COMPUTATIONAL PROBLEMS OF ELECTRICAL ENGINEERING

Vol. 13, No. 1, 2023

EVALUATION OF A SNIP PRUNING METHOD FOR A STATE-OF-THE-ART FACE DETECTION MODEL

Artem Melnychenko, Oleksii Shaldenko

National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine

artemxl@gmail.com, o.shaldenko@gmail.com

https://doi.org/10.23939/jcpee2023.01.

Abstract: With rapid development of machine learning and subsequently deep learning, deep neural networks achieved remarkable results in solving various tasks. However, with increasing the accuracy of trained models, new architectures of neural networks present new challenges as they require significant amount of computing power for training and inference. This paper aims to review existing approaches to reducing computational power and training time of the neural network, evaluate and improve one of existing pruning methods for a face detection model. Obtained results show that the presented method can eliminate 69 % of parameters while accuracy being declined only by 1.4 %, which can be further improved to 0.7 % by excluding context network modules from the pruning method.

Key words: pruning, deep neural networks, inference, optimization, face detection.

1. Introduction

Modern machine learning methods allow training models on large amounts of data so that they are able to achieve high accuracy in solving problems. Rapid development of deep neural network (DNN) architectures caused breakthroughs in various fields. For example, four research groups achieved significant improvements solving a speech recognition task with a deep neural network instead of classic Gaussian mixture models (GMM) and the Hidden Markov Model (HMM) approach [1]. In 2015 during the ILSVRC competition, ResNet architecture with 152 layers was introduced, achieving a 3.57 % error rate on the ImageNet test dataset for object detection [2]. ResNet model architecture also achieved state-of-the-art results in some other tasks, namely, object detection and object segmentation on the COCO dataset, and image localization on the ImageNet dataset. In 2016, the MTCNN model was introduced to solve face detection tasks, achieving 95 % average accuracy on the FFDB dataset [3].

More recent breakthroughs are focused on NLP models and image generation. A paper describing the BERT language model was published in 2019, achieving state-of-the-art results of eleven NLP tasks, including pushing the GLUE score to 80.5 % (7.7 % of absolute improvement) [4]. GPT-3 model followed the success in 2020, achieving 86.4 % accuracy with the LAMBADA word prediction challenge [5].

However, such training requires large amounts of computing power (sometimes even whole clusters), and a considerable amount of time. To train a GPT-3 model with 175 billion parameters a total of 3.14 * 1023 flops, the amount of computing power is equivalent to 355 years of training on a single NVIDIA Tesla V100 GPU. Also, questions of ecological impact arise when training such a large model [6]. The authors emphasize the importance of evaluating efficiency as an assessment criterion for research together with accuracy and associated indicators. Additionally, they suggest disclosing the monetary expense or "price tag" for conducting training and using models in order to provide benchmarks for the examination of progressively effective techniques.

Lastly, the inference of such models become impractical due to high hardware requirements (for example, the requirement of a powerful GPU). Inference on embedded systems has been made possible by a number of optimization strategies. However, those methods either have limitations or are architecture-specific [7]. Deep neural networks are frequently accelerated using model compression. A DNN can be made more efficient by using compression to lower its resource and computational demands which also results in a drop in model precision. [8]. Alternative methods have been developed to prevent this, such as offloading some or all processing to a cloud server, which has the resources needed for fast inference times [9]. However, those methods may be not available for latency requirements. Furthermore, due to privacy concerns, transferring sensitive data across a network could be forbidden.

In recent years, various approaches were tested to decrease the number of parameters in neural networks, speed up the computation and enable practical use on low-end devices. One of such techniques is foresight pruning, which allows pruning of DNN before training even starts, therefore reducing the need for computational power to conduct experiments in a reasonable time. This research is aimed to evaluate one of the state-of-the-art foresight pruning methods, Single-shot Network Pruning (SNIP), on the face detection neural network model based on the RetinaFace architecture, and improve its efficiency.

2. Related work

There is substantial redundancy in the parameterization of many deep learning models, according to research [10]. Authors claim that for each feature, it is possible to properly forecast the remaining values using just a few weight values. They also demonstrate that many parameter values can be predicted without the need to learn them.

Various techniques exist to reduce the number of parameters. They can be split into the following categories.

Dimensionality reduction. To drastically reduce the number of parameters while maintaining the expressive capacity of the layer, Jaderberg et al in their work transform the dense weight matrices of the fully-connected layers to the special format [11]. This technique allows fulfilling the task. A similar approach can be used for convolutional layers, to reduce convolutional filters by exploiting cross-channel or filter redundancy to construct a low-rank basis of filters [12].

Pruning after training. Song Han et al suggest a method that prunes redundant connections using a three-step method: after initial training, they reset low-valued weights to zero and conduct training again [13]. Other developments in the field include lookahead pruning, which works by extending the single-layer optimization to a multi-layer optimization [14], and incorporates the information from all second-order derivatives of the error function to perform pruning [15].

Pruning before training. Recent studies showed that pruning randomly initializing neural networks before training may be done with little to no loss in accuracy. The iterative Magnitude Pruning (IMP) algorithm repeats multiple cycles of training, pruning, and weight rewinding to identify extremely sparse neural networks [16]. However, it required several costly cycles of training and pruning and the use of particular sets of hyperparameters. An alternate strategy leverages the gradients of the training loss at initiation to prune the network in a single shot, avoiding these issues [17].

Namhoon Lee et al in their paper suggest an approach of pruning before training the network by computing a saliency criterion that identifies structurally important weights for the neural network and removing weights that are not important. Results are evaluated on a wide range of architectures (LeNet on MNIST image, LSTMs, and GRUs on MNIST sequential, VGGs, and AlexNets on CIFAR-10) and show less than one percent accuracy degradation compared to the reference network.

In this paper, the authors are focusing on pruning before training and the SNIP algorithm specifically.

3. Materials and methods

Face detection model. For the neural network architecture, a variation of the RetinaFace model was chosen as the state-of-the-art face detection model [18]. The architecture uses feature pyramids calculated from the outputs of the ResNet-50 convolution blocks (Table 1).

Table 1

Convolution blocks of ResNet-50

Block name	Layers (kernel size, channels)	Repeat count
C2	1×1, 64 3×3, 64 1×1, 256	3
C3	1×1, 128 3×3, 128 1×1, 512	4
C4	1×1, 256 3×3, 256 1×1, 1024	36

Outputs of convolution blocks are then used to calculate the layers of feature pyramids using top-down and lateral connections (Fig. 1).

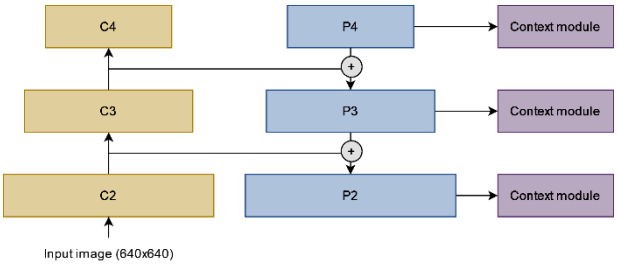

Fig. 1. RetinaFace model using ResNet-50 backbone.

The context module uses convolutional layers with a bigger filter size of 5 and 7 to increase the window size (Fig. 2).

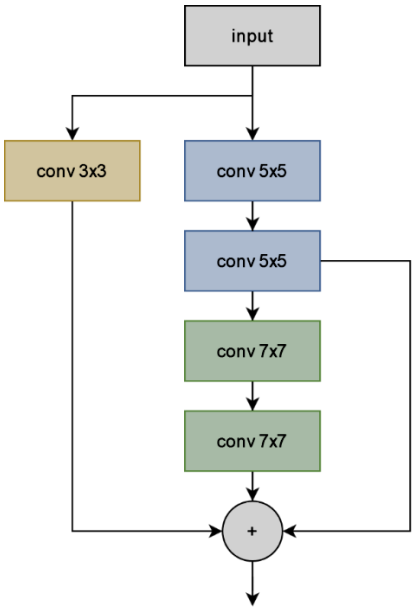

Fig. 2. Context block architecture

Finally, the model computes 3 outputs using multi-task learning with outputs from context modules (anchors).

- Face classification task, indicating whether the given anchor contains a face. Binary cross entropy is used as a loss function (L_{cls});
- Face box regression task, containing coordinates where the face is located. Smooth L1 is used as a loss function (L_{box});
- Face landmarks regression task, with coordinates of the five facial landmarks. Smooth L1 is used as a loss function (L_{pts}).

Multitask loss is computed with the following equation:

$$L = L_{cls} + \lambda_1 L_{box} + \lambda_2 L_{pts} \quad (1)$$

Foresight pruning. SNIP algorithm works as follows:

- Initialize weights *w* with variance scaling algorithm (eq. 2);
- Get the mini batch of training data D^b from dataset D;
- Evaluate connection sensitivity on each connection using (eq. 3);
- Select the top k connection based on the sensitivity score;
- Train the network using selected scores.

For this algorithm to be effective on various architectures, weights for each layer must have the same variance. This method is used on new layers which were not included in the pre-trained backbone network. To achieve this, a variance scaling algorithm is used, which works as follows:

$$w_j = \frac{\text{rand } n}{\bar{n}} \quad (2)$$

The connection sensitivity is evaluated using the following formula, where g_j is the magnitude of the derivative:

$$s_j = \frac{g_j\ w;D}{\sum_{k=1}^{m} g_k\ w;D} \quad (3)$$

To vary the number of weights to be pruned, a *compression ratio (ρ)* parameter was introduced, which denotes the number of parameters in the original network divided by the number of parameters remaining after pruning.

4. Results and discussion

The performance of the SNIP pruning algorithm was empirically benchmarked on the face detection task by single-shot pruning before training using the described RetinaFace model. The training was conducted on a WIDERFace dataset [6]. The pruned network is trained in the standard way. Specifically, for gradient descent an SGD with a momentum parameter of 0.9 batch size of 2 and a weight decay rate of 0.0005. The initial learning rate is set to 0.001 and decayed 2 times at 70 and 90 epochs. The total number of epochs is 100.

The training resulted pruned weights redistribution across the network, where more weights were pruned in first layers of the network. The results of the distribution of the pruned weights are shown in Figure 3.

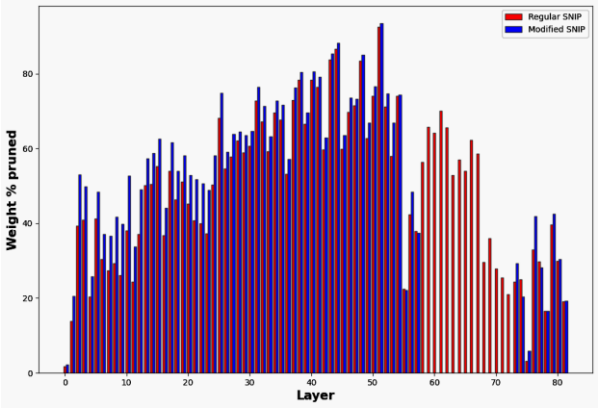

Fig. 3. Unpruned weights distribution by layer.

The loss function value during training was traced with Tensorboard and is displayed in Figure 4.

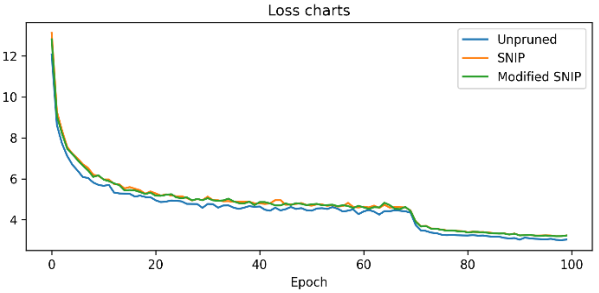

Fig. 4. Tensorboard loss charts.

Evaluation results, obtained after training pruned and unpruned models, are displayed in Table 2.

Table 2

Evaluation results

Model	WIDERFace (Easy)	WIDERFace (Medium)	WIDERFace (Hard)	Trainable parameters
Without pruning	93.5 %	91.6 %	75.9 %	3756032
Pruned (SNIP)	93.5 %	91.0 %	74.5 %	1187762
Pruned (Modified SNIP)	**93.5 %**	**91.0 %**	**75.2 %**	1187762

6. Conclusion

Foresight pruning is a promising approach to reducing the size of neural network model and the number of parameters before training starts. Evaluation

of the SNIP algorithm to eliminate weights, which contribute the least to the model has shown that number of parameters can be reduced by 69 % when losing only 1.4 % of accuracy evaluated on the hard part of the WIDERFace dataset. Additionally, modifying the algorithm to exclude context modules from pruning reduces the accuracy loss to only 0.7 %. Further research is required to evaluate importance of architecture components for using foresight pruning methods on computer vision models.

References

[1] G. Hinton *et al*., "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," in *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82-97, Nov. 2012.

[2] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 770-778.

[3] K. Zhang, Z. Zhang, Z. Li and Y. Qiao, "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks," in *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499-1503, Oct. 2016.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, vol. 1 *(Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota.

[5] Brown, T *et al.* "Language models are few-shot learners", *Advances in neural information processing systems*, *33*, pp.1877-1901.

[6] Schwartz, Roy, Jesse Dodge, Noah Smith and Oren Etzioni. "Green AI." *Communications of the ACM 63*, 2019, pp. 54-63.

[7] Ben Taylor, Vicent Sanz Marco, Willy Wolff, Yehia Elkhatib, and Zheng Wang. "Adaptive deep learning model selection on embedded systems", in *Proc. 19th ACM SIGPLAN/SIGBED International Conference on Languages, Compilers, and Tools for Embedded Systems*, New York, USA, pp. 31–43, 2018.

[8] Han, Song, Huizi Mao and William J. Dally. "Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding", *arXiv: Computer Vision and Pattern Recognition*, 2015.

[9] S. Teerapittayanon, B. McDanel and H. T. Kung, "Distributed Deep Neural Networks Over the Cloud, the Edge and End Devices", *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, Atlanta, USA, pp. 328-339, 2017.

[10] Misha Denil, Babak Shakibi, Laurent Dinh, Marc'Aurelio Ranzato, and Nando de Freitas. "Predicting parameters in deep learning", in *Proc. 26th International Conference on Neural Information Processing Systems*, vol. *2 (NIPS'13)*. Curran Associates Inc., Red Hook, USA, pp. 2148–2156, 2013.

[11] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. "Speeding up Convolutional Neural Networks with Low Rank Expansions", In *Proceedings of the British Machine Vision Conference*. BMVA Press, September 2014.

[12] Novikov, A., Podoprikhin, D., Osokin, A. and Vetrov, D.P.. "Tensorizing neural networks", *Advances in neural information processing systems*, *28*, 2015.

[13] Song Han, Jeff Pool, John Tran, and William J. Dally. "Learning both weights and connections for efficient neural networks", In *Proceedings of the 28th International Conference on Neural Information Processing Systems* – vol. *1 (NIPS'15)*, MIT Press, Cambridge, USA, pp. 1135–1143, 2015.

[14] S. Park, J. Lee, S. Mo and J. Shin, , "Lookahead: A far-sighted alternative of magnitude-based pruning", *arXiv preprint arXiv:2002.04809*, 2020.

[15] B. Hassibi and D. G. Stork. "Second order derivatives for network pruning: optimal brain surgeon", in *Proc. 5th International Conference on Neural Information Processing Systems (NIPS'92)*, Morgan Kaufmann Publishers Inc., San Francisco, USA, pp. 164–171, 1992.

[16] J. Frankle, and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks", in *Proc. 7th International Conference on Learning Representations*, New Orleans, USA, May 6-9, 2019..

[17] N. Lee, T. Ajanthan and P.H. Torr, "The lottery ticket hypothesis: Finding sparse, trainable neural networks", in *Proc. 7th International Conference on Learning Representations*, New Orleans, USA, May 6-9, 2019.

[18] J. Deng, J. Guo, E. Ververas, I. Kotsia, Stefanos Zafeiriou, "*RetinaFace: Single-stage Dense Face Localisation in the Wild*",in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5203-5212, 2020.

ОЦІНКА МЕТОДУ ПРУНІНГУ SNIP НА СУЧАСНІЙ МОДЕЛІ ДЕТЕКЦІЇ ОБЛИЧЧЯ

Артем Мельниченко, Олексій Шалденко

Із швидким розвитком машинного навчання та як наслідок глибокого навчання, глибокі нейронні мережі досягли помітних результатів у різних областях. Однак із збільшенням точності навчених моделей, нові архітектури нейронних мереж створюють нові виклики, оскільки вони вимагають значної кількості обчислювальних потужностей для навчання та подальшого використання. Ця стаття має на меті переглянути існуючі підходи до зменшення обчислювальних потужностей та часу потрібних для навчання нейронних мереж, оцінити та вдосконалити один із таких методів на моделі для детекції обличь. Результати показали, що представлений метод може усунути 69 % параметрів, втрачаючи лише 1,4 % у точності, і може бути додатково покращений зменшивши втрату точності до 0,7 %, виключивши контекстні модулі мережі із методу.

Artem Melnychenko, PhD student at the Department of Software Engineering for Power Industry, NTUU “Igor Sikorsky Kyiv Politechnic Institute”.

His research focused on computer vision technologies and application of deep learning to solve task in this field.

Oleksii Shaldenko graduated from the Heat power engineering department of NTUU "Igor Sikorsky KPI" in 2006, received Ph.D. degree in 2017. He has been working as an associate professor at the Heat power engineer department of NTUU "Igor Sikorsky KPI" since 2014.

His research is focused on computational fluid mechanics, dynamic systems modeling, computer vision technologies, and deep learning

Received: 28.04.2023, Accepted: 28.06.2023