Can Selfless Learning improve accuracy of a single classification task?

Soumya Roy*
Indian Institute of Technology
Kanpur, India
meetsoumyaroy@gmail.com

Bharat Bhusan Sau*
Indian Institute of Technology
Hyderabad, India
sau.bharatbhusan@gmail.com

Abstract

The human brain has billions of neurons. However, we perform tasks using only a few concurrently active neurons. Moreover, an activated neuron inhibits the activity of its neighbors. Selfless Learning exploits these neurobiological principles to solve the problem of catastrophic forgetting in continual learning. In this paper, we ask a basic question: can the selfless learning idea be used to improve the accuracy of deep convolutional networks on a single classification task? To achieve this goal, we introduce two regularizers and formulate a curriculum learning-esque strategy to effectively enforce these regularizers on a network. This has resulted in significant gains over vanilla cross-entropy training. Moreover, we have shown that our method can be used in conjunction with other popular learning paradigms like curriculum learning.

1. Introduction

The human brain has billions of neurons. However, [15] has shown that we perform tasks using only a few concurrently active neurons. Moreover, an activated neuron inhibits the activity of its neighbors resulting in compact and decorrelated neural representations of physical stimuli [29]. Like the human brain, deep neural networks also have millions of trainable parameters. In this paper, we try to understand whether these neurobiological principles can be used to improve classification accuracy of deep convolutional networks.

Lifelong learning or continual learning deals with the problem of learning a sequence of tasks, one after another, without using the training data from previous or future tasks. A key challenge here is that the network forgets the previous tasks as it learns the current task. This is called catastrophic forgetting [16] and several methods have been proposed over the past few years to solve this problem [19]. [1] used the concept of neural inhibition to solve the problem of catastrophic forgetting by introducing a regularizer which penalizes neurons, within a pre-defined local neighbourhood, which are active at the same time. This results in a pool of "free" neurons which can be utilized by upcoming tasks - a learning paradigm which the authors call selfless learning (neurons learn *selflessly* by leaving capacity for future tasks). The resulting neural representation is sparse and decorrelated and outperforms methods like [28][3][21]. In this paper, we ask a basic question: can the selfless learning idea be used to improve the accuracy of popular deep convolutional networks like ResNet [11] in the *usual* classification setting (i.e., we have only one task and have access to the complete training dataset)? The solution may not be trivial because of following reasons:

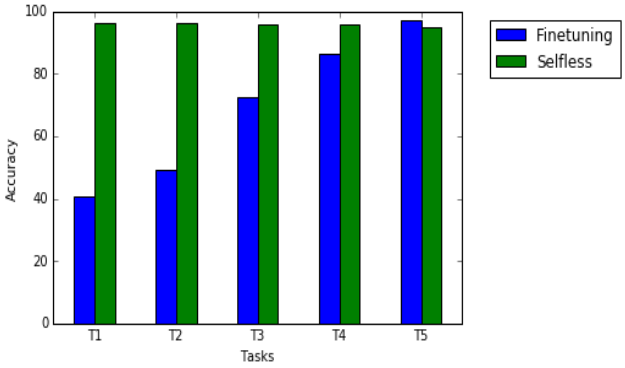

Figure 1. Selfless [1] vs Finetuning on permuted MNIST task sequence. Though the selfless method performs well on average, finetuning outperforms it on the last task.

1. In [1], the authors have applied the regularizer on neurons of fully connected layers. Consequently, it's efficacy on recent deep network architectures is unexplored.
2. On the permuted MNIST task sequence [7], the vanilla finetuning method (without continual learning regularizers) outperforms the selfless learning approach on the last task by around 2% (see Figure 1). So the self-

*Equal Contribution

less learning approach may not be immediately applicable to the usual classification setting.

Informally, the *representational capacity* of a neural network is it's ability to learn functions which map diverse input samples to their corresponding output labels [5]. The easiest way to increase it's *capacity* is to add more filters and layers to it. However, in this paper, we virtually increase the *capacity* of a neural network by borrowing the central idea of selfless learning. We hypothesize that easy samples would require less number of neurons to be learned, thereby leaving the harder samples more neurons to work with (thus improving their learnability). Specifically, we introduce two regularizers - one which penalizes neighbourhood activations of active neurons in a convolutional layer for easy samples and another which prevents representational overlap between the easy and hard samples (thus preventing the catastrophic forgetting of easy samples).

To summarize, our contributions are as follows:

1. We propose two regularizers which port the selfless learning idea to the usual classification setting. Moreover, we devise a curriculum learning-esque strategy which enables the network to better enforce these regularizers.
2. To demonstrate that our method is compatible with other popular learning paradigms, we combine our method with single step pacing method of [9].
3. We evaluate our method on 3 datasets - CUB-200-2011 [26], Stanford Cars [13] and ImageNet 2012 [22] and report significant improvements over vanilla cross-entropy training. Moreover, we perform through ablation studies to validate our hypothesis.

2. Previous works

Our method draws inspiration from two learning paradigms - selfless learning and curriculum learning.

2.1. Selfless Learning

Selfless learning is broadly related to increasing parameter sparsity and reducing representational overlap in neural networks. Parameter sparsity has been widely studied for network compression. In recent years, several approaches like model pruning [24, 18, 17] and SVD decomposition [28] have been used to reduce the network size without significantly impacting it's accuracy. However, parameter sparsity may not lead to sparsity in representation. [4] has minimized the L1 norm of the activation values from a rectifier activation function (ReLU) to promote sparsity. However, L1 norm gives equal importance to all active neurons resulting in small activation values. [3] has shown that minimizing the cross-covariance of neural activations help reduce overfitting in deep networks. Though this results in a decorrelated representation, the representation is not necessarily sparse. [1] combines [28] and [3] to produce a sparse and decorrelated representation that outperforms [28][3][21][8][25] in a sequential learning setting. The authors also empirically show that a sparse and decorrelated representation is better than applying parameter sparsity.

In lifelong learning or continual learning, a network learns a sequence of tasks without catastrophic forgetting of previously learned ones [16]. [1] migitated the problem of catastrophic forgetting by introducing a regularizer which penalizes nearby neurons which are active at the same time. As the regularizer promotes neural inhibition, the network has a pool of "free" neurons which can be used to learn future tasks (and hence the name *selfless* learning). Like [4], the use of ReLU activation function results in a sparse and decorrelated representation. Moreover, restricting neural suppression to a local neighbourhood enables the network to learn richer representations for complex tasks. The neighbourhood is defined by a Gaussian function parameterized by the hyperparameter σ. However, for tasks which are similar to previous tasks, the regularizer will encourage the network to only utilize neurons from previous tasks. This may lead to catastrophic forgetting of previous tasks. So the authors add a weight factor taking into account the importance of the neurons with respect to the previous tasks. It should be noted that the authors only apply this regularizer to the neurons of fully connected layers. Selfless learning should not be confused with the dropout technique where neurons are randomly masked. A key difference is that dropout forces different neurons to work independently, thereby decreasing the effective *capacity* of a network [6].

Our goal in this paper is to use the selfless learning idea to improve the accuracy of popular deep learning networks in the usual classification setting. To accomplish this goal, we propose two regularizers which increase the effective *representational capacity* of a neural network.

2.2. Curriculum Learning

In curriculum learning, we rank samples based on their *difficulty* and set up a pacing strategy to serve the classifier with progressively harder samples [2]. In [27], the features extracted from a network, trained on ImageNet [22], is used to rank samples in the current training dataset. [9] estimates sample hardness using bootstrapping - the given network is trained without curriculum which is then used for ranking samples. Furthermore, the authors investigate three pacing strategies which differ by the function used to increase mini-batch size - fixed exponential, varied exponential and single step.

Like curriculum learning, we use a scoring function to partition our training dataset. Moreover, we devise a curriculum learning-esque strategy to further improve the accuracy of our selfless learning approach.

3. Method

In the context of continual learning, [1] has proposed a regularizer which penalizes nearby neurons which are active at the same time. This method leaves enough "free" neurons in the network to learn future tasks and thus prevents catastrophic forgetting of previous tasks. Moreover, to discourage the current task from interfering with previous tasks, the authors add a weight factor to the regularizer which encodes the importance of the neurons with respect to the previous tasks. In this work, we want to use this idea to improve the classification accuracy of popular deep convolutional networks when there is only one task (i.e., we have access to the complete dataset). This problem can have two naive solutions:

1. Using *some* heuristics, partition the dataset into multiple smaller datasets (and hence multiple tasks) and apply the regularizer in the usual continual learning setting. However, as shown in [1], multi-task joint training significantly outperforms sequential training and hence we do not explore this direction in this paper.
2. We consider classification on the complete dataset as one task and apply the regularizer (without neuron importance as there is only one task). This solution is a special case of our method and is used as a baseline in Section 4.4 (see scenario *naive*). However, this approach muddles the original motivation of the regularizer as there is little use of the "free" neurons.

We hypothesize that easier samples would require less number of neurons to be learned and hence we try to minimize the number of active neurons in the local neighbourhood while learning these samples. This strategy gives the harder samples more neurons to work with, thereby increasing it's *representational capacity*.

Our method has three parts:

1. **Sample selection strategy**: The sample selection strategy helps us partition the training dataset into *easy* and *hard* samples. We can sort the training samples using either the current hypothesis of the network like [14] or some target hypothesis like [9]. Here we consider uncertainty sampling [23] as our sample selection strategy. The probability of the most confident class in a sample can be considered as the classifier confidence in that sample. We rank the samples using the confidence of the current network hypothesis (more confidence implies easier sample).
2. **Regularizers**: We fine-tune a given network using the following regularizers (along with the usual cross-entropy loss):

   (a) For a feature map of size $C \times H \times W$ and a mini-batch size M, we define the first regularizer as:

$$L_1 = \sum_c \sum_{i_c, j_c} e^{-\frac{(i_c - j_c)^2}{2\sigma^2}} \sum_m a^m_{i_c} a^m_{j_c} (i_c \neq j_c) \tag{1}$$

   where i_c and j_c are locations on a $H \times W$ feature map for a given channel c where $c \in 1, ..., C$, $m \in 1, ..., M$ and a^e_l is the activation for example e and location l. We normalize the regularizer by dividing it by $(M \times C \times H \times W)$. Furthermore, to reduce computational overhead and enable richer representations for complex tasks, we constraint i_c and j_c to take values from the same column/row of the H $\times$ W matrix (the Gaussian is 1D in the formulation). This regularizer is applied on the easy samples and has the following interpretation:

   If one neuron goes off for a easy sample, others in the neighbourhood (defined by σ) will be quieter.

   A similar loss function is used by [1] on fully connected layers in the context of continual learning.

   (b) For a feature map of size $C \times H \times W$ and a mini-batch size M, we define the second regularizer as:

$$L_2 = \sum_c \sum_{i_c} \sum_{m_d} \alpha^{m_d} \sum_{m_s} a^{m_s}_{i_c} a^{m_d}_{i_c}$$
$$\alpha^{m_d} = 1 - max_{m_s} \widehat{a^{m_d}} \cdot \widehat{a^{m_s}} \tag{2}$$

   where i_c is a location on a $H \times W$ feature map for a given channel c where $c \in 1, ..., C$, (m_s, m_d) are pairs of easy and hard samples (s is used to denote simple and d to denote difficult) in the minibatch, a^e_l is the activation for example e and location l, a^e is the activation for example e flattened to a column vector and α^{m_d} encodes the similarity between a hard sample m_d and all easy samples. We normalize the regularizer by dividing it by $(M \times C \times H \times W)$. This regularizer is applied on pairs of easy and hard samples and has the following interpretation:

   If one neuron goes off for an easy sample, that will be quieter for the hard sample. If the hard sample is similar to the easy samples, the penalization will be less.

   By including the contradictory themes of compartmentalization and similarity, this regularizer tries to strike a balance between reduction

of catastrophic forgetting for easy samples and wasteful allocation of "free neurons" to hard samples, which are very similar to easy samples.

If the activation function is ReLU (as is the case in this paper), we can replace the summations by L1 norm.

3. **Curriculum Learning-esque strategy**: In each training epoch, we first train our network on the easy samples using the following loss function:

$$\text{Cross-entropy loss} + \gamma L_1 \tag{3}$$

Subsequently, we train on both easy and hard samples using the following loss function:

$$\text{Cross-entropy loss} + \beta L_1 + \theta L_2 \tag{4}$$

So, in every epoch, the complete training dataset is used and thus this strategy is different from traditional curriculum learning approaches like [9]. This strategy enables the network to better enforce the selfless learning regularizers.

We use the term *suppression ratio* to denote the fraction of training samples on which neural suppression (i.e., Eq. 1) is applied. In other words, this ratio decides the number of *easy* samples. The availability of "free" neurons for hard samples is affected by two major factors - number of easy samples (or suppression ratio) and inter-class similarity. If there is considerable similarity among samples, we can increase the suppression ratio as the different samples can re-use currently active neurons. So, for datasets like CUB [26], the suppression ratio is kept higher than that for ImageNet as there is an inherent similarity between images of different classes in fine-grained datasets.

Our regularizers are applied on the feature maps obtained from pre-determined convolutional layers. For example, the ResNet-18 network [11] has four residual blocks and we apply our regularizers after every block (barring the last one). If r_1, r_2 and r_3 are the suppression ratios for the first three feature maps (going from input to output), then we will use the (nonstrictly) increasing suppression ratio sequence $r_1 <= r_2 <= r_3$ in this paper.

4. Experimental Evaluation

To judge the efficacy of our method, we consider two kinds of datasets - fine-grained (see Section 4.1) and regular (see Section 4.2). In Section 4.3, we plug our approach into curriculum learning. In Section 4.4, we perform through ablation studies on our approach. In Section 4.5, we examine the impact of our method on representation sparsity.

4.1. Fine-grained datasets

We evaluate our method on three fine-grained datasets - CUB-200-2011 [26], Stanford Cars [13] and cat subset of ImageNet 2012 [22].

4.1.1 Experimental Setup

We evaluate our method using two architectural families - ResNet [10] and MobileNet [12]. The selfless learning regularizers are applied on the feature map obtained at the end of each residual block (except the last one).

A network can be initialized randomly or with pre-trained weights. For pre-trained networks, we use L2 norm in Equations 1 and 2, while for randomly initialized networks, we use L1 norm. The network initialization method has the following implications for our approach:

1. Our sample selection strategy depends on the current network hypothesis.
2. As shown in [1], the selfless approach works better when the network is randomly initialized.

So for CUB and Stanford Cars, we experiment with two initialization strategies - random and pre-trained on ImageNet. We use the pre-trained weights available in PyTorch library [20] for simplicity of reproduction. For MobileNetv1, pre-trained weights are not available and hence we do not evaluate our method using pre-trained MobileNetv1.

Our method has five hyperparameters - γ, β, θ, σ and suppression ratio. So to reduce the scope of hyperparameter tuning, we use the following constraints:

- $\beta = \theta \in \{1, 0.1\}$
- $\gamma \in \{1, 0.1, 0.01\}$
- $\sigma = 32$
- suppression ratios are respectively 0.55, 0.6 and 0.65. We have selected these values for suppression ratios as they perform well on average for all three datasets.

The other hyperparameters are fixed for any network, i.e. max epoch is 100, batch size is 128 (for ResNet-50, we use a batch size of 64 because of limitations on GPU memory), optimizer is Stochastic Gradient Descent (SGD), initial learning rate is 0.1, weight decay is 5e-4 and the learning rate is decayed by 0.1 after every 30 epochs. We use our regularizers after training the network with cross-entropy for 30 epochs. We also apply the sample selection strategy on the current network hypothesis after every 30 epochs and rank the training samples.

In this section, we consider the vanilla cross-entropy loss function as our baseline. It should be noted that the optimization hyperparameters are kept same for both our method and the baseline.

4.1.2 CUB

CUB-200-2011 [26] is a popular dataset for fine-grained image classification. It has 200 classes and over 11000 images. We use the standard Top-1 test accuracy as our metric. Results are presented in Table 1.

	pre-trained		random	
architecture	CE	SL	CE	SL
ResNet-18	72.24	**72.98**	48.62	**50.96**
ResNet-34	73.63	**74.77**	47.53	**50.28**
ResNet-50	51.04	**65.87**	27.56	**46.86**
MobileNetv1	-	-	50.82	**53.54**

Table 1. Results on CUB: Here CE=cross-entropy, SL=our selfless learning method, pre-trained=network pre-trained on ImageNet and random=randomly initialized network

4.1.3 Stanford Cars

Stanford Cars [13] is another popular dataset for fine-grained image classification. It has 196 classes and over 16,000 images. We use the standard Top-1 test accuracy as our metric. Results are presented in Table 2.

	pre-trained		random	
architecture	CE	SL	CE	SL
ResNet-18	75.13	**85.92**	63.89	**69.75**
ResNet-34	87.58	**88.20**	59.73	**71.3**
ResNet-50	81.22	**84.76**	16.2	**66.48**
MobileNetv1	-	-	73.40	**73.42**

Table 2. Results on Stanford Cars: Here CE=cross-entropy, SL=our selfless learning method, pre-trained=network pre-trained on ImageNet and random=randomly initialized network

4.1.4 ImageNet Cats

We evaluate our method on cat subset of ImageNet 2012 [22]. It is used as a baseline for curriculum learning in [9]. It has 7 classes and over 9000 images each of size 224×224. We use the standard Top-1 validation accuracy as our metric. Results are presented in Table 3.

4.1.5 Observations

- Our method outperforms cross-entropy by a significant margin on all the three datasets. These results demonstrate that selfless learning can indeed be used to improve the accuracy of popular deep convolutional networks on (at least) fine-grained datasets.
- Irrespective of the network depth, the gap between our method and cross-entropy holds.

	random	
architecture	CE	SL
ResNet-18	66.25	**67**
ResNet-34	62.75	**66**
ResNet-50	57.25	**70**
MobileNetv1	72.75	**74**

Table 3. Results on ImageNet Cats: Here CE=cross-entropy, SL=our selfless learning method and random=randomly initialized network

- As depicted in Table 4, even though our method is sensitive to hyperparameters, all hyperparameter combinations significantly outperform the cross-entropy method.

	γ		
β	1.0	0.1	0.01
1.0	69.75	69.11	69.34
0.1	69.22	69.01	68.79

Table 4. Results on Stanford Cars: Understanding hyperparameter sensitivity for a randomly initialized ResNet-18 network with baseline accuracy of 63.89

4.2. ImageNet

The results in Section 4.1 raise an important question: will the gap between our method and cross-entropy also hold for regular datasets like ImageNet 2012 [22]?

ImageNet 2012 has 1000 classes and over 1.2 million training images each of size 224×224. The models are evaluated on a validation set of 50,000 images using Top-1 single-crop validation accuracy. It should be noted that the ImageNet dataset can be considered as an union of fine-grained and non fine-grained datasets and thus presents a more practical scenario. Results are presented in Table 5.

Here, we initialize our network with the pre-trained weights and set our selfless method hyperparameters as follows:

- $\beta = \theta = 0.1$
- $\gamma = 1.0$
- $\sigma = 32$
- suppression ratios are respectively 0.02, 0.04 and 0.08

The other hyperparameters are fixed for any network, i.e., max epoch is 40, batch size is 256, optimizer is Stochastic Gradient Descent (SGD), initial learning rate is 0.001, weight decay is 1e-4 and the learning rate is decayed by 0.1 after every 15 epochs.

As before, we consider the vanilla cross-entropy loss function as our baseline. It should be noted that the optimization hyperparameters are kept same for both our method and the baseline.

As shown in Table 5, the gains on ImageNet from the selfless learning method is small when compared to fine-grained datasets.

	pre-trained	
architecture	CE	SL
ResNet-18	70.53	**70.69**
ResNet-34	74.04	**74.15**

Table 5. Results on ImageNet Cats: Here CE=cross-entropy, SL=our selfless learning method and pre-trained=network pre-trained on ImageNet

4.3. Curriculum Learning

As any modern deep learning method is much more than just training using cross-entropy loss, we need to demonstrate that our method can be combined with other learning paradigms. Specifically, we plug our approach into the *single step pacing* method of [9]. This method is simple to implement and performs just as well as the other variants in [9]. Our method helps improve the accuracy of the final stage of curriculum learning in which mini-batches are sampled uniformly from the entire training dataset.

4.3.1 Experimental Setup

Here we consider ResNet-18 as our base network. Like [9], we evaluate our methods on the cat subset of ImageNet. However, unlike [9], we use images of size 224×224.

We start off by training a ResNet-18 network from scratch using the following hyperparameters:
max epoch is 100, batch size is 128, optimizer is Stochastic Gradient Descent (SGD), initial learning rate is 0.1, weight decay is 5e-4 and the learning rate is decayed by 0.1 after every 30 epochs.

This model is then used to sort the training samples by their confidence scores (more confidence implies easier samples) - an example of bootstrap scoring function. We select the first 15% high scoring samples and train a ResNet-18 network from scratch for 70 epochs with initial learning rate of 0.01 and learning rate decay of 0.1 after every 30 epochs. We have arrived at these hyperparameters through a hyperparameter search on the test set. This model serves as the initial model for selfless learning.

Subsequently, we train the same network on the complete training dataset for 100 more epochs with initial learning rate as 0.1 and learning rate decay of 0.1 every 30 epochs. It should be noted that for curriculum learning we use cross-entropy as our loss function and for selfless learning we use Equation 4. We use the following hyperparameters for selfless learning:

- $\beta = \theta = 0.1$
- $\gamma = 0.01$
- $\sigma = 32$
- suppression ratios are respectively 0.55, 0.6 and 0.65 (same as the fine-grained case in 4.1)

model type	Top-1
initial model	48.25
cross-entropy	66.25
single step pacing [9]	70.0
selfless learning	**70.25**

Table 6. Results on ImageNet Cats for curriculum learning

The results in Table 6 demonstrate that our approach would work well with any initial model and thus can be plugged into popular learning paradigms.

4.4. Ablation Studies

We use the following scenarios to judge the efficacy of our method:

- **random:** What happens when we train a network using the selfless learning method and random as the sample selection strategy? So in this scenario, we select random samples from the training dataset and treat them as *easy* samples (i.e., Equation 1 is applied on random samples). This is a popular baseline in curriculum learning [9] and helps prove our hypothesis. This experiment is repeated three times and we report the mean accuracy. We set β, θ, γ to the best hyperparameters of the selfless method for a fair comparison.

- **anti-curriculum:** What happens when we train a network using the selfless learning method and anti-curriculum as the sample selection strategy? So in this scenario, we select the hardest samples from the training dataset and treat them as *easy* samples (i.e., Equation 1 is applied on hard samples). This is another popular baseline in curriculum learning [9] and helps prove our hypothesis. We set β, θ, γ to the best hyperparameters of the selfless method for a fair comparison.

- **θ=0:** How does our method perform when θ is set to 0? This scenario demonstrates the usefulness of the compartmentalization of easy and hard samples using Equation 2. We set β and γ to the best hyperparameters of the selfless method for a fair comparison.

- **curriculum** What happens when we set $\beta = \theta = \gamma$ to 0? This scenario demonstrates the usefulness of Equations 1 and 2.

- **naive:** What happens when we set $\theta = 0$, remove the curriculum learning strategy and suppression ratio as 1? This scenario is the second naive solution discussed at the beginning of this paper. We report our results for the best $\beta \in \{1, 0.1\}$ (same search range as the selfless approach).

- **DeCov:** We apply the Decov loss [3] on the average pooled feature vector from the last convolutional layer as follows:

$$\text{Cross-entropy loss} + \beta \, \text{DeCov} \tag{5}$$

  We vary the value of β from 1e-5 to 1 in steps of 10 and report the best results.

- **equal:** What happens when we use a constant suppression ratio sequence? We set β, θ, γ to the best hyperparameters of the selfless method for a fair comparison.

- **decreasing:** What happens when we use a strictly decreasing suppression ratio sequence? We set β, θ, γ to the best hyperparameters of the selfless method for a fair comparison.

To evaluate each scenario, we use MobileNetv1 as our base network, ImageNet Cats as our dataset and Top-1 validation accuracy as our metric. As is clear from Table 7, most of these scenarios fail to even outperform the vanilla cross-entropy training.

scenario	Top-1
cross-entropy	72.75
selfless	**74**
random	72
anti-curriculum	72.5
θ=0	71.5
curriculum	72.5
naive	72.25
DeCov	73.25
equal	**74**
decreasing	73.75

Table 7. Ablation Studies on ImageNet Cats

4.5. Representation sparsity

To understand the impact of our method on representation sparsity, we perform the following experiments:

- We use Equation 4 to train a randomly initialized ResNet-18 network on Stanford Cars dataset. Then we evaluate the mean activation values of three feature maps corresponding to the first three residual blocks. Figure 2 demonstrates that our regularizers promote lower mean activation values than vanilla cross-entropy.

- We visualize the feature maps of an *easy* image for both selfless and cross-entropy methods. As is clear from Figure 3, the selfless method promotes neural inhibition for *easy* images.

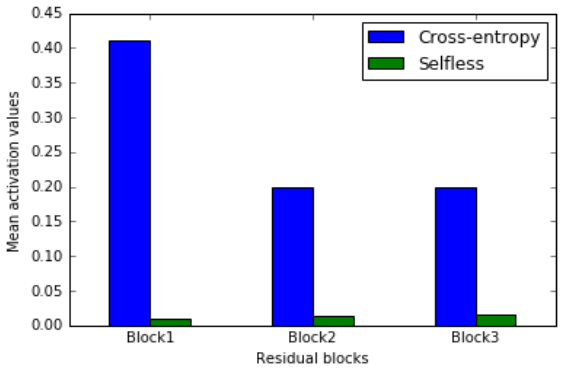

Figure 2. Here we compare the selfless method with cross-entropy using mean activation value as our metric.

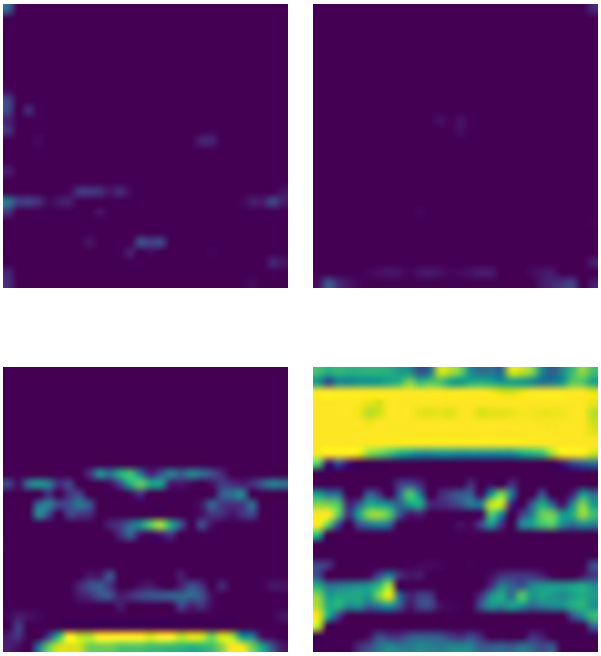

Figure 3. Feature maps from selfless (top) and cross-entropy (bottom) methods corresponding to an *easy* image

5. Conclusion

In this paper, we have used the selfless learning idea to solve the usual classification problem. This has resulted in majors gains over the vanilla cross-entropy training. Moreover, we have shown that our method can be used in conjunction with other popular learning paradigms. In future, we would like to devise a principled approach to determine the best suppression ratio for a given dataset.

References

[1] Rahaf Aljundi, Marcus Rohrbach, and Tinne Tuytelaars. Selfless sequential learning, 2018.

[2] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 41–48, New York, NY, USA, 2009. Association for Computing Machinery.

[3] Michael Cogswell, Faruk Ahmed, Ross Girshick, Larry Zitnick, and Dhruv Batra. Reducing overfitting in deep networks by decorrelating representations, 2015.

[4] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. volume 15 of *Proceedings of Machine Learning Research*, pages 315–323, Fort Lauderdale, FL, USA, 11–13 Apr 2011. JMLR Workshop and Conference Proceedings.

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[6] Ian J. Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks, 2013.

[7] Ian J. Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks, 2015.

[8] Ian J. Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks, 2013.

[9] Guy Hacohen and Daphna Weinshall. On the power of curriculum learning in training deep networks. *CoRR*, abs/1904.03626, 2019.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[12] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[13] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

[14] M. P. Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1189–1197. Curran Associates, Inc., 2010.

[15] Peter Lennie. The cost of cortical computation. *Current Biology*, 13(6):493–497, 2003.

[16] Zhizhong Li and Derek Hoiem. Learning without forgetting. *CoRR*, abs/1606.09282, 2016.

[17] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming, 2017.

[18] Christos Louizos, Karen Ullrich, and Max Welling. Bayesian compression for deep learning, 2017.

[19] German Ignacio Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *CoRR*, abs/1802.07569, 2018.

[20] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[21] Pau Rodríguez, Jordi Gonzàlez, Guillem Cucurull, Josep M. Gonfaus, and F. Xavier Roca. Regularizing cnns with locally constrained decorrelations. *CoRR*, abs/1611.01967, 2016.

[22] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[23] Burr Settles. Active learning literature survey. Technical report, 2010.

[24] Suraj Srinivas and R Venkatesh Babu. Data-free parameter pruning for deep neural networks. *arXiv preprint arXiv:1507.06149*, 2015.

[25] Rupesh K Srivastava, Jonathan Masci, Sohrob Kazerounian, Faustino Gomez, and Jürgen Schmidhuber. Compete to compute. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2310–2318. Curran Associates, Inc., 2013.

[26] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

[27] D. Weinshall and Gad Cohen. Curriculum learning by transfer learning: Theory and experiments with deep networks. *ArXiv*, abs/1802.03796, 2018.

[28] Jian Xue, Jinyu Li, and Yifan Gong. Restructuring of deep neural network acoustic models with singular value decomposition. In *Interspeech*, January 2013.

[29] Yuguo Yu, Michele Migliore, Michael L. Hines, and Gordon M. Shepherd. Sparse coding and lateral inhibition arising from balanced and unbalanced dendrodendritic excitation and inhibition. *Journal of Neuroscience*, 34(41):13701–13713, 2014.