively have comparably low discriminative information as selective features and could also be pruned. These non-selective features are however not captured by greedy local search heuristics because pruning them can have a significant impact on the accuracy. Though the accuracy can presumably can be recouped after fine-tuning.

References

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *European conference on computer vision*, pp. 630–645. Springer, 2016.

He, Y., Lin, J., Liu, Z., Wang, H., Li, L.-J., and Han, S. Amc: Automl for model compression and acceleration on mobile devices. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 784–800, 2018.

Li, H., Kadav, A., Durdanovic, I., Samet, H., and Graf, H. P. Pruning filters for efficient convnets. In *ICLR*, 2017.

Lin, J., Rao, Y., Lu, J., and Zhou, J. Runtime neural pruning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 2181–2191. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/6813-runtime-neural-pruning.pdf.

Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., and Zhang, C. Learning efficient convolutional networks through network slimming. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pp. 2755–2763. IEEE, 2017.

Loshchilov, I. and Hutter, F. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101*, 2017.

Mehta, D., Kim, K.-I., and Theobalt, C. On implicit filter level sparsity in convolutional neural networks. In *Proc. IEEE CVPR*, 2019.

Molchanov, P., Tyree, S., Karras, T., Aila, T., and Kautz, J. Pruning convolutional neural networks for resource efficient inference. 2017.

Morcos, A. S., Barrett, D. G., Rabinowitz, N. C., and Botvinick, M. On the importance of single directions for generalization. 2018.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Xiang, Y., Kim, W., Chen, W., Ji, J., Choy, C., Su, H., Mottaghi, R., Guibas, L., and Savarese, S. Objectnet3d: A large scale database for 3d object recognition. In *European Conference Computer Vision (ECCV)*. 2016.

Ye, J., Lu, X., Lin, Z., and Wang, J. Z. Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. In *ICLR*, 2018.

Zhou, B., Sun, Y., Bau, D., and Torralba, A. Revisiting the importance of individual units in cnns via ablation. *arXiv preprint arXiv:1806.02891*, 2018.