sensors

MDPI

Article

Building a Compact Convolutional Neural Network for Embedded Intelligent Sensor Systems Using Group Sparsity and Knowledge Distillation

Jungchan Cho [1] and Minsik Lee [2,*]

[1] Department of Software, Gachon University, Seongnam 13120, Korea; thinkai@gachon.ac.kr
[2] Division of Electrical Engineering, Hanyang University, Ansan 15588, Korea
* Correspondence: e-mail: mleepaper@hanyang.ac.kr; Tel.: +82-31-400-5173

Received: 14 August 2019; Accepted: 1 October 2019; Published: 4 October 2019

Abstract: As artificial intelligence (AI)- or deep-learning-based technologies become more popular, the main research interest in the field is not only on their accuracy, but also their efficiency, e.g., the ability to give immediate results on the users' inputs. To achieve this, there have been many attempts to embed deep learning technology on intelligent sensors. However, there are still many obstacles in embedding a deep network in sensors with limited resources. Most importantly, there is an apparent trade-off between the complexity of a network and its processing time, and finding a structure with a better trade-off curve is vital for successful applications in intelligent sensors. In this paper, we propose two strategies for designing a compact deep network that maintains the required level of performance even after minimizing the computations. The first strategy is to automatically determine the number of parameters of a network by utilizing group sparsity and knowledge distillation (KD) in the training process. By doing so, KD can compensate for the possible losses in accuracy caused by enforcing sparsity. Nevertheless, a problem in applying the first strategy is the unclarity in determining the balance between the accuracy improvement due to KD and the parameter reduction by sparse regularization. To handle this balancing problem, we propose a second strategy: a feedback control mechanism based on the proportional control theory. The feedback control logic determines the amount of emphasis to be put on network sparsity during training and is controlled based on the comparative accuracy losses of the teacher and student models in the training. A surprising fact here is that this control scheme not only determines an appropriate trade-off point, but also improves the trade-off curve itself. The results of experiments on CIFAR-10, CIFAR-100, and ImageNet32 $\times$ 32 datasets show that the proposed method is effective in building a compact network while preventing performance degradation due to sparsity regularization much better than other baselines.

Keywords: convolutional neutral network; deep learning; group sparsity; knowledge distillation; parameter reduction.

1. Introduction

Embedded intelligence is characterized by the ability to deliver smart systems or services to the industry through the integration of software and hardware. This kind of embedded intelligence can serve many purposes, such as the smart monitoring of user-health or human behaviors and in systems to be used by the aged or the disabled. However, for the information collected from sensors to be used for such embedded intelligence, it is necessary for the sensor signal to be processed by machine learning [1–5]. The use of deep learning is the best choice for this task because deep networks have achieved state-of-the-art performance in many major machine learning areas, such as computer vision and natural language processing [6–8]. Moreover, deep learning has expanded beyond academic research into various commercial fields, which has been considered a quasi-revolution.

However, there are still problems to be solved for embedding deep-network-based functionality on intelligent sensors. A major problem is that the expressive power comes from the increased number of learnable parameters, making it impractical to use deep networks on limited platforms, such as mobile phones and robots, and on sensors with low-capacity processing power in devices like autonomous vehicles. Furthermore, a large number of parameters could lead to model overfitting, if there are insufficient training data [9].

Unfortunately, even though it is well known that deep neural networks have many redundant parameters [10,11] and can be replaced by more compact architectures, designing compact deep architectures for new tasks still remains a dark art: it requires the determination of the number of parameters or the complexity of the model, which can be a difficult problem itself. The complexity is typically set manually through trial and error, resulting in a hard trade-off between accuracy and the number of trainable parameters.

Several attempts have been made to overcome such difficulties in creating lightweight networks. One method is to transfer knowledge from a cumbersome (teacher) model to a smaller one (student network), which is more appropriate for limited platforms [12–17]. However, this method has a problem: it is assumed that the correct number of parameters for a student network is already known during the training phase.

Another approach to achieve a more compact deep network is to learn a compact structure by removing unnecessary weights from a large initial network (parameter reduction) [18–21]. For example, utilizing sparse regularizes, such as the l_1-norm [22] or $l_{2,1}$-norm [23], helps in reducing the redundant neurons or layers. However, a major problem with this approach is that it often achieves this efficiency at the expense of accuracy. How can we then obtain a compact network without sacrificing accuracy? It still remains an open problem.

In this paper, we propose two strategies to overcome these difficulties and design a compact network while minimizing the degradation of accuracy, as shown in Figure 1:

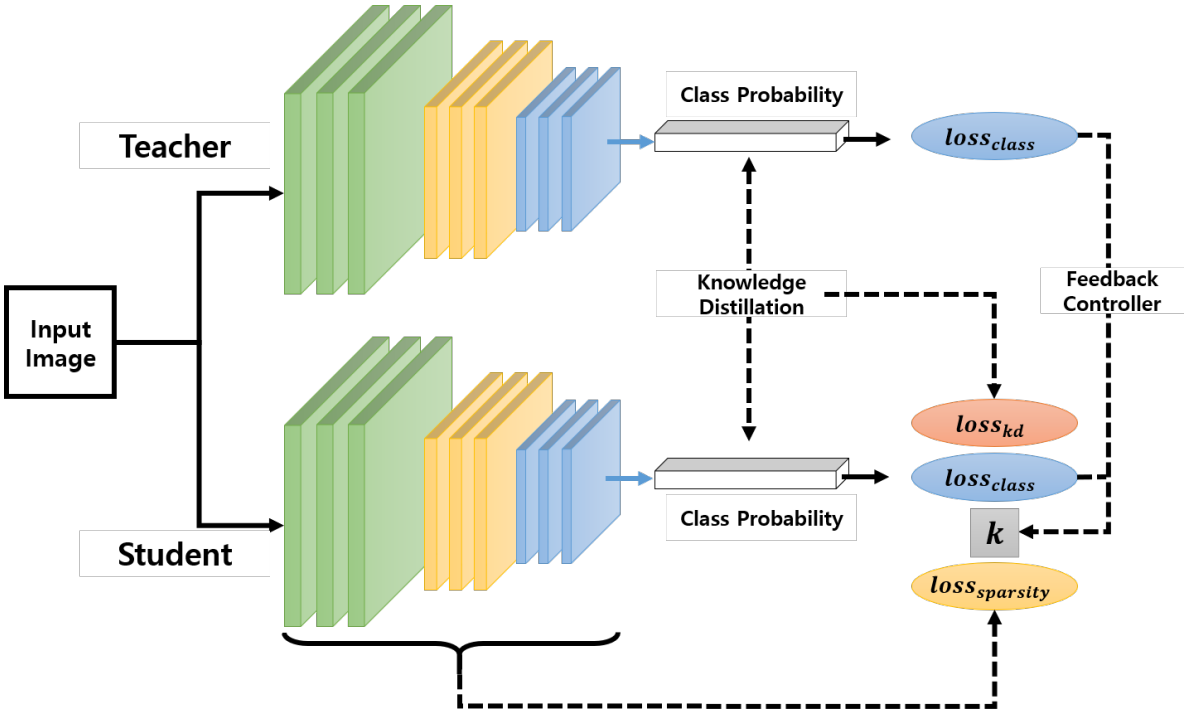


Figure 1. Overview.

- The first strategy is determining the number of parameters of a student network automatically by utilizing group sparsity combined with knowledge transfer in the training process. It is already well known that a student network with a small number of parameters trained by using

knowledge distillation (KD) loss leads to a better solution than those without KD. Our intuition at this point is that after the number of parameters of the student network is reduced based on a sparse loss, the resulting network can be regarded as a smaller network and the performance of such a compact network can be improved by the KD loss.

- The second strategy is to control the regularization parameter for balancing the accuracy and sparsity losses. For achieving this, we utilize a feedback control mechanism based on the proportional control theory [24–27], where the controller output is proportional to the difference between the desired setpoint and a measured value from a process (error signal). In our case, one of our goals is to make the student model mimic a cumbersome teacher model. Considering this goal in terms of control, the desired setpoint and measured process variable can be set to the accuracy losses of the teacher and student models, respectively. In addition, the feedback mechanism corresponds to adjusting the weight of the sparsity loss, i.e., the amount of emphasis put on the sparsity during the training.

Due to our assumption that the embedded sensors are low-capacity processing units, we evaluated the proposed algorithm on small-sized image datasets, i.e., CIFAR-10, CIFAR-100 [28], and ImageNet32×32 datasets [29]. The results of our experiments on these three datasets show that the proposed method helps to build a compact network by preventing performance degradation when using sparse regularization and improves the classification performance even better than KD.

This paper comprises the following sections. Section 2 describes related works on the utilization of KD, sparsity-regularized methods to obtain a compact network, and adaptive regularization. Section 3 illustrates the proposed method with KD and sparsity losses and then introduces the proposed control method for adjusting the regularization parameter to achieve a balance between the accuracy and compactness of a deep network. The detailed results of the conducted experiments are shown in Section 4. Section 5 presents the conclusions of this paper.

2. Related Work

It is well known that deep neural networks have many redundant parameters, and can be replaced with more compact architectures. The main problem in deploying a compact network is the increased difficulty in training the network even though the expressive power of the model is sufficient. There are two types of approaches to design a small model.

2.1. Knowledge Distillation

Hinton et al. [12] proposed the very first "knowledge distillation" method. In their seminal paper, they eased the training of deep networks by following a student-teacher paradigm, where the student model is trained based on the teacher network's output, which can be viewed as a softened version of the ground-truth label, alongside the true classification label. Romero et al. [13] extended this idea by allowing the training of a deeper and thinner student than the teacher by using not just the outputs but also the intermediate representations learned by the teacher network as a hint for the student network to improve the KD. Zagoruyko et al. [14] proposed a method inspired by the attention mechanism that plays a significant role in the human visual system. In their work, the attention of a convolutional neural network (CNN) is defined as the aggregated information of the channel responses from an intermediate layer, i.e., attention maps, and a student network is forced to mimic the attention maps of a powerful teacher network. Recently, there have been efforts to use generative adversarial schemes for KD [16] and a learning strategy for one-stage online distillation [17]. But most of the studies assume that the number of parameters of the student network is fixed beforehand.

2.2. Parameter Reduction

This approach allows us to learn a compact structure directly from a bigger deep neural network (DNN). Alvarez et al. [19] introduced an approach to automatically determine the number of neurons

in each layer of a deep network during learning. Starting from an overcomplete network, they reduced the number of parameters with a group sparsity regularizer. Wei et al. [20] also proposed a group sparsity learning method to regularize the DNN structures (i.e., filters, channels, filter shapes, and layer depth). Yoon et al. [9] proposed an exclusive sparsity regularization approach based on both $l_{1,2}$-norm and $l_{2,1}$-norm, which exploits both the positive and negative correlations among the features to enforce sparsity on the network, and removes any redundancy among the features to help utilize the full capacity of the network. However, one main problem with this approach is that it often achieves this efficiency at the expense of accuracy.

2.3. Adaptive Regularization

Adaptive regularization itself is a long-studied subject in various fields [30]. Several attempts have been made to apply adaptive regression for sparse learning. Zhang et al. [31], proposed a dynamic shrinking scheme for adaptive l1-regularization. Dong et al., [32] applied adaptive sparse domain selection and adaptive regularization to image blurring and super-resolution problems. There have also been attempts to apply adaptive regularization to neural networks [33–35]. Larsen et al. [33] proposed a method to minimize the estimate of the generalization error, which is obtained by cross-validation, by adjusting the weight decay parameter. On the other hand, Leung and Chow [34] discretely changed the parameter whenever it seems that the optimization stalls in a suboptimal solution, i.e., the gradients of the data term and the regularization term are not zero. Cho et al. [35] proposed a method to update the regularization parameter based on a diffusion process driven by a heat equation. However, all these methods focus on determining the traditional neural network parameters such as weight decay to improve generalization methods. As far as we know, there is yet an attempt to formulate adaptive regularization based on knowledge distillation for determining the sparsity of a neural network, as proposed in this paper. The proposed method is fundamentally different from the existing works in two aspects: (i) the goal of adaptive regularization in the proposed method is to determine the sparsity of neural networks, and (ii) it brings information from other (teacher) models for the control of the regularization parameter.

3. Proposed Method

This section details the proposed student-teacher framework with group sparsity and proportional control. First, we review the representative KD method proposed by Hinton et al. [12]. Second, we combine it with a group sparsity regularizer on the student network to find a network smaller than the initial one throughout the training phase. Finally, we show how to use the proportional control theory to balance the accuracy and sparsity losses.

3.1. Review of Knowledge Distillation Based on a Soft Target Distribution

The main idea of [12] was to allow the student network to learn the soft target distribution produced by the teacher network as well as the true label. The framework can be summarized as follows.

Let T and S be the teacher and student networks, respectively, and $\mathcal{I}$ be an indicator for representing either the teacher or the student, i.e., $\mathcal{I} \in \{T, S\}$. Then, the class probabilities produced by a softmax output layer, a typical output of a neural network, can be represented as $\mathbf{p}_\mathcal{I} = \mathsf{softmax}(\mathbf{a}_\mathcal{I})$, where $\mathbf{a}_\mathcal{I}$ is the vector of the pre-softmax activations of a neural network. Hinton et al. [12] introduced a relaxation parameter $\tau \geq 1$ to soften the output of the neural network, i.e., $\mathbf{p}(\tau)_\mathcal{I} = \mathsf{softmax}(\mathbf{a}_\mathcal{I}/\tau)$, thereby allowing more information to be transferred from the teacher to the student network during training.

To force the student network to learn the pre-trained teacher's output, as well as the true label, the weighted average of the two-loss functions is used as follows:

$$\mathcal{L}_{KD}(\mathbf{W}_S) = \mathcal{H}(\mathbf{y}_{true}, \mathbf{p}_S) + \lambda_{kd}\mathcal{H}(\mathbf{p}_T(\tau), \mathbf{p}_S(\tau)), \tag{1}$$

where $\mathbf{W_S}$ denotes the collection of weights in the student network, $\mathcal{H}(\cdot)$ indicates the cross-entropy, λ_{kd} is a tunable parameter to balance the two loss terms, and $\mathbf{y}_{true}$ is the true label of a training sample.

However, knowing the right number of parameters in advance for a student network is difficult in practice, i.e., it requires many trials and errors to arrive at the appropriate number of parameters for a student. To address this issue, we introduce the first strategy: incorporating a sparsity regularizer in the optimization. Starting from the next section, we will omit the subscript S in $\mathbf{W}_S$ because the only weights being optimized in the proposed method are those of the student network, not those of the teacher's.

3.2. Knowledge Distillation With Group Sparse Regularization

To explain the first strategy of the proposed method for designing a compact network, we first define a sequence of 4-D tensors as the weights of convolutional layers. Let L be the number of convolutional layers and $\mathbf{W}^{(l)} \in \mathbb{R}^{M_l \times K_l \times C_l \times N_l}$ be the l-th ($1 \leq l \leq L$) weight tensor, where M_l, K_l, C_l, and N_l are the dimensions along the axes of spatial height, spatial width, channel, and filter, respectively. Then, the proposed optimization objective with KD and group sparsity regularization can be formulated as:

$$\arg\min_{\mathbf{W}} \mathcal{L}_{KD}(\mathbf{W}) + \lambda_r \Sigma_{l=1}^{L} R(\mathbf{W}^{(l)}), \tag{2}$$

where $R(\cdot)$ is the group sparsity regularization on each layer. One such candidate for the sparse regularization is the l_1-norm. However, such non-grouped sparsity regularization addresses each neuron in a layer independently, often producing inconsistent connectivity in a deep network. This can produce an irregular memory access pattern, which adversely impacts the practical acceleration performance of the hardware platform [20]. Therefore, we use the *group lasso* instead for regularization, which can be represented as

$$R(\mathbf{W}^{(l)}) = \Sigma_{g=1}^{G} \|\mathbf{W}_g^{(l)}\|_F = \Sigma_{g=1}^{G} \sqrt{\Sigma_{i=1}^{|\mathbf{W}_g^{(l)}|} (w_{g,i}^{(l)})^2}, \tag{3}$$

where $\|\mathbf{A}\|_F$ is a Frobenius norm of $\mathbf{A}$, $\mathbf{W}_g^{(l)}$ is a group of partial weights in $\mathbf{W}^{(l)}$, and G is the total number of groups. Minimizing (3) has the effect of reducing the number of non-zero groups, which can help in reducing the intrinsic complexity of the model and prevent random access to memory.

There are several options for penalizing unimportant neurons by learning the "structure", which is decided by how the groups of $\mathbf{W}_g$ are defined. In this paper, we formulate it as filter-wise group sparsity because the reduced computation is proportional to the percentage of the removed filter [20]. This has an effect of automatically deciding the number of channels used in each layer. Notably, zeroing out a filter in the l-th layer results in a zero-output feature map, which in turn makes a corresponding channel in the $(l+1)$-th layer inactive as shown in Figure 2.

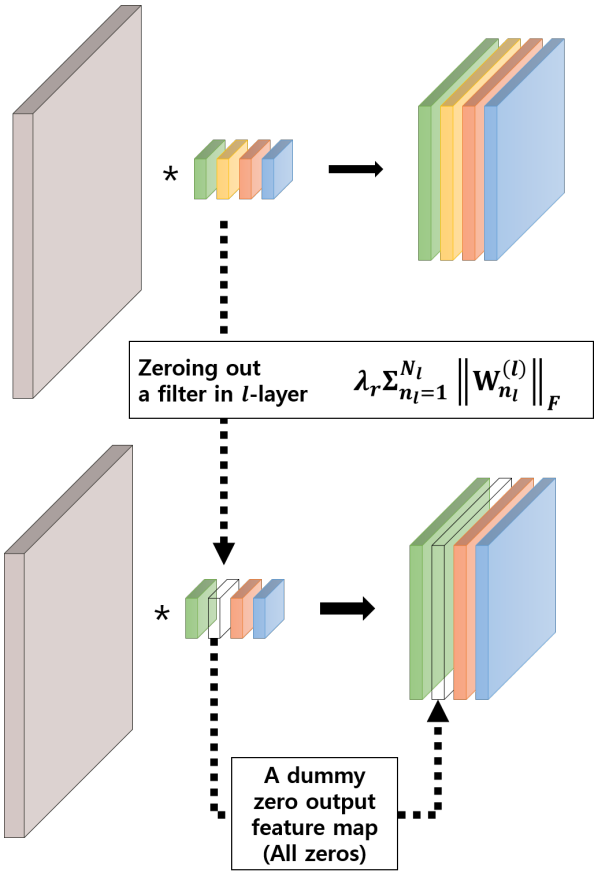


Figure 2. A graphical illustration of a filter-wise group sparsity regularization. Notably, zeroing out a filter in the l-th layer results in a dummy zero output feature map, which in turn makes a corresponding channel in the $(l+1)$-th layer useless.

Let us assume that $\mathbf{W}_{n_l}^{(l)}$ is the n_l-th filter in the l-th layer. Then, the objective function with the filter-wise group sparsity is defined as

$$\arg\min_{\mathbf{W}} \mathcal{L}_{KD}(\mathbf{W}) + \lambda_r \Sigma_{l=1}^{L} \left(\Sigma_{n_l=1}^{N_l} \| \mathbf{W}_{n_l}^{(l)} \|_F \right). \tag{4}$$

As can be seen in (4), this objective contains both the accuracy (including KD) and sparsity terms. It can be solved using the proximal gradient descent method [36]. This particular method is considered efficient for optimizing non-smooth functions using proximity operators [36]. In our case, the proximity operator can be represented as

$$prox_{GL}(\mathbf{W}^{(l)}; \lambda_r) \triangleq \arg\min_{\mathbf{W}^{(l)}} R(\mathbf{W}^{(l)}) + \frac{1}{2\lambda_r} \| \mathbf{W}^{(l)} - \widehat{\mathbf{W}}^{(l)} \|_F^2, \tag{5}$$

where λ_r is the regularization parameter and $\widehat{\mathbf{W}}^{(l)}$ is the intermediate solution of $\mathbf{W}^{(l)}$ after taking a gradient step computed only on the accuracy term, i.e., $\mathcal{L}_{KD}(\mathbf{W})$, in (4). Thereafter, the proximity operator optimizes for the sparsity term by performing the Euclidean projection of the intermediate solution into the solution space. When $R(\mathbf{W}^{(l)})$ in (4) is a group sparsity regularizer, the above problem has a closed-form solution [37]:

$$\mathbf{W}_{n_l}^{(l)*} = \left(1 - \frac{\lambda_r}{\| \widehat{\mathbf{W}}_{n_l}^{(l)} \|_F} \right)_+ \widehat{\mathbf{W}}_{n_l}^{(l)}, \tag{6}$$

where n_l is the index of a group, i.e., a filter in the l-th layer in our case. As one can see in (6), the regularization parameter λ_r plays a role in deciding how many zero groups are included in the solution. Greater the λ_r, the more sparse the solution. Because there is the trade-off between sparsity and accuracy, the method of selecting the value of λ_r is important, but unfortunately, it is still an open question.

To overcome this difficulty, we focus on the teacher-student relationship. There is a chance that the accuracy loss from a teacher network in (4) can be used as a hint to control the regularization parameter, as explained in the next section. Intuitively, if the student model is significantly different in accuracy from the teacher, it may be more important to focus on minimizing the accuracy loss in the training process rather than enforcing the student network to be smaller. Conversely, if the student model is closer in accuracy to the teacher model, we can focus on making a smaller network. To reflect this intuition, we utilize the proportional control theory explained in the next section.

3.3. Proportional Control of Group Sparsity Regularization

The objective of developing a control model based on the control theory is to find a control action that stabilizes a continuously operating dynamical system. To achieve this, a feedback control loop is designed based on the difference, i.e., error signal, between the desired value (set point, SP) and the measured value (process value, PV) [24]. Following the control theory, we set the SP and PV to the loss values obtained from the teacher and student models, respectively. Then, the loss function for training and the update equation for the control variable k become

$$\mathcal{L}_{KD}(\mathbf{W}) + \exp(-k) \cdot \lambda_r \Sigma_{l=1}^{L} R(\mathbf{W}^{(l)}), \tag{7}$$

$$k \leftarrow k + \lambda_k(\gamma \mathcal{H}_S(\mathbf{y}_{true}, \mathbf{p}_S) - \mathcal{H}_T(\mathbf{y}_{true}, \mathbf{p}_T)), \tag{8}$$

respectively, where $\mathcal{H}(\cdot)$ is the cross-entropy between the output of a network and the label, as explained in Section 2.1. We introduce a hyper-parameter $\gamma \in [0, 1]$, which is a relaxation variable to set the goal of the student slightly less than the performance of the teacher. This concept allows us to balance the effort allocated to the sparsity and the accuracy so that neither wins over the other [38]. Algorithm 1 describes the actual procedure for training the student and adjusting k. It is noteworthy that the algorithm is described assuming that only a single sample is used in each iteration; but in reality, each iteration is based on a mini-batch. Moreover, the adjustment of k is done only once in every epoch to prevent any instability in the training procedure. The values of $(\gamma \mathcal{H}_S(\mathbf{y}_{true}, \mathbf{p}_S) - \mathcal{H}_T(\mathbf{y}_{true}, \mathbf{p}_T))$ calculated from all mini-batches are averaged for each epoch to update the variable k. Figure 3 shows a graphical illustration of the feedback loop for adjusting the regularization parameter.

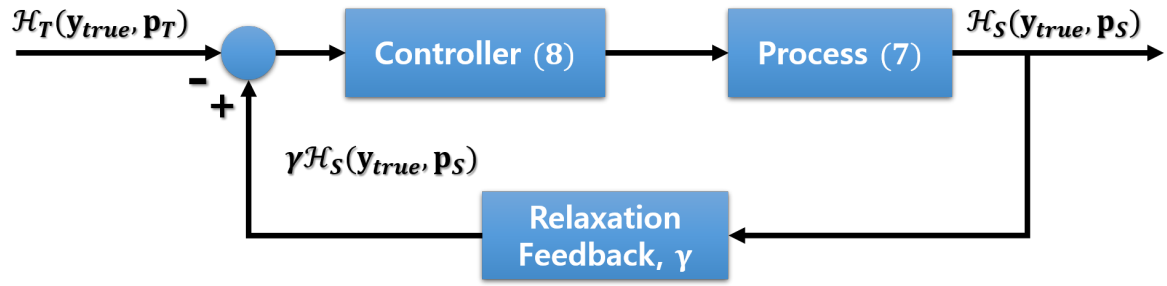


Figure 3. A graphical illustration of the feedback loop for adjusting the regularization parameter λ_r.

Algorithm 1 Optimization with a closed-loop feedback controller.

INPUT: $\mathbf{W}$, λ_r, k, γ, learning rate η
Initialize $\mathbf{W}$.
for each epoch **do**
  for each iteration **do**
    for each layer l of a sparsity target **do**
      $\widehat{\mathbf{W}}^{(l)} \leftarrow \mathbf{W}^{(l)} - \eta \nabla \mathcal{L}_{KD}(\mathbf{W}^{(l)})$
      $\mathbf{W}^{(l)} \leftarrow prox_{GL}(\widehat{\mathbf{W}}^{(l)}; \eta \exp(-k) \cdot \lambda_r)$
    end for
  end for
  Update the control variable k using (8)
end for

In essence, this can be thought of as a form of closed-loop feedback control implemented using a variable k, thereby allowing us to control the amount of emphasis put on the sparsity cost during the gradient descent. We initialize $k = 0$. λ_k is the proportional gain for k. In machine learning terms, it is the learning rate for k [38]. The regularization parameter should be changed slowly, so we use an integrated control value as shown in (8), and the controller is designed as $\exp(-k)$ to ensure that the regularization parameter is non-negative. This control, the same as our intuition described earlier, is as follows.

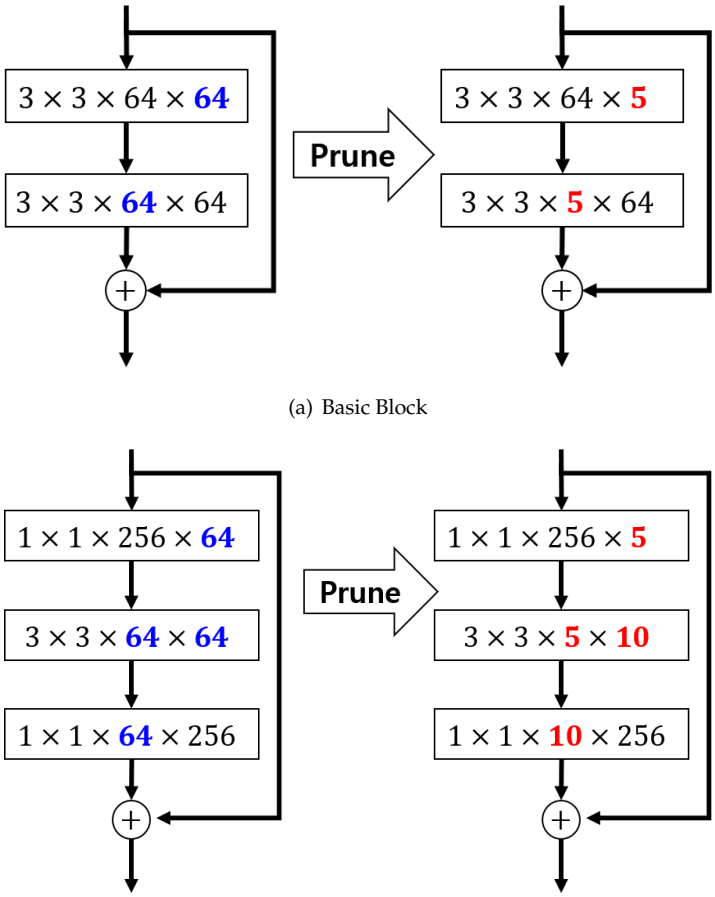
- At the start, k is set to zero, and the control value $\exp(-k)$ is set to one so that the regularization parameter is λ_g.
- If the performance of the student model is estimated to be worse than that of the teacher during the training, i.e., the cross-entropy value (the feedback value) from the student network is larger than that of the teacher, $\exp(-k)$ is reduced to focus more on performance enhancement.
- However, if the student's performance is not bad compared with that of the teacher, i.e., $\gamma \mathcal{H}_S(\mathbf{y}_{true}, \mathbf{p}_S) - \mathcal{H}_T(\mathbf{y}_{true}, \mathbf{p}_T)$ is near zero, the value of $\exp(-k)$ is maintained at the same level, so that the relative strength for sparsity terms remains more or less the same. If the performance of the student is estimated to be better than that of the teacher, $\exp(-k)$ is controlled so that the overall sparsity is increased.

4. Experiments

4.1. Implementation and Environment Details

All the experiments were performed using residual networks with three modules as used for analysis in [8], wherein the network inputs are 32×32 images with per-pixel mean subtraction. To measure the accuracy for a wide range of sparsity levels, we conducted the experiments by setting λ_r in the following range: 0.0001×1.5^i where $i = 6, 7, \ldots, 21$ (Originally, the range of i was $0, 1, \ldots, 21$ but a small i did not have much significance in our experiments.), and the λ_r values ranged roughly between 0.001 and 0.5. We did not apply the sparsity constraint to the last convolutional layer in each residual block as was done in [39] because the residual block has a shortcut connection as shown in Figure 4. In other words, to perform the sum operator, the last convolutional layer and the projection shortcut layer must have the same number of output feature maps. (see [8] for more details). Throughout the experiments, sparsity was measured by the number of zero-valued parameters (after training) divided by the total number of parameters. We used random crops and horizontal flips for data augmentation and normalized an input image based on the mean (0.4914, 0.4822, 0.4465) and the variance (0.2023, 0.1994, 0.2010). Following [12], the number of temperature values, i.e., τ, of the

KD loss were fixed at three in all experiments. We used SGD with Nesterov momentum and set the momentum at 0.9. The learning rate η was initially set at 0.1 and was then dropped to 0.01 and to 0.001 at the half and the 3/4 points, respectively, of the entire training session. The momentum used in the study was 0.9 and the weight decay was set at 0.0001.



(a) Basic Block

(b) Bottleneck Block

Figure 4. An example of a pruning ResNet. The red values are the number of remaining filters/channels.

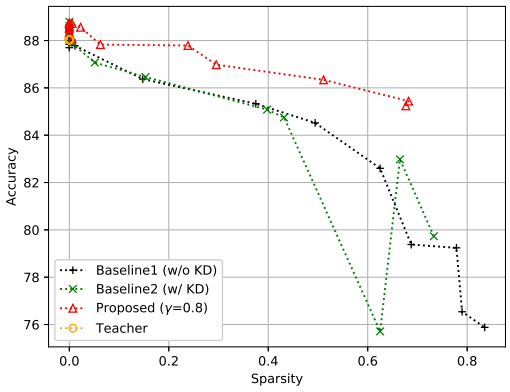
4.2. Evaluation of the Proposed Strategies

This section demonstrates the effectiveness of the proposed method through experiments. The experiments were performed on CIFAR-10, CIFAR-100 [28], and ImageNet32×32 datasets [29] using residual networks with various depths.

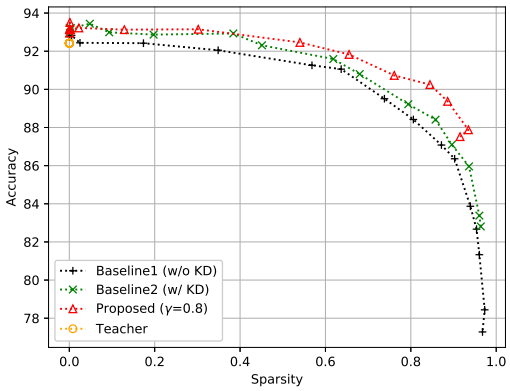
4.2.1. CIFAR-10 Dataset

CIFAR-10 is an image classification dataset with 10 classes, and comprises 50k training images and 10k test images with 32 × 32 resolution. For this dataset, training was performed with 128 batch size, and all the compared methods including the baseline models and the teacher model were trained over 150 epochs. The experiment was designed to verify that the proposed control-based sparsity regularization has better results in the expected trade-off of accuracy and efficiency compared with the baseline methods. The first baseline method uses only sparse regularization without KD, which is denoted as Baseline1 (w/o KD) in the figures, and the second baseline method includes both KD and sparse regularization but does not have the proposed feedback control, which is denoted as Baseline2 (w/ KD) in the figures. With these baseline settings, we conducted two experiments by changing the depth of the networks, i.e., to eight and 32, as shown in Figure 5. As expected, we can

observe that the methods that include KD tend to give more accurate solutions than the sparsity-only methods. However, Figure 5a shows that the combination of KD and sparsity losses (Baseline 2) may have unstable results when there is no control. Conversely, both Figure 5a and Figure 5b show that the proposed approach involving a feedback control consistently gives more accurate results than the baselines and the results are stable even with the change of sparsity level. As reported in recent literature, when the sparsity of the network is close to zero, we can observe that KD helps to achieve better performance than the teacher.



(a) depth 8

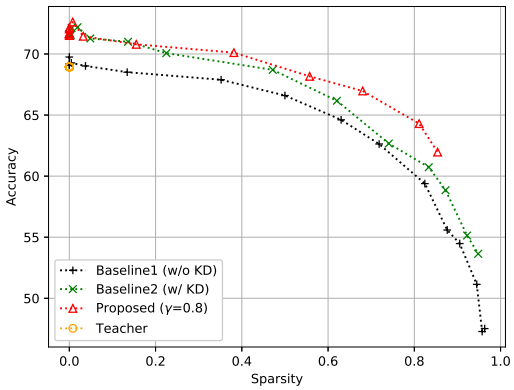


(b) depth 32

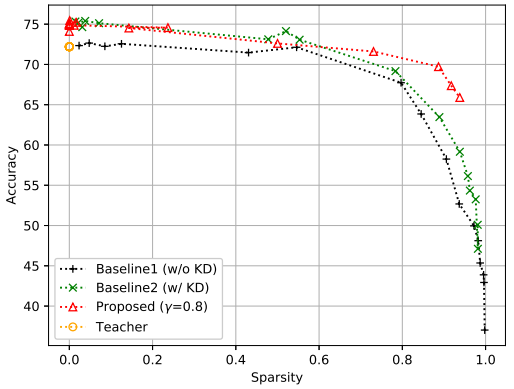
Figure 5. Effect of the proposed strategies on CIFAR-10. Here, the hyper-parameter γ is the relaxation variable shown in Equation (8).

4.2.2. CIFAR-100 Dataset

We conducted experiments similar to CIFAR-10 on the CIFAR-100 dataset also. This dataset also consists of 50k training images and 10k test images with 32×32 resolution, but has 100 classes instead of 10. Our experimental batch size for training was 128, and all the compared methods, including the teacher model, were trained over 150 epochs. The depth of the networks was either 32 or 50 in this experiment. We used basic blocks for depth 32 models, and bottleneck blocks for depth 50 models, as shown in Figure 4. It is evident that the proposed method still gives better accuracy than the two baselines as shown in Figure 6. In addition, the sparser the model is, the more the improvement that can be observed under the proposed control scheme. This means that the proposed feedback control helps find a better solution by controlling k in the optimization process.



(a) depth 32



(b) depth 50

Figure 6. Effect of the proposed strategies on CIFAR-100. Here, the hyper-parameter γ is the relaxation variable shown in Equation (8).

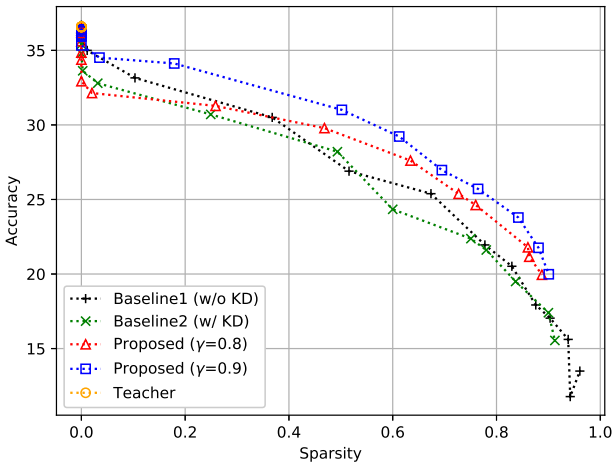
4.2.3. ImageNet32×32 Dataset

Similar to the previous datasets, experiments were conducted on ImageNet32×32 [29]. ImageNet32×32 is a 32×32-downsampled version of the ImageNet dataset [40]. This dataset is a classification dataset with 1000 classes and comprises 1,281k training images and 50k validation images. Training was performed with a batch size of 256, and all the compared methods, including the teacher model, were trained over 40 epochs. We set the depth to 32 and compared the top-1 and top-5 performances of the proposed method with the two baselines. For this experiment, KD-based methods showed slightly worse performance than non-KD-based methods when sparsity was near zero, regardless of whether feedback control was applied or not, as shown in Figure 7. Unlike in the cases of the CIFAR-10 and CIFAR-100 datasets, the teacher network on the ImageNet32×32 dataset did not achieve sufficient accuracy in the top-5, which means that the students should focus more on accuracy than on sparsity. This can be achieved by adjusting γ, i.e., increasing this variable value will set the accuracy goal of the student higher. As expected, you can see that the accuracy of the proposed method improved over that of the baselines when γ was increased.

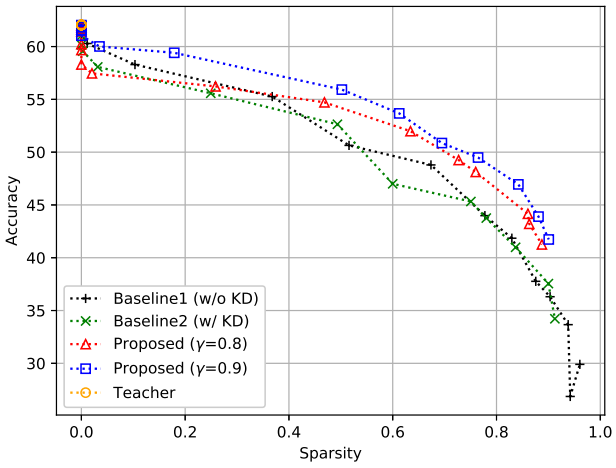
4.3. Analysis of Hyper-Parameter γ

We evaluated the change in performance according to the change in γ on the CIFAR-10 and CIFAR-100 datasets [28]. Here, the depths were set to eight and 32 for CIFAR-10 and CIFAR-100, respectively. Figure 8 shows the results, and we can see a γ value of 1.0 does not always give the best

accuracy, as shown in Figure 8b. As intended, a smaller γ allows the student to avoid putting too much emphasis on improving its accuracy, and we chose γ of 0.8 through two experiments.

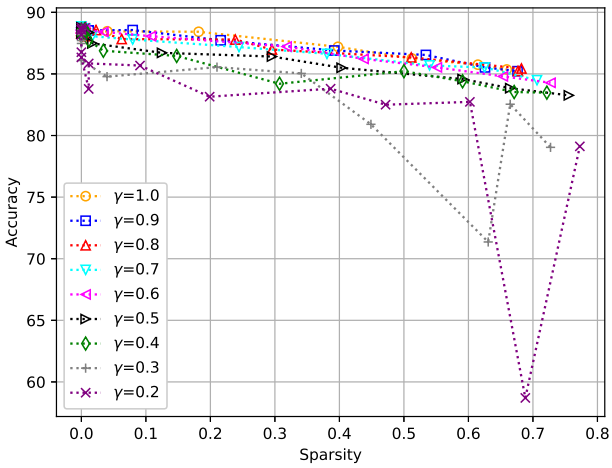


(a) top-1

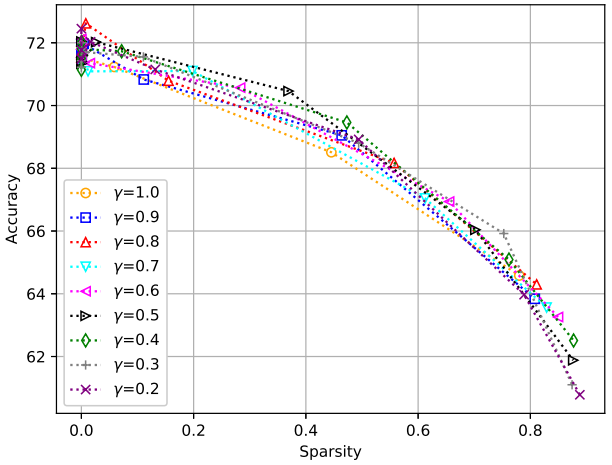


(b) top-5

Figure 7. Effect of the proposed strategies on ImageNet32×32. Here, the hyper-parameter γ is the relaxation variable shown in Equation (8).



(a) CIFAR-10, 8 depths



(b) CIFAR-100, 32 depths

Figure 8. Analysis of the hyper-parameter γ. Here, the hyper-parameter γ is the relaxation variable shown in Equation (8).

5. Conclusions

In this paper, we investigated the effects of two strategies for building a compact network. We demonstrated that (1) incorporating knowledge distillation when compressing the network with group sparsity achieves better performance than in other cases: a student network trained by the proposed strategies achieved better accuracy than when trained by a model with the same sparsity. (2) Moreover, it was observed that the second strategy based on proportional control can adjust the level of sparse regularization, thereby leading to significantly better results. The proposed strategies can make the application of deep networks more practical on intelligent sensors in resource-constrained platforms.

Author Contributions: J.C. conceived the idea, and he designed and performed the experiments; M.L. refined the idea; J.C. and M.L. wrote the paper.

Funding: This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. NRF-2019R1F1A1058666)

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yang, H.; Li, J.; Shen, S.; Xu, G. A Deep Convolutional Neural Network Inspired by Auditory Perception for Underwater Acoustic Target Recognition. *Sensors* **2019**, *19*, 1104. [CrossRef] [PubMed]
2. Hong, J.; Cho, B.; Hong, Y.W.; Byun, H. Contextual Action Cues from Camera Sensor for Multi-Stream Action Recognition. *Sensors* **2019**, *19*, 1382. [CrossRef] [PubMed]
3. Xu, S.; Tang, Q.; Jin, L.; Pan, Z. A Cascade Ensemble Learning Model for Human Activity Recognition with Smartphones. *Sensors* **2019**, *19*, 2307. [CrossRef] [PubMed]
4. Chen, Y.; Tao, J.; Wang, J.; Chen, X.; Xie, J.; Xiong, J.; Yang, K. The Novel Sensor Network Structure for Classification Processing Based on the Machine Learning Method of the ACGAN. *Sensors* **2019**, *19*, 3145. [CrossRef] [PubMed]
5. Shi, Y.; Wang, Y.; Zhao, L.; Fan, Z. An Event Recognition Method for Φ-OTDR Sensing System Based on Deep Learning. *Sensors* **2019**, *19*, 3421. [CrossRef] [PubMed]
6. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.
7. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In Proceedings of the IEEE conference on computer vision and pattern recognition, Madison, WI, USA, 26 June–1 July 2016.
8. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Madison, WI, USA, 26 June–1 July 2016.
9. Yoon, J.; Hwang, S.J. Combined Group and Exclusive Sparsity for Deep Neural Networks. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017.
10. Han, S.; Liu, X.; Mao, H.; Pu, J.; Pedram, A.; Horowitz, M.A.; Dally, W.J. EIE: Efficient Inference Engine on Compressed Deep Neural Network. In Proceedings of the ACM/IEEE Annual International Symposium on Computer Architecture, Seoul, Korea, 18–22 June 2016.
11. Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. SqueezeNet: AlexNet-Level Accuracy with 50x Fewer Parameters and< 0.5 MB Model Size. *arXiv* **2016**, arXiv:1602.07360
12. Hinton, G.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. *arXiv* **2015**, arXiv:1503.02531
13. Romero, A.; Ballas, N.; Kahou, S.E.; Chassang, A.; Gatta, C.; Bengio, Y. Fitnets: Hints for Thin Deep Nets. *arXiv* **2014**, arXiv:1412.6550.
14. Zagoruyko, S.; Komodakis, N. Paying More Attention to Attention: Improving the Performance of Convolutional Neural Networks via Attention Transfer. *arXiv* **2016**, arXiv:1612.03928.
15. Yim, J.; Joo, D.; Bae, J.; Kim, J. A Gift from Knowledge Distillation: Fast Optimization, Network Minimization and Transfer Learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
16. Wang, X.; Zhang, R.; Sun, Y.; Qi, J. KDGAN: Knowledge Distillation with Generative Adversarial Networks. In Proceedings of the Advances in Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018.
17. Lan, X.; Zhu, X.; Gong, S. Knowledge Distillation by On-the-Fly Native Ensemble. In Proceedings of the Advances in Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018.
18. Zhou, H.; Alvarez, J.M.; Porikli, F. Less is More: Towards Compact CNNs. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016.
19. Alvarez, J.M.; Salzmann, M. Learning the Number of Neurons in Deep Networks. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016.
20. Wen, W.; Wu, C.; Wang, Y.; Chen, Y.; Li, H. Learning Structured Sparsity in Deep Neural Networks. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016.
21. He, Y.; Zhang, X.; Sun, J. Channel Pruning for Accelerating Very Deep Neural Networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
22. Daubechies, I.; Defrise, M.; De Mol, C. An Iterative Thresholding Algorithm for Linear Inverse Problems with a Sparsity Constraint. *Commun. Pure Appl. Math. A J. Issued Courant Inst. Math. Sci.* **2004**, *57*, 1413–1457. [CrossRef]

23. Simon, N.; Friedman, J.; Hastie, T.; Tibshirani, R. A Sparse-Group Lasso. *J. Comput. Graph. Stat.* **2013**, *22*, 231–245. [CrossRef]
24. DiStefano, J.J.; Stubberud, A.R.; Williams, I.J. *Feedback and Control Systems*; McGraw-Hill: New York, NY, USA, 1967.
25. Bequette, B.W. *Process Control: Modeling, Design, and Simulation*; Prentice Hall: Upper Saddle River, NJ, USA, 2003.
26. Control Theory—Wikipedia, The Free Encyclopedia. Available online: https://en.wikipedia.org/wiki/Control_theory (accessed on 18 June 2018).
27. Proportional Control—Wikipedia, The Free Encyclopedia. Available online: https://en.wikipedia.org/wiki/Proportional_control (accessed on 18 June 2018).
28. Krizhevsky, A.; Hinton, G. Learning Multiple Layers of Features from Tiny Images. Available online: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.222.9220&rep=rep1&type=pdf (accessed on 3 October 2019).
29. Chrabaszcz, P.; Loshchilov, I.; Hutter, F. A Downsampled Variant of Imagenet as an Alternative to the Cifar Datasets. *arXiv* **2017**, arXiv:1707.08819.
30. Galatsanos, N.P.; Katsaggelos, A.K. Methods for Choosing the Regularization Parameter and Estimating the Noise Variance in Image Restoration and Their Relation. *IEEE Trans. Image Process.* **1992**, *1*, 322–336. [CrossRef] [PubMed]
31. Zhang, K.; Zhe, S.; Cheng, C.; Wei, Z.; Chen, Z.; Chen, H.; Jiang, G.; Qi, Y.; Ye, J. Annealed Sparsity via Adaptive and Dynamic Shrinking. In Proceedings of the International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016.
32. Dong, W.; Zhang, L.; Shi, G.; Wu, X. Image Deblurring and Super-Resolution by Adaptive Sparse Domain Selection and Adaptive Regularization. *IEEE Trans. Image Process.* **2011**, *20*, 1838–1857. [CrossRef] [PubMed]
33. Larsen, J.; Svarer, C.; Andersen, L.N.; Hansen, L.K. Adaptive Regularization in Neural Network Modeling. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016.
34. Leung, C.T.; Chow, T.W.S. Adaptive Regularization Parameter Selection Method for Enhancing Generalization Capability of Neural Networks. *Artif. Intell.* **1999**, *107*, 347–356. [CrossRef]
35. Cho, J.; Kwon, J.; Hong, B.W. Adaptive Regularization via Residual Smoothing in Deep Learning Optimization. *IEEE Access* **2019**, *7*, 122889–122899. [CrossRef]
36. Parikh, N.; Boyd, S. Proximal Algorithms. *Found. Trends® Optim.* **2014**, *1*, 127–239. [CrossRef]
37. Cong, Y.; Yuan, J.; Liu, J. Sparse Reconstruction Cost for Abnormal Event Detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, Colorado Springs, CO, USA, 20–25 June 2011.
38. Berthelot, D.; Schumm, T.; Metz, L. Began: Boundary Equilibrium Generative Adversarial Networks. *arXiv* **2017**, arXiv:1703.10717.
39. Lin, S.; Ji, R.; Li, Y.; Deng, C.; Li, X. Towards Compact ConvNets via Structure-Sparsity Regularized Filter Pruning. *arXiv* **2019**, arXiv:1901.07827.
40. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A Large-Scale Hierarchical Image Database. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009.