Green AI

Bangguo Xu, Simei Yan, Liang Liu Supervisor: Prof. Dr. Frank-Michael Schleif

bangguo.xu@study.thws.de, simei.yan@study.thws.de, liang.liu@study.thws.de, frank-michael.schleif@thws.de

What is Green AI?

Green AI[1] refers to the development and use of **sustainable and energy-efficient AI technologies** with the aim of **reducing energy consumption and carbon emissions**, and helping to mitigate the impacts of climate change.

Green AI vs. Green IT

		Green AI	Green IT
Similarity		Reduce energy consumption and computing load by optimizing algorithms, hardware, and software.	
Difference	**Technical focus**	sustainability of machine learning algorithms and models	energy efficiency, resource utilization efficiency and environmental protection of computer hardware and software
	Application scope	• image recognition • natural language processing (NLP) • intelligent transportation • intelligent manufacturing • other fields	• data centers • servers • network equipment • mobile phones • laptops • other computing equipment

CNN and its relavance to Green AI

Convolutional Neural Networks (CNN) are deep learning models initially used for image recognition tasks. They are inspired by the functioning of the human visual system, mimicking the connectivity of neurons to process image data.

In the context of Green AI, efficient utilization of computational resources is crucial. CNN is widely applied in mobile devices and embedded systems due to its lightweight, efficiency, and accuracy, enabling the implementation of Green AI. By optimizing network structures, reducing model parameters, and computational complexity, CNN can maintain high performance while reducing computational resource requirements, thereby driving the development of Green AI.

Feasible Research Methods

Research directions for **lightweight networks:**

- **Compress the trained model:**
  1. Model Pruning
  2. Knowledge Distillation
- **Direct training of lightweight networks:**
  1. MobileNet
  2. ShuffleNet

Knowledge Distillation

Knowledge distillation[2] transfers knowledge from a complex neural network (teacher network) to a smaller network (student network).

1. Train the Teacher Model.
2. Generate soft targets using a high temperature, Thigh.
3. Train the Student Model simultaneously using soft target, Thigh and hard target, Thigh.
4. Set temperature T = 1 for the Student Model during online inference.

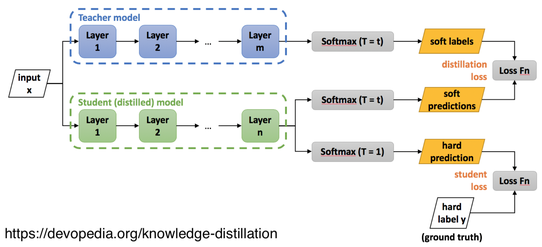

Figure 1. Knowledge distillation

Model Pruning

According to the smallest unit of pruning, model pruning can be divided into two types:

- **Non-structural pruning**: weight pruning, vector pruning, kernel pruning.
- **Structural pruning**: convolution kernel pruning, channel pruning and layer pruning.

L1-norm based Channel Pruning:

1. Perform L1-norm on the convolution kernel of each convolution layer, **add the L1-norm term to the cost function**, and optimize the cost function using optimization methods such as gradient descent.
2. Perform threshold processing on the optimized convolution kernel, and **set the weight value of the convolution kernel smaller than the threshold to 0**.
3. **Remove** the pruned convolution kernel and the corresponding output channel, and adjust the input channel of the next layer.
4. **Fine-tuning** is performed on the pruned model to restore the prediction accuracy of the model.

MobileNet

MobileNet V1[3] is to utilize Depthwise Separable Convolution operation.

Depthwise Separable Convolution decomposes the conventional convolution into two steps: **depthwise convolution and pointwise convolution.**

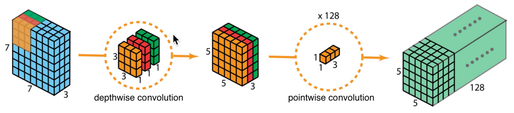

Figure 2. Mobilenet Convolution operation.[3]

ShuffleNet

ShuffleNet[4] utilizes two new operations:

1. **Pointwise group convolution**:
   1. **Group convolution**: is used to divide the input feature map into several groups and perform a convolution operation on each group, and finally stitch the convolution results of each group together to obtain the output feature map.
   2. **Pointwise convolution**: Convolution operation with kernel size *1x1*.
2. **Channel shuffle**: suppose a convolutional layer with *g* groups whose output has *g * n* channel;
   1. first reshape the output channel dimension into *(g,n)*;
   2. transposing;
   3. flattening it back as the input of next layer.

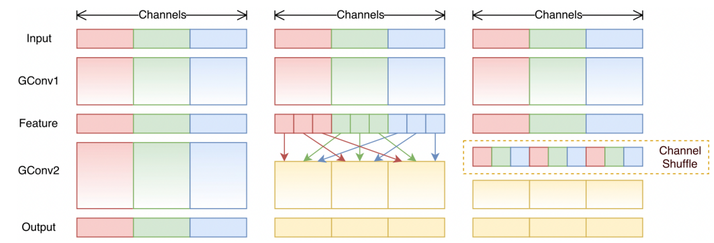

Figure 3. Channel shuffle[4]

Experiment

In order to verify the effectiveness of the proposed approach, we designed two simple experiments using knowledge distillation and model compression, respectively. The experimental code can be found in our GitHub repository: https://github.com/Sakayanagi-Arisu/Green-AI-project

1.Knowledge distillation

We designed a convolutional neural network (CNN) with two convolutional layers as the teacher model and a neural network comprised solely of fully connected layers as the student model. Both models were trained on the FashionMNIST dataset. We replaced the cross-entropy loss function in the student network with a new loss function that combines distillation loss and cross-entropy loss, aiming to make the student model mimic the output distribution of the teacher model while maintaining high prediction accuracy, thus improving the generalization performance of the student model. The experimental results are as follows:

	Teacher Net	Student Net(without kd)	Student Net(with kd)
Train accuracy	98.90%	81.11%	92.94%
Test accuracy	92.41%	79.04%	89.06%
Model size	4690KB	430KB	430KB

Table 1. Knowledge distillation result

2.Model Pruning

In this experiment, we used the same CNN architecture as the teacher network in knowledge distillation as the original model, and also trained it on the FashionMNIST dataset. Then, we applied the L1Norm pruning strategy to the model with different **sparsity**. The experimental results are shown below:

Sparsity The proportion of weight parameters that are zero in the network.

	Original Net	Pruned Net	Pruned Net	Pruned Net
Sparsity	0	0.6	0.8	0.95
Model size	4690KB	1893KB	948KB	297KB
Total parameters	1,119,882	483,848	241,936	75,242
Test accuracy	92.50%	92.30%	91.68%	89.77%

Table 2. Model Pruning result.

Next Step

In the next step of our plan, we will study how to design more lightweight neural network models using methods as mentioned before to reduce the number of model parameters and computational complexity without reducing performance as much as possible, thereby achieving efficient and energy-saving models. We will also establish suitable evaluation indicators and conduct experimental comparisons to measure the results before and after optimization.

References

[1] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, "Green ai," *Communications of the ACM*, vol. 63, no. 12, pp. 54–63, 2020.

[2] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[3] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[4] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Our Github repository: https://github.com/Sakayanagi-Arisu/Green-AI-project