Pruning rPPG Networks: Toward Small Dense Network with Limited Number of Training Samples

Changchen Zhao[1], Pengcheng Cao[2], Shoushuai Xu[3], Zhengguo Li[4], and Yuanjing Feng[2,*]
[1]Hangzhou Innovation Institute, Beihang University, Hangzhou 310053, China
[2]School of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China
[3]Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110169, China
[4]SRO department, Institute for Infocomm Research, Singapore
fyjing@zjut.edu.cn

Abstract

Neural network pruning reduces network complexity and storage by removing unimportant connections in the network, enabling network miniaturization, fast training and inference, easy deployment to portable devices, etc. The emerging lottery ticket hypotheses and sparse initialization technique have shed new lights on the pruning research. However, few research focuses on the pruning of the networks for remote photoplethysmography (rPPG) pulse signal extraction. Opposite to the existing pruning researches that prune large network, rPPG networks are relatively small. It is interesting to see how it behaves when the pruning is applied. In this paper, we investigate the behavior of common pruning techniques when applied to an existing rPPG network. Experiments on PURE dataset show that the pruning rate decay is beneficial to the performance improvement, whereas the connection regeneration has a detrimental effect. Given the same final sparsity, dense initialization generally performs better than sparse initialization. The network seems insensitive to initial sparsity. The combination s_i=1.0, s_f=0.1, with decay, and without regeneration is the best trade-off between SNR and FLOPs, achieving average SNR 9.78 dB, increased by 0.48 dB in comparison with the original PhysNet.

1. Introduction

The measurement of vital signs has revolutionized from contact way to non-contact way, mitigating the inconvenience of physical contact to the human body and broadening the application range. The vital signs that, in previous years, are measured in the hospital can be accessed easily nowadays in daily life. This shift stems from a new technology called remote photoplethysmography (rPPG). The digital camera can capture subtle skin color variations caused by cardiac activities that cannot be perceived by naked eyes. By mean of computer vision technologies and optical/physiological principles, various vital signs can be reliably extracted such as heart rate [14], respiratory rate [6], heart rate viability [10], blood pressure [15], oxygen saturation, etc. The application has stepped out from hospitals to everyday life, *e.g.*, fitness exercise [51], sleep monitoring [18], driver assistance, face anti-spoofing, etc. rPPG has become an active research topic as it is attracting more and more researchers worldwide.

In recent years when deep neural networks are refreshing the record of many computer vision tasks, the measurement accuracy of rPPG has been boosted significantly with the introduction of neural networks in the rPPG pulse extraction model. Existing approaches can be roughly characterized into end-to-end approaches [3, 14, 17, 36, 49] and two-step approaches [28, 32, 35, 35] according to the network architecture. The end-to-end approaches take raw video data as input and output the pulse waveform or heart rate value directly, while the two-step approaches first transform the video into a hand-designed representation and then follow a network. In comparison with conventional signal processing or model based approaches, the performance of neural network based approaches usually have a superior performance.

However, neural networks bring tremendous computational load and storage consumption, making network training and inference a tedious work. It is well known that the network performance is supported by a large amount of data and strong hardware computational power. On the one hand, the optimization of millions of network parameters needs huge amount of training data to prevent overfitting. The commonly used datasets for image classification research are usually large-scale, *e.g.*, ImageNet [37], CIFAR [19], and MNIST [21] consist of 14,197,122, 60,000, and 70,000 images, respectively. It is difficult for rPPG to

collect such huge datasets, which includes recruiting subjects and financial resources. On the other hand, the training and inference of a deep neural network need strong computational power. The training of AlexNet [20] or ResNet [12] often take days or weeks based on multiple graphics processing units (GPUs). This hinders the fast design of rPPG networks. Huge training load brings tremendous energy consumption, which is not a desirable solution in nowadays when carbon dioxide emission problem is becoming increasingly serious [9, 40]. In addition, the large model is difficult to deploy on the application site, especially for the scenarios of rPPG such as fitness equipment, hospital wards, automobile on-board computers, etc., where the computational resource and band-width are very limited.

Neural network pruning is an effective way of reducing training load, inference time, and energy consumption while keeping network performance comparable simultaneously. It miniaturizes the network by cutting *irrelevant* connections in the model [23]. The main issue focuses on keeping or even outperforming the performance of the dense counterpart after pruning. The connections can be pruned according to gradient, weight magnitude, low-rank decomposition, etc. Neural network pruning can be roughly categorized into structured pruning and unstructured pruning [7]. Structured pruning treats a group of neurons, filters, or channels as a whole, enabling fast acceleration and deployment. Unstructured pruning has finer grains, focusing on the individual weights. Unstructured pruning shows more promising potential due to its outstanding performance at extreme sparsities. The rapid development of hardware with sparse operation also provides strong support for the deployment of structured pruning [25].

In the early studies, researchers pointed out that there is a lot of redundancy in the neural network. Deleting the unimportant weight in the network can obtain better generalization ability, less training and inference time [31]. Pioneering works model the pruning task by adding a regularization term, leveraging a second-order Taylor expansion to select parameters for deletion [22]. A line of research prunes the weight in an iterative manner, learning weights and prunes the weights simultaneously [29, 38, 52]. The network architecture is updated dynamically during training, enabling better adapt to the training data. Lee *et al*. [23] proposed single-shot network pruning scheme that prunes a given network once at initialization prior to training, mitigating the limitations of during-training-pruning approaches that employs either heuristically designed pruning schedules or additional hyperparameters. More recently, the lottery ticket hypotheses (LTH) is proposed [8] which highlights the importance of the sparse initialization for a subnetwork with comparative performance to the dense trained network.

Network pruning is a general network slimming approach. Theoretically, any field using neural network can apply network pruning. Unfortunately, to the best of our knowledge, we have not seen any research addressing pruning for rPPG signal extraction networks. In this paper, we conduct a series of experiments to investigate the performance of common pruning techniques when applied on rPPG network. Opposite to the existing pruning researches that the network is large-scale, rPPG network has a relatively small scale. It is interesting to see how it behaves when pruning is applied. We choose one existing good-performing rPPG network as backbone, apply common pruning techniques to the network and observe the performance. This research serves as a first attempt that introduces network pruning to rPPG networks, with an aim to give insights to the design of pruning methods for small networks with limited number of training samples. Moreover, this research can promote the deployment of rPPG networks to portable devices.

2. Related work

2.1. Remote photoplethysmography

Remote photoplethysmography assumes that the captured raw trace from facial video is a combination of the target pulse signal, motion artifacts, and other noises. The main purpose is to extract the target pulse signal out from the observations. In the early studies, researchers employ signal separation techniques to realize noise suppression, *e.g*., independent component analysis (ICA) [34], principal component analysis (PCA) [24], or ensemble empirical mode decomposition (EEMD) [4]. These approaches treat the raw traces as pure signal but do not consider the physiological and optical principles of the imaging process. To address this issue, the skin reflection model is established, which quantitatively models the incident light, specular and diffusion reflection of the skin, and the camera quantization noise. Based on this model, several pulse extraction algorithms are proposed, *e.g*., CHROM [5], PBV [11], and POS [44].

In recent years, neural networks have been introduced into the field of rPPG. Neural network based rPPG extraction approaches usually obtain superior performance than signal processing based approaches due to the powerful ability in spatial-temporal feature extraction and noise suppression. Chen and McDuff [3] designed a convolutional attention network DeepPhys, which is a two-path architecture with one path characterizing motion information and one path characterizing appearance information. Spetlik *et al*. [36] proposed a two-stage network HR-CNN, which consists of an Extractor for rPPG signal extraction from image frames and an Estimator for heart rate prediction. These two approaches are based on 2D or 1D convolution. In order to extract spatial and temporal features simultaneously, Yu *et al*. [49] designed an end-to-end spatio-temporal network

PhysNet, based on which two kinds of spatio-temporal convolutions, *i.e*., 3D convolution and long short term memory (LSTM), are embedded. Huang *et al*. [17] proposed PRNet, which first employs 3D convolution modules for spatial and local temporal feature extraction, followed by LSTM modules for global temporal feature extraction. Hu *et al.* [14] proposed ETA-rPPGNet, in which a time-domain attention mechanism is designed. In this mechanism, the 1-D convolution is used to effectively model the information association in the local time domain, so as to improve the anti-noise ability of the model. In order to mitigate the noise inference induced by motion and illumination changes, Wu *et al*. [45] proposed a framework that includes an error compensation neural network after conventional signal-based rPPG extraction to improve the measuring results.

2.2. Neural network pruning

The most common way is to cut the unimportant weights based on a pre-trained network, which is called post-training pruning. Mozer *et al*. [31] proposed a method of using the knowledge in a network to determine the relevance of individual units. The least relevant units are then be trimmed to construct a skeleton version of the network. LeCun *et al*. [22] used information-theoretic ideas to derive a class of practical and nearly optimal schemes for removing unimportant weights from a network. Molchanov *et al*. [30] prunes the neural networks using Taylor expansion-based criterion that approximates the change in the cost function. You *et al*. [50] considered a new block decomposition algorithm that combines the effectiveness of combinatorial search methods and the efficiency of coordinate descent methods for sparse optimization of a neural network.

During training pruning attempts to adjust network connections in the process of network training instead of waiting for the pre-trained model to be completed. Because the pruning criterion is associated with network status, the topology of the network connections can be dynamically adjusted during the training process, which exhibits more flexibility compared with post-training pruning. Srinivas *et al*. [38] introduced additional gate variables to perform parameter selection and showed that this is equivalent to using a spike-and-slab prior. Zhu and Gupta [52] introduced a gradual pruning algorithm where the pruning rate starts from an initial sparsity and gradually decreases to the target final sparsity. Mocanu *et al*. [29] evolved an initial sparse topology of two consecutive layers of neurons into a scale-free topology during learning.

Before-training pruning is a recently emerged technique that prunes a given network once at initialization prior to training with an aim to reduce the need for either heuristically designed pruning schedules or additional hyperparameters used in post-training pruning. Lee *et al*. [23] proposed single-shot network pruning (SNIP) that employs a saliency criterion based on connection sensitivity that identifies structurally important connections in the network. After pruning, the sparse network is trained in the standard way. Paul *et al*. [33] introduced combined pruning scores (COPS) instead of a single pruning criterion to create more powerful pruning strategies. Wang *et al*. [43] proposed the gradient signal preservation (GraSP) algorithm in order to preserve the gradient flow through the network for more efficient training.

The lottery ticket hypotheses (LTH) [8] stated that dense, randomly-initialized, feed-forward networks contain subnetworks (winning tickets) that, when trained in isolation, reach test accuracy comparable to the original network in a similar number of iterations. However, the identification of the winning ticket still needs heavy train-prune-retrain operation, which limits practical application. Evci *et al*. [7] proposed RigL for training sparse models without the need of a "lucky" initializations. You *et al*. [48] discovered that the winning tickets can be identified at a very early training stage, termed as Early-Bird (EB) tickets, via lowcost training schemes. Chen *et al*. [2] examined several supervised and self-supervised pre-trained models widely used in computer vision using LTH and discovered that core LTH observations remain generally relevant in the pre-training paradigm of computer vision.

2.3. Pruning rPPG networks

To the best of our knowledge, we have not seen published literatures addressing pruning rPPG pulse extraction networks. But the need for reducing computational burden and running on resource limited devices exists. On the one hand, typical rPPG application scenarios, *e.g*., driver monitoring, sleep monitoring, fitness exercise, etc. have hardware with restricted computational power. On the other hand, the rPPG network cannot be designed very deep due to limited number of training samples. Tables 1 and 2 summarize common datasets and algorithms in rPPG, from which one can see that, in comparison with ImageNet or ResNet, rPPG has much smaller datasets and networks. The benefits of introducing neural network pruning into rPPG are two-fold: 1) deeper and larger-scale networks for rPPG pulse extraction can be designed in order to obtain higher accuracy, and 2) the demand for hardware requirements is alleviated.

3. Methodology

3.1. Preliminaries

Sparse distribution refers to the ways of distributing non-zero elements over the layers while maintaining the overall sparsity. The following three sparse distributions are widely used.

1) Uniform. The sparsity of each layer keeps the same

Table 1. Comparison of common networks proposed for rPPG pulse extraction.

Name	Input size	# of layers	# of parameters ($\times 10^6$)	storage (MB)	FLOPs ($\times 10^9$)
DeepPhys [3]	$3 \times 150 \times 36 \times 36$	9	1.46	5.70	9.62
HR-CNN [36]	$3 \times 300 \times 192 \times 168$	13	1.87	7.32	988.97
PhysNet [49]	$3 \times 128 \times 128 \times 128$	15	0.83	3.26	130.52
MTTS-CAN [27]	$3 \times 150 \times 36 \times 36$	9	1.45	5.70	9.61
DeeprPPG [26]	$3 \times 120 \times 128 \times 64$	15	0.54	2.12	26.76
RhythmNet [32]	$3 \times 10 \times 300 \times 25$	21	11.42	44.64	1.70

Table 2. Comparison of selected publicly available datasets used for rPPG research.

Name	# of subjects	# of frames	# of videos	resolution	average duration/video (sec)	storage (GB)
PURE [39]	10	125,366	60	640×480	69.9	38.6
COHFACE [13]	40	202,092	164	640×480	61.6	0.662
ECG-fitness [36]	17	407,232	202	1920×1080	67.2	1044
UBFC-rPPG [1]	42	81,401	42	640×480	64.4	69.8
BUAA-MIHR [46]	13	257,339	143	640×480	59.9	220
NBHR [16]	257	886,001	1130	640×480	32.7	921

as the overall sparsity. Some works [7] keep the first layer dense because the sparsity of the first layer will have a significant effect on the performance but has little effect on reducing the model size.

2) Erdos-Renyi. Named after the famous mathematician Paul Erdos and Alfred Renyi, Erdos-Renyi model is originally proposed for generating random graphs. Given a positive integer n and a probability value $0 \leq p \leq 1$, define the graph $G(n, p)$ to be the undirected graph on n vertices whose edges are chosen as follows. For all pairs of vertices v, w, there is an edge (v, w) with probability p. For network pruning, the sparsity of layer l is set as follows, $s^{(l)} \propto \left(1 - \frac{C_{in}+C_{out}}{C_{in} \times C_{out}}\right)$, where C_{in}, C_{out} denote the number of input and output channels, respectively.

3) Erdos-Renyi-Kernel. ERK modifies the Erdos-Renyi rule by including kernel height and width in the scaling factors so that the sparsity of layer l is associated with not only input and output channels but also height h and width w, *i.e.*, $s^{(l)} \propto \left(1 - \frac{C_{in}+C_{out}+w+h}{C_{in} \times C_{out} \times w \times h}\right)$.

In this paper, we use uniform distribution to keep the method simple. The bias and batch-norm layers are kept dense because it has little effect on reducing the total model size.

Layer-wise pruning vs. global pruning. Layer-wise pruning prunes layer by layer and keeps every layer sparsity equal to the overall sparsity while global pruning puts all parameters from different layers together and sorts with threshold according to the overall sparsity. As a result, the sparsity of each individual layer in global pruning is different and may not be equal to the overall sparsity.

Dense-to-sparse vs. sparse-to-sparse training. The difference between dense-to-sparse and sparse-to-sparse training lies in the initialization of the network. Let s_i be the initial sparsity. Then, $s_i = 1$ denotes using the dense model as the initial network. Sparse-to-sparse training $0 < s_i < 1$ starts from a sparse subnetwork by setting $(1 - s_i) \times 100\%$ parameters of the dense model to zeros.

Gradual pruning. Instead of keeping the pruning rate unchanged over the entire training process, gradual pruning starts with the initial sparsity s_i and decreases the pruning rate gradually to the final sparsity s_f. The decay rule can be polynomial [25], sinusoidal [7], or other forms.

Regeneration. Inspired by the neuroregeneration mechanism of the human brain that connections can be regenerated or synthesized to recover the damage in the nervous system, it is beneficial to regenerate some of the connections being pruned during the pruning process. The number of the connections to be regenerated can be the same as that of the connections being pruned or only a small proportion is regenerated.

Pruning criterion. Pruning is realized by sorting a certain criterion in descending or ascending order and thresholds them according to the target sparsity. Weight magnitude [25] is perhaps the most commonly used criterion. Other criteria include gradient [7], low-rank decomposition [47], Hessian [22], etc. In this paper, we use weight magnitude for cutting the neuron connections and gradient for regeneration.

3.2. Backbone network

PhysNet [49] is chosen as our backbone network for evaluation because it can achieve satisfactory results on various occasions. In this paper, we only use the 3D CNN based version. The network consists of one 2D convolutional block, four 3D convolutional blocks, and one spatial global average pooling layer. The network takes as input the raw video segment of T-frame face images with RGB channels (in this paper, $T = 150$). The 2D convolutional block serves as a preprocess to extract spatial features as representation, and the 3D convolution block takes into account spatial and temporal correlations in the feature tensor simultaneously for more robust feature extraction. The spatial global average pooling layer is used for finalizing the feature tensor to the output with the same size as the ground truth pulse signal. Detailed network specifications are summarized in Table 3.

Table 3. PhysNet architecture specification. The 2D kernel is of size $H \times W$, and the 3D kernel is of size $T \times H \times W$, where C, T, H, W denote channel, time, height, and width, respectively. The dimension of the output size is $C \times T \times H \times W$.

Name	Kernel	Output
Input	none	$3 \times T \times 192 \times 128$
conv2D	5×5	$32 \times T \times 192 \times 128$
maxpooling_1	$1 \times 2 \times 2$	$32 \times T \times 96 \times 64$
conv3D_{11}	$3 \times 3 \times 3$	$64 \times T \times 96 \times 64$
conv3D_{12}	$3 \times 3 \times 3$	$64 \times T \times 96 \times 64$
maxpooling_2	$1 \times 2 \times 2$	$64 \times T \times 48 \times 32$
conv3D_{21}	$3 \times 3 \times 3$	$64 \times T \times 48 \times 32$
conv3D_{22}	$3 \times 3 \times 3$	$64 \times T \times 48 \times 32$
maxpooling_3	$1 \times 2 \times 2$	$64 \times T \times 24 \times 16$
conv3D_{31}	$3 \times 3 \times 3$	$64 \times T \times 24 \times 16$
conv3D_{32}	$3 \times 3 \times 3$	$64 \times T \times 24 \times 16$
maxpooling_4	$1 \times 2 \times 2$	$64 \times T \times 12 \times 8$
conv3D_{41}	$3 \times 3 \times 3$	$64 \times T \times 12 \times 8$
conv3D_{42}	$3 \times 3 \times 3$	$64 \times T \times 12 \times 8$
avgpooling	$1 \times 12 \times 8$	$64 \times T \times 1 \times 1$
conv	$1 \times 1 \times 1$	$1 \times T \times 1 \times 1$

3.3. Pruning rate decay

We follow the common practice of gradual pruning strategy that the pruning rate attenuates gradually to the target sparsity instead of attenuation only once. Let N be the total number of pruning iterations, s_i be the initial sparsity, s_f be the final sparsity. The pruning rate at the n-th iteration is calculated as follows:

$$s_n = s_f + (s_i - s_f)\left(1 - \frac{n}{N}\right)^3 \tag{1}$$

for $n = 0, 1, ..., N$. We employ layer-wise pruning in our experiments.

3.4. Drop criterion

We follow the weight magnitude pruning strategy that sets the neurons with the least weight magnitude absolute value to zero. Specifically, let $\Theta^{(l)}, s^{(l)}, N^{(l)}$ denote the weight magnitude, sparsity, and the number of parameters of the l-th layer, respectively. We prune the $1 - s^{(l)}$ proportion of weights with least magnitude absolute, which is calculated as follows:

$$\mathbb{1}_{\Theta^{(l)}} = ArgTopK(|\Theta^{(l)}|, s^{(l)}N^{(l)}) \tag{2}$$

where $ArgTopK(v, k)$ returns the first k element indices of vector v sorted in descending order, and $|\cdot|$ denotes the absolute operation. The dropped elements are set to zero.

3.5. Regeneration

In the process of training a neural network, there exist such elements whose weight magnitudes are small but the gradients are significant. Small weight magnitudes are usually regarded to have a small contribution to the training loss and can be removed, whereas the effect of gradient cannot be ignored in the backward path. Therefore, these elements should be regenerated and continue to participate in the training. We follow RigL [7], which regenerates part of the pruned connections based on the gradient magnitude. Specifically, let $\mathbf{g}^{(l)}$ be the gradient of the l-th layer, and r be the proportion of regenerated connections with zero weight magnitude. The regenerated connections are identified according to the following equation:

$$\underset{i \notin \mathbb{1}_{\Theta^{(l)}}}{\mathbb{1}_{\mathbf{g}^{(l)}}} = ArgTopK(|\mathbf{g}^{(l)}|, r \cdot (1 - s^{(l)})N^{(l)}) \tag{3}$$

The connections in the mask associated with identified indices are set to 1. The corresponding weights remain zero so that the overall loss function is kept unchanged.

4. Experiment settings

4.1. PURE dataset

First collected by Stricker *et al*. [39], the PURE dataset recruited 10 subjects (8 males and 2 females) performing 6 head movements in front of a digital camera, including steady, talking, slow translation, fast translation, small rotation, and medium rotation. 60 one-minute videos with resolution 640×480 pixels and frame rate 30 frames per second (fps) are recorded. The data are saved in individual images

with portable network graphics (PNG) format. Ground-truth PPG is collected using a fingertip pulse oximeter when video recording simultaneously. The PURE dataset is split into train/val/test set with the ratio of $8:1:1$.

4.2. Implementation details

The video is partitioned into a segment of 5 second time window with a step length of 1 second. Viola-Jones face detector [42] is used to locate face in the first frame and Kanade-Lucas-Tomasi (KLT) algorithm [41] is used for the tracking in the following frames. The bounding box is cropped and resized to 192×128. We use mean squared error (MSE) as the loss function and Adam as the optimizer. We prune every 300 iterations. The maximum number of epoch is 4. For the experiments with pruning rate decay, the final epoch SNR is reported. For the experiments without decay, the best SNR is reported.

4.3. Evaluation metrics

1) Signal-to-noise-ratio (SNR). First proposed by de Haan *et al*. [5], SNR is used for evaluating the signal quality pulse predictions output by rPPG algorithms. Both the predicted signal and the ground truth signal are transformed to the frequency domain and the SNR is calculated as the ratio between the power of signal component and the noise component. The frequency of the signal component (dominant frequency) is determined by calculating the heart rate of the ground truth signal. The dominant frequency and its second harmonics together with a window length of ω is selected as the signal region. The frequency outside of the signal region is regarded as the noise region. In this paper, we set $\omega = 0.4$ Hz.

2) Floating point operations (FLOPs). FLOPs is typically used for measuring the computational complexity of a given network by summing up all the multiplication and summation operations needed to obtain the prediction. It is related with both input feature size and network configurations such as the number of layers, kernel size, layer types, etc. Sparse network has fewer FLOPs compared with its dense counterpart because the "damaged" connections are not included in the FLOPs.

5. Results and discussions

5.1. Naive pruning

We start from the naive pruning strategy where there is no pruning rate decay and connection regeneration. We investigate the performance by varying the initial sparsity s_i and final sparsity s_f, respectively. The maximum number of epoch is set to be 4 because we found that, when training PhysNet, the best score is usually obtained at the third or fourth epoch. The best SNR scores are selected for each round. We run five rounds and report the mean and standard deviation. The results are reported in Fig. 1.

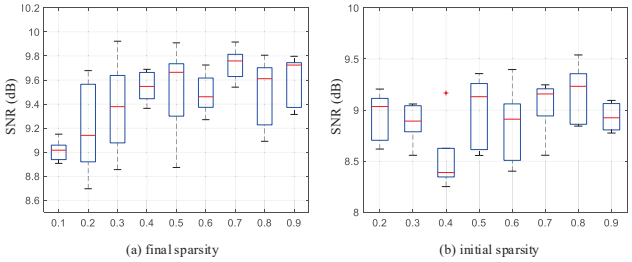

Figure 1. Boxplot of SNR results on PURE dataset. (a) $s_i = 1.0$, varying s_f, (b) varying s_i, $s_f = 0.1$.

The results show that the SNR decreases as s_f decreases and has no significant relationship with s_i. Fig. 1(a) shows that SNR starts to drop when s_f is less than 0.5. Mean SNR decreases from 9.59 dB ($s_f = 0.9$) to 9.01 dB ($s_f = 0.1$). Fig. 1(b) shows that SNR keeps basically stable (mean: 8.91 dB) when s_i changes. No monotonic increasing or decreasing trend is observed.

5.2. Dense-to-sparse vs. sparse-to-sparse

Table 4. SNR results of the pruning method in comparison with dense-to-sparse and sparse-to-sparse training. The result of original PhysNet is included for benchmarking.

Name	s_i	s_f	SNR (dB)
PhysNet	–	–	9.30± 0.14
Dense-to-sparse	1.0	0.9	9.59± 0.22
	1.0	0.8	9.49± 0.30
	1.0	0.7	9.73± 0.14
	1.0	0.6	9.49± 0.17
	1.0	0.5	9.51± 0.39
	1.0	0.4	9.55± 0.13
	1.0	0.3	9.37± 0.40
	1.0	0.2	9.21± 0.40
	1.0	0.1	9.01± 0.09
Sparse-to-sparse	0.9	0.1	8.93± 0.14
	0.8	0.1	9.15± 0.30
	0.7	0.1	9.05± 0.28
	0.6	0.1	8.84± 0.39
	0.5	0.1	8.90± 0.36
	0.4	0.1	8.52± 0.37
	0.3	0.1	8.88± 0.20
	0.2	0.1	8.93± 0.25

The results comparing dense-to-sparse training and sparse-to-sparse training are shown in Table 4. The results show that dense initialization generally performs better

than sparse initialization. In dense-to-sparse experiments, the SNRs when s_f is greater than 0.3 are higher than that of PhysNet. Only when s_f is less than 0.3 drops the SNR score down below PhysNet. Nevertheless, the SNR is still higher than most of sparse-to-sparse experiments. In general, dense-to-sparse experiments have average SNR 9.43 dB while sparse-to-sparse have average SNR 8.91 dB.

5.3. Effect of pruning rate decay and connection regeneration

In this experiment, we investigate the effect of pruning rate decay and connection regeneration by combining with and without these two variables while keeping $s_i = 1.0$, $s_f = 0.1$, resulting in four combinations. Each combination is ran five rounds and the average SNR scores are reported, as shown in Table 5.

Table 5. SNR results of the pruning method in comparison with pruning rate decay and connection regeneration. The result of original PhysNet is included for benchmarking.

	decay	regeneration	SNR (dB)
PhysNet	–	–	9.30± 0.14
PhysNet_{00}	×	×	9.01± 0.09
PhysNet_{10}	✓	×	9.78± 0.26
PhysNet_{01}	×	✓	7.86± 0.24
PhysNet_{11}	✓	✓	9.38± 0.25

The results show that the pruning rate decay is beneficial to the SNR improvement while the connection regeneration has a detrimental effect to the SNR. Specifically, the naive pruning method PhysNet_{00} achieves SNR score 0.29 dB lower than that of PhysNet. When pruning rate decay is added, the SNR increases to 9.78 dB, *i.e.*, 0.48 dB higher than PhysNet. The lowest SNR is achieved by the method PhysNet_{01} when only the connection regeneration is added, *i.e.*, 7.86 dB, 1.44 dB lower than PhysNet and 1.15 dB lower than PhysNet_{00}. The negative effect of regeneration can also be observed when comparing PhysNet_{11} and PhysNet_{10}, where SNR decreased by 0.40 dB when connection regeneration is added when there exists pruning rate decay. PhysNet_{10} achieves the highest SNR, even higher than the original PhysNet.

5.4. Finding the best combination

In this experiment, we concatenate the best-performing components together obtained in the previous experiments with an aim to find the best combination. The pruning detail is as follows: $s_i = 1.0$, varying s_f, with pruning rate decay, and without connection regeneration. The results are shown in Fig. 2.

The results show that, SNR increases as s_f decreases, which is opposite to that in Fig. 1(a). In order to show the relationship more intuitively, we plot in Fig. 3 the mean SNR as a function of s_f comparing with and without pruning rate decay. When pruning rate decay is added, SNR with larger s_f decreases while SNR with smaller s_f increases. This can be explained by the fact that when s_f is large, the network is close to dense. Adding decay introduces additional parameter perturbation. When s_f is small, however, the network is very sparse. If decay is not added, a lot of connections will be lost at the beginning of training, which is detrimental to the final performance.

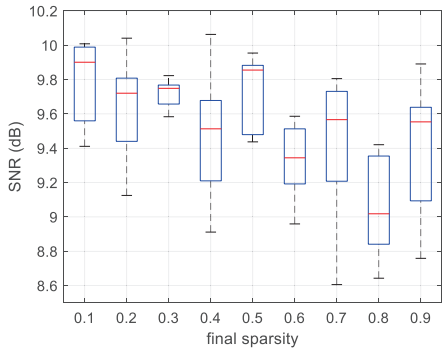

Figure 2. Boxplot of SNR results on PURE dataset under the setting: $s_i = 1.0$, varying s_f, with pruning rate decay, without connection regeneration.

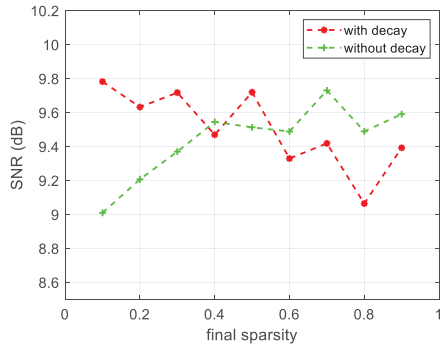

Figure 3. Average SNR results on PURE dataset comparing with and without pruning rate decay under the setting: $s_i = 1.0$, varying s_f, and without connection regeneration.

5.5. Computational complexity

The FLOPs needed for training and inference are calculated. For PhysNet model and the given input size in this paper, the FLOPs of the dense model for inference is 2.29×10^{11}. For the sparse model, the FLOPs can be approximately calculated as 2.29×10^{11} multiplied by corresponding sparsity due to the fact that the float operation is mainly concentrated in convolution, while that of batch

normalization and ReLU activation can be ignored. There are 2264 iterations in one epoch and the network is pruned every 300 iterations. We accumulate all the FLOPs in every iteration. The results are given in Table 6.

Table 6. FLOPs needed for training and inference.

Method			FLOPs	
s_i	s_f	decay	training	inference
PhysNet			2.07×10^{15}	2.29×10^{11}
1.0	0.1	×	2.69×10^{14}	2.29×10^{10}
1.0	0.1	✓	7.03×10^{14}	2.29×10^{10}
0.5	0.1	×	2.35×10^{14}	2.29×10^{10}
0.5	0.1	✓	4.28×10^{14}	2.29×10^{10}

The results show that, for both dense-to-sparse and sparse-to-sparse training, the FLOPs needed is one order of magnitude less than that of the dense model. Training with decay needs more FLOPs than that without decay. For the same final sparsity, dense initialization needs more FLOPs than sparse initialization.

6. Conclusion

rPPG features small-scale network architecture and limited number of training samples, which is also common in many medical imaging fields. This paper introduces pruning techniques which is currently an active research topic to rPPG network, observes network behavior, and draws some rules. PhysNet is chosen as the backbone architecture, based on which we add pruning rate decay, connection regeneration, varying initial sparsity and final sparsity. Experiments are conducted on a publicly available dataset PURE. The results show that the pruning rate decay is beneficial to the performance improvement, whereas the connection regeneration has a detrimental effect. Given the same final sparsity, dense initialization generally performs better than sparse initialization. The network seems insensitive to initial sparsity. The combination $s_i = 1.0$, $s_f = 0.1$, with decay, and without regeneration is the best trade-off between SNR and FLOPs, achieving average SNR 9.78 dB, increased by 0.48 dB in comparison with the original PhysNet.

It should be mentioned that the connection regeneration has a detrimental effect to the network performance, which contradicts the conclusion in image classification. The reason needs to be further investigation. The network sensitivity to the initialization, *i.e.*, network performance given different initializations, is another crucial issue worth to be investigated, which will be our future work. More experiments on other datasets and other rPPG networks are expected.

Acknowledgement

This research was supported by National Natural Science Foundation of China under Grant No.s 61903336, 61976190, and the Natural Science Foundation of Zhejiang Province under Grant No. LY21F030015.

References

[1] Serge Bobbia, Richard Macwan, Yannick Benezeth, Alamin Mansouri, and Julien Dubois. Unsupervised skin tissue segmentation for remote photoplethysmography. *Pattern Recognition Letters*, 124:82–90, 2019. 4

[2] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Michael Carbin, and Zhangyang Wang. The lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16306–16316, June 2021. 3

[3] Weixuan Chen and Daniel Mcduff. Deepphys: Video-based physiological measurement using convolutional attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 1, 2, 4

[4] Juan Cheng, Xun Chen, Lingxi Xu, and Z. Jane Wang. Illumination variation-resistant video-based heart rate measurement using joint blind source separation and ensemble empirical mode decomposition. *IEEE Journal of Biomedical and Health Informatics*, 21(5):1422–1433, 2017. 2

[5] Gerard De Haan and Vincent Jeanne. Robust pulse rate from chrominance-based rppg. *IEEE Transactions on Biomedical Engineering*, 60(10):2878–2886, 2013. 2, 6

[6] Jingda Du, Si-Qi Liu, Bochao Zhang, and Pong C. Yuen. Weakly supervised rppg estimation for respiratory rate estimation. In *Proceedings of the International Conference on Computer Vision (ICCV) Workshops*, 2021. 1

[7] Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery: Making all tickets winners. In *Proceedings of the International Conference on Machine Learning*, 2020. 2, 3, 4, 5

[8] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *Proceedings of the International Conference on Learning Representations*, 2020. 2, 3

[9] Eva Garcia-Martin, Crefeda Faviola Rodrigues, Graham Riley, and Hakan Grahn. Estimation of energy consumption in machine learning. *Journal of Parallel and Distributed Computing*, 134:75–88, 2019. 2

[10] Amogh Gudi, Marian Bittner, Roelof Lochmans, and Jan Van Gemert. Efficient real-time camera based estimation of heart rate and its variability. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, 2019. 1

[11] G De Haan and A Van Leest. Improved motion robustness of remote-ppg by using the blood volume pulse signature. *Physiological Measurement*, 35(9):1913–1926, 2014. 2

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceed-*

ings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016. 2

[13] Guillaume Heusch, Andr Anjos, and Sbastien Marcel. A reproducible study on remote heart rate measurement. In *arXiv*, 2017. 4

[14] Min Hu, Fei Qian, Dong Guo, Xiaohua Wang, Lei He, and Fuji Ren. Eta-rppgnet: Effective time-domain attention network for remote heart rate measurement. *IEEE Transactions on Instrumentation and Measurement*, 70:1–12, 2021. 1, 3

[15] Bin Huang, Weihai Chen, Chun-Liang Lin, Chia-Feng Juang, and Jianhua Wang. Mlp-bp: A novel framework for cuffless blood pressure measurement with ppg and ecg signals based on mlp-mixer neural networks. *Biomedical Signal Processing and Control*, 73:103404, 2022. 1

[16] Bin Huang, Weihai Chen, Chun-Liang Lin, Chia-Feng Juang, Yuanping Xing, Yanting Wang, and Jianhua Wang. A neonatal dataset and benchmark for non-contact neonatal heart rate monitoring based on spatio-temporal neural networks. *Engineering Applications of Artificial Intelligence*, 106:104447, 2021. 4

[17] Bin Huang, Chun-Liang Lin, Weihai Chen, Chia-Feng Juang, and Xingming Wu. A novel one-stage framework for visual pulse rate estimation using deep neural networks. *Biomedical Signal Processing and Control*, 66:102387, 2021. 1, 3

[18] Zhengjie Huang, Wenjin Wang, and Gerard De Haan. Nose breathing or mouth breathing a thermography-based new measurement for sleep monitoring. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2021. 1

[19] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 1

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the International Conference on Neural Information Processing Systems*, 2012. 2

[21] Yann LeCun, Corinna Cortes, and Christopher J.C. Burges. http://yann.lecun.com/exdb/mnist/. 1

[22] Yann LeCun, John S. Denker, and Sara A. Solla. Optimal brain damage. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 598–605, 1990. 2, 3, 4

[23] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. Snip: Single-shot network pruning based on connection sensitivity. In *Proceedings of the International Conference on Learning Representations*, 2019. 2, 3

[24] Magdalena Lewandowska, Jacek Rumiski, Tomasz Kocejko, and Jedrzej Nowak. Measuring pulse rate with a webcam - a non-contact method for evaluating cardiac activity. In *Proceedings of the Federated Conference on Computer Science and Information Systems*, 2011. 2

[25] Shiwei Liu, Tianlong Chen, Xiaohan Chen, Zahra Atashgahi, Lu Yin, Huanyu Kou, Li Shen, Mykola Pechenizkiy, Zhangyang Wang, and Decebal Constantin Mocanu. Sparse training via boosting pruning plasticity with neuroregeneration. In *Proceedings of the Conference on Neural Information Processing Systems*, 2021. 2, 4

[26] Si-Qi Liu and Pong C. Yuen. A general remote photoplethysmography estimator with spatiotemporal convolutional network. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, 2020. 4

[27] Xin Liu, Josh Fromm, Shwetak Patel, and Daniel McDuff. Multi-task temporal shift attention networks for on-device contactless vitals measurement. In *Proceedings of the Conference on Neural Information Processing Systems*, 2020. 4

[28] Birla Lokendra and Gupta Puneet. And-rppg: A novel denoising-rppg network for improving remote heart rate estimation. *Computers in Biology and Medicine*, 141:105146, 2022. 1

[29] Decebal Constantin Mocanu, Elena Mocanu, Peter Stone, Phuong H. Nguyen, Madeleine Gibescu, and Antonio Liotta. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature Communications*, 9:2383, 2018. 2, 3

[30] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. In *Proceedings of the International Conference on Learning Representations*, 2017. 3

[31] Michael C. Mozer and Paul Smolensky. Skeletonization: A technique for trimming the fat from a network via relevance assessment. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 107–115, 1988. 2, 3

[32] Xuesong Niu, Shiguang Shan, Hu Han, and Xilin Chen. Rhythmnet: End-to-end heart rate estimation from face via spatial-temporal representation. *IEEE Transactions on Image Processing*, 29:2409–2423, 2020. 1, 4

[33] Wimmer Paul, Mehnert Jens, and Condurache Alexandru. Cops: Controlled pruning before training starts. In *Proceedings of the International Joint Conference on Neural Network*, 2020. 3

[34] Ming-Zher Poh, Daniel J McDuff, and Rosalind W Picard. Non-contact, automated cardiac pulse measurements using video imaging and blind source separation. *Optics Express*, 18(10):10762–10774, 2010. 2

[35] Ying Qiu, Yang Liu, Juan Arteaga-Falconi, Haiwei Dong, and Abdulmotaleb El Saddik. Evm-cnn: Real-time contactless heart rate estimation from facial video. *IEEE Transactions on Multimedia*, 21(7):1778–1787, 2019. 1

[36] Spetlik Radim, Vojtech Franc, Jan Cech, and Jiri Matas. Visual heart rate estimation with convolutional neural network. In *Proceedings of the Brtish Machine Vision Conference*, 2018. 1, 2, 4

[37] 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)*, 115:211–252, 2015. 1

[38] Suraj Srinivas, Akshayvarun Subramanya, and R. Venkatesh Babu. Training sparse neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 455–462, 2017. 2, 3

[39] Ronny Stricker, Steffen Mller, and Horst-Michael Gross. Non-contact video-based pulse rate measurement on a mobile service robot. In *Proceedings of The IEEE International*

Symposium on Robot and Human Interactive Communication, pages 1056–1062, 2014. 4, 5

[40] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2019. 2

[41] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. *International Journal of Computer Vision (IJCV)*, 9(3):137–154, 1991. 6

[42] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2001. 6

[43] Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020. 3

[44] Wenjin Wang, Albertus C. Den Brinker, Sander Stuijk, and Gerard De Haan. Algorithmic principles of remote ppg. *IEEE Transactions on Biomedical Engineering*, 64(7):1479–1491, 2017. 2

[45] Bing-Fei Wu, Yi-Chiao Wu, and Yi-Wei Chou. A compensation network with error mapping for robust remote photoplethysmography in noise-heavy conditions. *IEEE Transactions on Instrumentation and Measurement*, 71:1–11, 2022. 3

[46] Lin Xi, Weihai Chen, Changchen Zhao, Xingming Wu, and Jianhua Wang. Image enhancement for remote photoplethysmography in a low-light environment. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, 2020. 4

[47] Yuhui Xu, Yuxi Li, Shuai Zhang, Wei Wen, Botao Wang, Yingyong Qi, Yiran Chen, Weiyao Lin, and Hongkai Xiong. Trp: Trained rank pruning for efficient deep neural networks. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2020. 4

[48] Haoran You, Chaojian Li, Pengfei Xu, Yonggan Fu, Yue Wang, Richard G. Baraniuk, and Yingyan Lin. Drawing early-bird tickets: Towards more efficient training of deep networks. In *Proceedings of the International Conference on Learning Representations*, 2020. 3

[49] Zitong Yu, Xiaobai Li, and Guoying Zhao. Remote photoplethysmograph signal measurement from facial videos using spatio-temporal networks. In *Proceedings of the British Machine Vision Conference*, 2019. 1, 2, 4, 5

[50] Ganzhao Yuan, Li Shen, and Wei-Shi Zheng. A block decomposition algorithm for sparse optimization. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020. 3

[51] Changchen Zhao, Chun-Liang Lin, Weihai Chen, Ming-Kun Chen, and Jianhua Wang. Visual heart rate estimation and negative feedback control for fitness exercise. *Biomedical Signal Processing and Control*, 56:101680, 2020. 1

[52] Michael H. Zhu and Suyog Gupta. To prune, or not to prune: Exploring the efficacy of pruning for model compression. In *Proceedings of the International Conference on Learning Representations*, 2018. 2, 3