An Improved Neural Network Pruning Technology for Automatic Modulation Classification in Edge Devices

Yun Lin, *Member, IEEE*, Ya Tu, *Student Member, IEEE*, and Zheng Dou, *Member, IEEE*

***Abstract*—Automatic modulation classification (AMC) plays an important role in both civilian and military applications. Today, increasingly more researchers apply a deep learning framework in AMC. However, few papers take into account that a typical deep model is difficult to deploy on resource constrained devices. In this paper, we propose a new filter-level pruning technique based on activation maximization (AM) that omits the less important convolutional filter. Compared to other network pruning techniques, the convolutional neural network pruned *via* the AM method achieves equal or higher classification accuracy in the RadioML2016.10a dataset.**

***Index Terms*—Automatic modulation classification, deep learning, edge device, network pruning.**

I. Introduction

AUTOMATIC modulation classification (AMC) was first arose when it was applied in military scenarios, in which people are required to prepare for jamming signals, identify adversary transmitting units, and recover the intercepted signal. Nowadays, scholars are continuously finding new possible applications for AMC in both military and civilian scenarios [1]. It is widely recognized that deep learning (DL) has achieved state-of-the-art performance in many fields, such as computer vision and natural language processing. Researchers have also studied the potential use of DL in signal processing. Tang *et al.* [1] converted raw I/Q data into contour stellar images, and directly used generative adversarial networks (GANs) to transfer the data augmentation trick to the signal process, which improved the accuracy of AMC. To achieve higher classification accuracy, Wang *et al.* [2] replaced pooling operation with dropout in convolutional neural network (CNN). Tu *et al.* [3] exploited GANs to conduct semi-supervised learning experiments on AMC, and their results revealed that GANs with three-way data will outperform the traditional supervised learning method, which only uses labeled data. Zhou *et al.* [4] explored the possibility of applying a DL framework to fog computing enabled mobile communication networks (FogMNW). Peng *et al.* [5] represented modulated signals in data formats with grid-like topology for the CNN and obtained higher classification accuracy than traditional method. Tang *et al.* [6] proposed a new, real-time DL-aided intelligent network traffic control method. However, none of these methods considered the deployment in devices constrained by computational capabilities and storage resource, such as mobile phones or embedded gadgets.

Manuscript received May 29, 2019; revised September 12, 2019, December 9, 2019, and March 9, 2020; accepted March 10, 2020. Date of publication March 27, 2020; date of current version May 14, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61771154 and in part by the Fundamental Research Funds for the Central Universities under Grant HEUCFG201830. The review of this article was coordinated by Prof. Y. Zhou. *(Corresponding author: Yun Lin.)*

The authors are with the College of Information and Communication Engineering, Harbin Engineering University, Harbin 150001, China (e-mail: linyun@hrbeu.edu.cn; tuya@hrbeu.edu.cn; douzheng@hrbeu.edu.cn).

Digital Object Identifier 10.1109/TVT.2020.2983143

To address this deficiency, neural network prune technology [7] is a way to reduce network computational complexity and storage cost. In this paper, we introduce a new criterion based on activation maximization (AM) to measure the activation information in DL for AMC. Compared to the methods used in studies involving weight sum (WS) [7] and average percentage of zeros (APoZ) [8], the experimental results in this study demonstrate that the AM method can effectively and precisely pick out neural network redundant information, which leads to better accuracy in different signal to noise ratio (SNR) at the same compression rate (CR).

The remainder of this paper is organized as follows: Section II briefly discusses neural network pruning technology. Section III introduces our proposed method. Then, an experiment is conducted in Section IV to prove that AM is more suitable for neural network pruning in AMC, and to determine how neural network pruning techniques influence the forward propagation time of edge devices. Finally, a conclusion to our proposed method is provided in Section V.

II. Related Work

A. Convolutional Filter Pruning Process

The convolutional filter pruning method attempts to remove unnecessary information by assuming that a smaller norm parameter or feature plays a less informative role in the inference time [9]. The main pruning process can be divided into the following three stages:

1) Training From Scratch: At this stage, the CNN will learn useful features from a dataset, which serves as the starting point for the pruned CNN model.

2) Network Pruning: This is the core operation of the neural network pruning technique. We should choose neural networks neurons according to the different pruning granularities, and evaluate whether it is crucial for classification due to different evaluation criteria. Different pruning methods have different criteria, such as describe in previous studies [7], [8].

3) Finetune: The neural network pruning operation will inevitably impair the network accuracy after prune processing. Therefore, finetuning is a necessary process to recover the classification accuracy [7], [8].

B. Weight Sum Convolutional Filter Pruning Method

Li *et al.* [7] proposed the criterion based on the absolute sum of each filter. The main premise of this method is that filters with smaller kernel weight are more likely to produce weaker activation. The importance score of each filter can be calculated as follows:

$$s_i = |Weight(i)|, \tag{1}$$

where s_i denotes the importance score of the i-th convolutional filter.

All connections with weights below a certain threshold will be removed from the network.

C. APoZ Convolutional Filter the Pruning Method

This criterion proposed by Hu *et al.* [9] measures each channel's output activation sparsity after rectified linear unit (ReLU) mapping. The importance score of each filter can be calculated as follows:

$$\text{APoZ}(O_c^{(m)}) = \frac{\sum_a^N \sum_b^M f(O_{(c,b)}^{(m)}(a) = 0)}{MN}, n \tag{2}$$

where $O_c^{(m)}$ refers to the output of the c-th convolutional filter channel in the m-th layer, $O_{(c,b)}^{(m)}(a)$ denotes the corresponding a validation image output and b neuron of the c-th convolutional channel in the m-th layer, M represents the dimension of the output channel of $O_c^{(m)}$, and N is the total number of validation examples, $f(*)$ is a indicator function, where $f(*) = 1$ if true, and $f(*) = 0$ if false. The filters with higher APoZ are considered to be less activated, and will be pruned before finetuning.

III. Activation Maximization-Based Deep Neural Network Pruned for AMC

AM is proposed to visualize the preferred inputs of the neurons in each layer. The preferred input can indicate what features a neuron has learned. The learned features are represented by a synthesized input pattern that can cause maximal activation of a neuron. In order to synthesize such an input pattern, each pixel of the CNN's input is iteratively changed to maximize the activation of the neuron.

The fundamental algorithm of the AM can be viewed as synthesizing a pattern image x^*, which maximizes the activation of a target neuron. This process can be formulated as follows:

$$x^* = \arg\max_x h_{m,n}(\theta, x), \tag{3}$$

where $h_{m,n}(\theta, x)$ denotes the activation of a given convolutional filter n from a given layer m in the network, and θ is the parameter of the CNN.

From a given layer m in the network, the AM process can be divided into three stages:

1) Input Image Setting: A random pixel value image $x = x_0$ is set to be the start point for activation computation.

2) Gradient Computation: The CNN parameter is fixed and the gradients are computed with respect to the noise image $\frac{\partial a_{m,n}(x)}{\partial x}$ via backpropagation.

3) Image Pixel Updating: Guided by the direction of the gradient $\frac{\partial a_{m,n}(x)}{\partial x}$, every pixel is updated iteratively to maximize the neuron activation. This stage can be formulated as follows:

$$x \leftarrow x + \alpha \frac{\partial a_{m,n}(x)}{\partial x}, \tag{4}$$

where α denotes the gradient ascent step size.

In order to stabilize the numerical calculations, we use the normalization trick, which is:

$$\frac{\partial a_{m,n}(x)}{\partial x} = \frac{\frac{\partial a_{m,n}(x)}{\partial x}}{\sqrt[2]{\left\| \frac{\partial a_{m,n}(x)}{\partial x} + \varepsilon \right\|_2}}, \tag{5}$$

where ε is a very small value to prevent the denominator from equaling zero.

In this paper, we calculate the mean value of $\frac{\partial a_{m,n}(x)}{\partial x}$ to represent the importance of the convolutional filter:

$$s_n = \frac{1}{KL} \sum_{k=1}^{K} \left| \sum_{l=1}^{L} \frac{\partial a_{m,n,k,l}(x_k)}{\partial x_k} \right|, \tag{6}$$

where K denotes iterative steps, and L is the dimension of the convolutional filter $a_{m,n}$. Just as the Fisher information matrix [10], the AM method can be a good first-order representation of what a convolutional filter is doing. A large AM gradient indicates a crucial filter with a large impact on the classification, and this is the filter we want to prune the least. A filter that has a zero AM gradient is a "dead filter", which learns nothing from the dataset and can safely be pruned without losing much accuracy.

Algorithm 1: Convolutional Layer Channel Level Pruning Base on AM.

```
Input: An image $x = x_0$ with random pixel values; Select $n$-th
    channel in $m$-th layer $a_{m,n}$; Total channel number $N$;
    Gradient ascent steps $K$; The dimension of convolutional filter
    $L$; Gradient ascent step size $\alpha$; Prune filter number $P$;Training
    set $\{\hat{\boldsymbol{x}}, \hat{y}\}$;
Output: Convolutional Layer Pruned CNN Model
  initialize: CNN parameter fixed;
 1: for $n = 1, 2, ..., N$ do
 2:   for $k = 1, 2, ..., K$ do
 3:     Get image $x_k$ pixel updated from $n$-th channel
 4:     $x_k \leftarrow x_k + \alpha \frac{\partial a_{m,n,k}(x_k)}{\partial x_k}$
 5:   end for
 6:   Compute $s_n = \frac{1}{KL} \sum_{k=1}^{K} | \sum_{l=1}^{L} \frac{\partial a_{m,n,k,l}(x_k)}{\partial x_k} |$
 7: end for
 8: Sort $s_n$ and Find the least P convolutional filter $index$
 9: Remove Channel $(m$,index) from CNN
10: Finetune CNN with $\{\hat{\boldsymbol{x}}, \hat{y}\}$
```

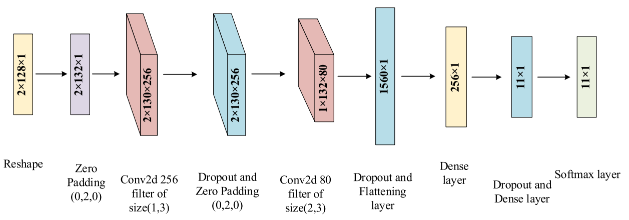

Fig. 1. VTCNN2 architecture.

The entire AM-based deep neural network algorithm pruned for AMC can be formulated as shown in Algorithm 1.

IV. Experiment

A. Dataset

To study the influence of the network pruning techniques on the DL-based AMC method, we used the RML2016.10a dataset [10]. The RML2016.10a dataset is comprised of 220,000 input samples and 11 different modulation types. The samples are generated for 20 different SNR levels from -20 to 20 dB with a stride of 2 dB. Each sample input corresponds to 128 in-phase and 128 quadrature components.

We chose the 4, 0, and -4 dB data from the RML2016.10a dataset to evaluate the performance of our proposed method in different wireless communication SNR environments. A total of 9,000 samples were randomly set to trainset, and the remaining 2,000 samples were considered to be testset per SNR.

B. CNN Architecture

We considered VT-CNN2 to be the CNN classifier [10]. The structure of VT-CNN2 is depicted in Fig. 1, and follows the TensorFlow's default format for data. We used this network in our experiment.s

We used the Adam optimizer in this experiment, and the learning rate was 0.0001. All the activation functions are ReLUs. In addition, we set the gradient ascent step size α to 1 and ε to $1e-7$. The neural network weight initial method was the Xavier initialization.

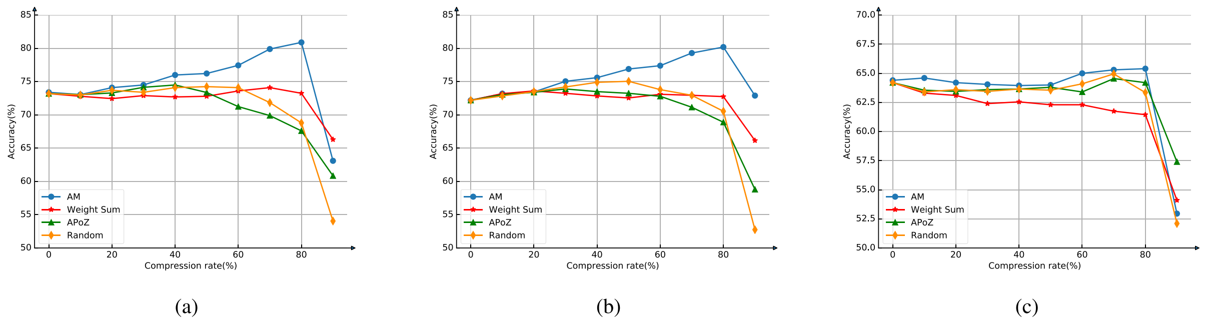

Fig. 2. Performance comparison of different channel selection criteria of the VTCNN2 model pruned on the RML2016.10a dataset with different compression rates (CRs) at different SNRs, and with the sequences: (a) SNR = 4 dB, (b) SNR = 0 dB, and (c) SNR = −4 dB.

C. Accuracy at Different Compression Rates

We applied our algorithm to iteratively prune the convolutional filters in the CONV1 and CONV2 layers. After experimenting with different ways to prune the convolutional filters according to the AM measurements, we found that the performance would drop by a considerable margin when too many filters were empirically trimmed in one step. Therefore, we chose an iterative scheme to trim the network and 5% of the weakest filters were pruned once.CR means the prune rate of convolutional filters in one specified convolutional layer. Since updating the weight of the neural networks is an iterative process, we call it one epoch when an entire dataset is passed forward and backward through the neural network only once. Each pruning is followed by several epochs of finetuning, and the early-stop strategy is set to 30 epochs. For a fair comparison, all the settings except for the convolutional filter criterion are kept constant.

After carrying out the network pruning process described in Section II, we compared the accuracy at different CRs, as shown in Fig. 2.

It is evident from the data in Fig. 2 that the AM criterion achieves an accuracy of about 0.5 to 7% higher than other selection methods under most circumstances. In the convolutional layer filter selection stage, the AM method is an indicator to seek the “death filter” or “near death filter,” which learns almost nothing from the dataset. Every pruning operation will remove unused or less used convolutional filters. After pruning, a finetuning stage will bring in more unused or less used convolutional filters due to the CNN’s sparsity encoding attribute. In this way, AM becomes a better way to conduct neural network pruning process.

In addition, it can be also observed that, the accuracy will increase, to some extent, along with the increase in the CR, and will begin to decline at 80% CR. The reason for this phenomenon is that the VTCNN2 architecture may overparameterize the RML2016.10a dataset and cause an overfitting problem. Network pruning will remove the redundancy in the convolutional layer that grabs ungeneralizable features [11], thereby in return improving the VTCNN2 performance. However, the VTCNN2 has been pruned of so many neurons that it leads to an underfitting problem at 90% CR. In this case, VTCNN2 performance will drop. It is evident from this phenomenon that the neural network pruning technique may be an alternative method to improve the DL-aided AMC.

D. The Amount of Calculation at Different Compression Rates

To verify the usefulness of the proposed network pruning technique, in this section, we will determine the amount of calculation of the VTCNN2 model at different CRs, as well as the real forward propagation time of edge devices. To measure the amount of calculation, we introduce a float point operations (FLOPs) concept. The FLOPs is a more accurate measure than measuring the instructions per second. The VTCNN2 model is primarily comprised of a convolutional layer and a fully connected layer. The FLOPs for the convolutional layer can

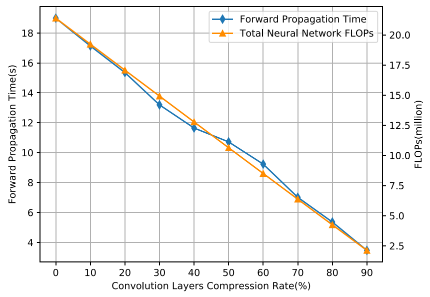

Fig. 3. Forward propagation time changes along with convolutional layer CRs.

be calculated with the following equation [12]:

$$FLOPs = 2HW(C_{in}K^2 + 1)C_{out}, \tag{7}$$

where W, H, and C_{in} are the width, height, and number of the input feature map, respectively; K and C_{out} denotes the kernel width, and the number of output channels, respectively.

For the fully connected layer, the FLOPs can be computed as [12]:

$$FLOPs = (2M - 1)N, \tag{8}$$

where M denotes the input shape and N denotes the input shape.

To simulate the actual computational capability and resource constrained scenario, we choose NVIDIA Jetson TX2 as the blueexperimental device and only used its CPU Quad ARM@ A57/2 MB L2. The forward propagation time is obtained via the VTCNN2 the prediction time of 10,000 samples at different CRs. In order to overcome the random nature of the prediction time, we repeated our measurement 100 times, and report the averaged result and the various FLOPs tendencies in Fig. 3.

As revealed in Fig. 3, the linear decrease in the forward propagation time of the edge device is approximately correlated with linear increase in the CR , which proves the usefulness of the network pruning technique in reducing the requirement of the edge device computational capability. It is also evident that the forward propagation time and FLOPs have a significant positive correlation with CR. This result implies that the convolutional layer accounts the overwhelming majority of the amount of calculation, and that the network pruning process on the convolutional layer will greatly minimize the forward propagation time of the edge device.

V. Conclusion

In this paper, we proposed an AM network pruning criterion for a CNN model acceleration and compression. The AM criterion achieves an accuracy about 0.5 to 7% higher than that with WS, APoZ, and random selection criteria under most circumstances. We also found that the pruned model achieves a higher accuracy than the original accuracy at 80% CR. We believe that the AM criterion may help in getting rid of the ungeneralizable feature, and may serve as an alternative method to improve the DL-aided AMC. In the VTCNN2 model, we found that the convolutional layer is responsible for most of the amount of calculation via FLOPs tendency variation. This phenomenon suggests that we should concentrate on convolutional layer pruning if we want to accelerate our model in edge devices.

Some researchers have found that small-norm parameter may still have a large impact on the loss function. Additionally, when the deviation of the filter norm distributions is too small or filter norm may not be arbitrarily small in practical scenario, it is hard to decide which convolutional filter should be pruned [13]. Thus, an alternative method for better channel selection is also worth studying in the future. The convolutional filter visualization technique also tells us that the functionality of many of these filters may be identical in CNN. This means that we could potentially compress the number of filters used in CNN by a large factor by finding the most representative filter using the cluster techniques.

References

[1] B. Tang, Y. Tu, Z. Zhang, and Y. Lin, “Digital signal modulation classification with data augmentation using generative adversarial nets in cognitive radio networks,” *IEEE Access*, vol. 6, pp. 15713–15722, 2018.

[2] Y. Wang, M. Liu, J. Yang, and G. Gui, “Data-driven deep learning for automatic modulation recognition in cognitive radios,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 4074–4077, Apr. 2019.

[3] Y. Tu, Y. Lin, J. Wang, and J. U. Kim, “Semi-supervised learning with generative adversarial networks on digital signal modulation classification,” *Comput. Mater. Continua*, vol. 55, no. 2, pp. 243–254, May 2018.

[4] Y. Zhou, L. Tian, L. Liu, and Y. Qi, “Fog computing enabled future mobile communication networks: A convergence of communication and computing,” *IEEE Commun. Mag.*, vol. 57, no. 5, pp. 20–27, May 2019.

[5] S. Peng *et al.*, “Modulation classification based on signal constellation diagrams and deep learning,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 99, pp. 1–10, Mar. 2019.

[6] F. Tang *et al.*, “On removing routing protocol from future wireless networks: A real-time deep learning approach for intelligent traffic control,” *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 154–160, Feb. 2018.

[7] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, “Pruning filters for efficient ConvNets,” in *Proc. Int. Conf. Learn Representations*, 2017.

[8] H. Hu, R. Peng, Y. Tai, and C. Tang, “Network trimming: A data-driven neuron pruning approach towards efficient deep architectures,” in *Proc. Int. Conf. Learn Representations*, 2016, pp. 1–9.

[9] J. Kirkpatrick *et al.*, “Overcoming catastrophic forgetting in neural networks,” *Proc. Nat. Acad. Sci.*, vol. 114, no. 13, pp. 3521–3526, Mar. 2017.

[10] T. OShea and J. Hoydis, “An introduction to deep learning for the physical layer,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[11] J. Luo, H. Zhang, H. Zhou, C. Xie, J. Wu, and W. Lin, “ThiNet: Pruning CNN filters for a thinner net,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 10, pp. 2525–2538, Oct. 2019.

[12] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, “Pruning convolutional neural networks for resource efficient inference,” in *Proc. Int. Conf. Learn Representations*, 2017, pp. 1–17.

[13] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang, “Filter pruning via geometric median for deep convolutional neural networks acceleration,” *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4340–4349.