UNIVERSITY OF GOTHENBURG

Literature review

Knowledge Distillation

Advisor: Yinan Yu

Student: Yu-Ping Hsu

Email: gushsuyu@student.gu.se

Abstract

In theory, the models with the better performance are often very large or assembled using multiple models. However, due to hardware constraints, lightweight models are needed when deploying deep machine learning in resource-constrained environments such as mobile terminals. Knowledge distillation is one of the methods that have been used to compress models for a couple of year. It is a process of transferring knowledge from a teacher model to a student model in order to improve the performance of deep learning models on mobile devices. In this review, the principle of knowledge distillation is analyzed and two optimization methods, early stopping and adding noise in the training student model as well as the parallel training method online distillation, are studied. According to the study, knowledge distillation is able to improve the general performance of student models.

1. Introduction

In recent years, due to the development of deep machine learning, network models have become larger and larger in the race for higher performance. An increasing amount of computing resources are required to extract information from large datasets during training. In theory, the models with the better performance are often very large or assembled using several models. However, because of the speed of inference and source requirement, it is not easy to deploy large models on mobile resources. As the application scenarios of deep machine learning continue to expand, the need for lightweight networks has begun to increase. A lightweight network refers to a deep machine learning model that uses a series of optimization techniques to significantly reduce the number of model parameters while ensuring model accuracy.

There are many ways to reduce the weight of the network: (1) compress the trained models. (2) train lightweight network directly. (3) speed up operations of convolution computing and (4) deploy hardware. This survey focuses on knowledge distillation which is one kind of model compression. Knowledge distillation (KD for short) refers to the process of distilling the knowledge contained in the trained model and then transferring it to smaller models. Through knowledge transfer, small networks can have performance similar to large networks while requiring less computational resources. This complex large network can be regarded as a teacher, while the small network can be regarded as a student.

2. Knowledge Distillation Algorithm [1]

In Hinton's paper, KD algorithm is only used in classification problems. The common characteristic of the classifications is that the models have a Softmax layer at the end. The output from Softmax is the probability value of the corresponding classes. When a teacher model which has strong generalization ability is used to distill and train the student model, the student model will learn the generalization ability from the teacher model. The method which Hinton proposes is to use the probability of the class output by the Softmax layer from the teacher model to be the "soft-target".

2.1. Hard-target and soft-target

Hard-targets are actually the annotated labels of the data. In the task of classifying horses and donkeys, if the target is horses, the hard-target only has the value 1 at the label of horses while the value is 0 at the donkey labels. Soft-target is the probability output from the Softmax layer of the teacher model. Each class is assigned a probability.

The training method of the traditional neural network is to define a loss function. The loss is the difference between the predicted value and the real value. The training process consists of estimating the maximum likelihood of ground truth. The problem is that the information entropy of hard targets is not large. Suppose the probability of the horse class given by the model is 0.6, and the label is 1. Because the value of the soft-target is less than one, the loss calculated using hard labels will be larger than the loss from model using soft-target. Because the loss calculated by using hard labels is large, the amplitude of backward propagation becomes larger. This can easily cause the model to prefer certain features and reduce the generalization ability of the model.

2.2. Temperature of distillation

In order to let the student model learn more valuable information, Hinton added the temperature parameter T to the Softmax function:

$$q_i = \frac{exp\left(\frac{z_i}{T}\right)}{\sum_j exp\left(\frac{z_j}{T}\right)}, \tag{1}$$

Where q is the probability of each class, z is the logits of each class and T is the temperature.

When T is equal to one, the equation (1) is equal to the original Softmax function. When T < 1, the probability distribution is steeper than the original Softmax function. When T > 1, the probability distribution becomes flatter. Figure 1 shows how different T values affect the probability distribution.

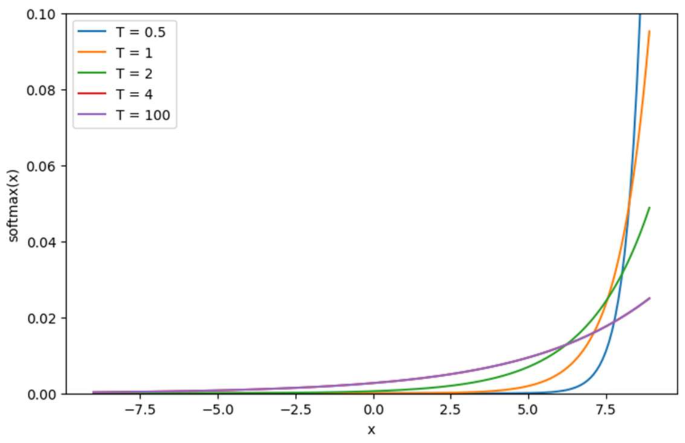


Figure 1 Graph of Softmax function results for different T

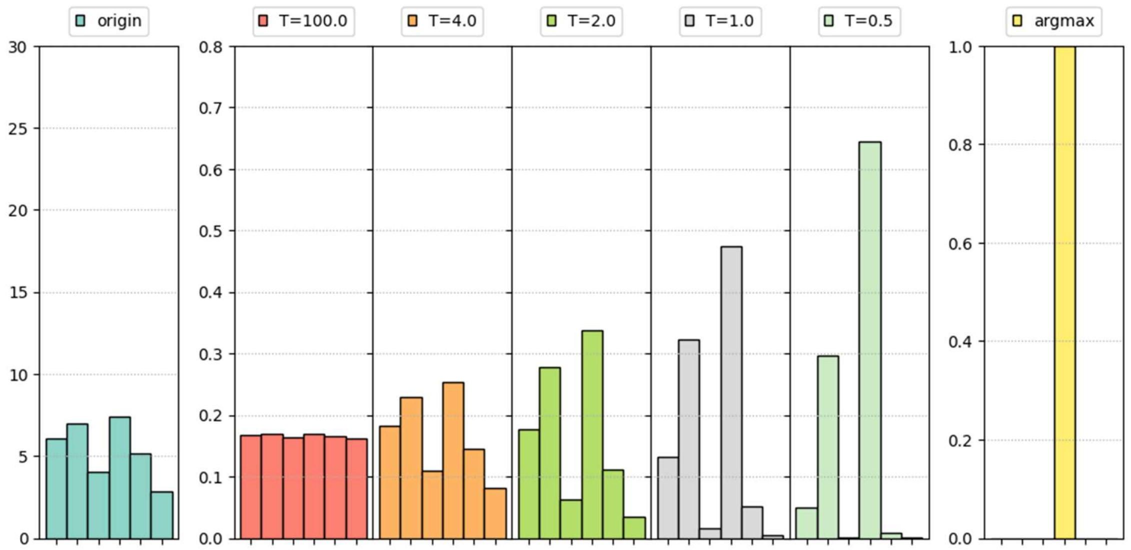
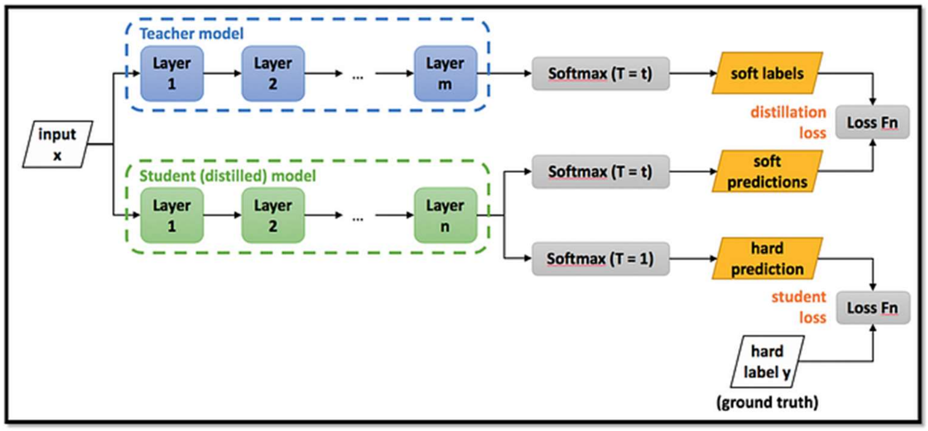


Figure 2: From left to right: Histogram plot for the classes, the middle five plots are probabilities distribution for different T values and the rightmost plot is the output from argmax.

In order to understand how the temperature in equation (1) affects the probability distribution data for six classes generated randomly. Figure 2 shows that when T becomes smaller, the probability distribution for different classes gradually moves from uniform distribution to concentrated distribution. In other words, when T is smaller, the smaller weights will be given to classes other than the correct class. Less information for these positive labels will be saved.

The setting of the temperature depends on the degree of attention given to negative labels during the training of the student model. When a lower temperature is used, less attention is given to negative labels, especially those that are significantly lower than the average. Conversely, when a higher temperature value is used, the values associated with negative labels will increase relatively. The student model will give more attention to negative labels. In fact, negative labels contain certain information, especially those whose probability value is significantly higher than the average value.

2.3. Progress of distillation and loss function



Step 1: Compute distillation loss

First, when the teacher network makes predictions, temperature t is used to calculate Softmax. The calculated result is called soft label. Temperature t is also used to calculate Softmax for the predictions from the student model. The calculated result is called soft prediction. The loss originating from soft label and soft predictions is called distillation loss.

Step 2: Compute student loss

Temperature T equal to one is used to compute the prediction output from the student model and get hard predictions. Subsequently the loss for hard predictions and ground truth is calculated in order to get student loss.

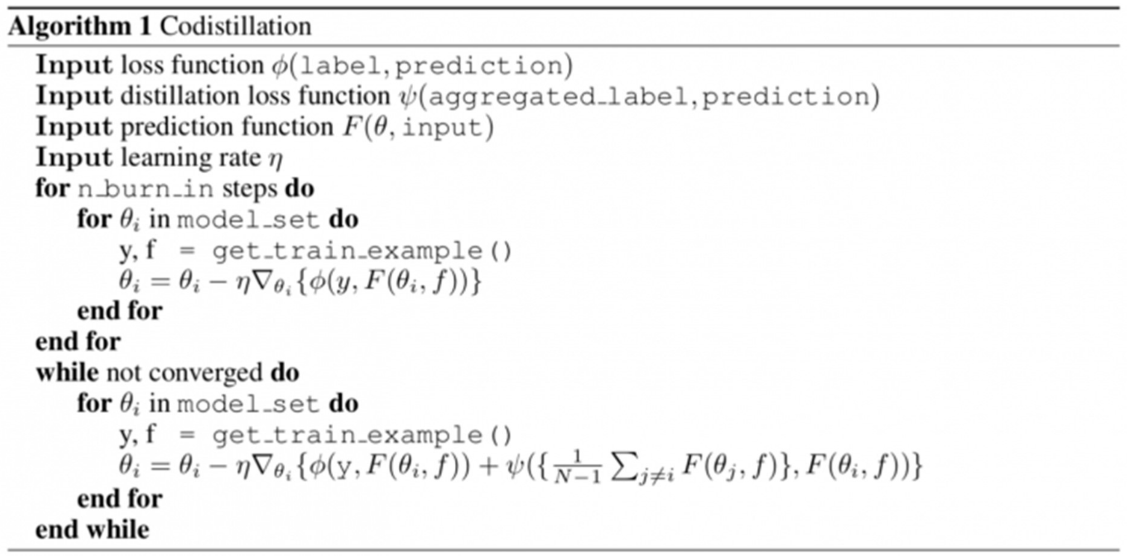
The equation for total loss is

$$L(x;W) = \alpha * H\big(y, \sigma(z_s; T = 1)\big) + \beta * H\big(\sigma(z_t, T = t), \sigma(z_s, T = t)\big), \tag{2}$$

where x is the input, W are parameters used in the student model, y is the ground truth label, H is the cross-entropy loss function, σ is the Softmax function parameterized by the temperature T, and α and β are coefficients. z_s and z_t are the predictions from the student and teacher model respectively.

3. Online distillation [2]

Geoffrey Hinton and a researcher from Google, Google Brain, DeepMind published the paper "Large scale distributed neural network training through online distillation" in ICLR 2018. The article discusses another method of using distillation. This approach does not require complex multi-level setups or the use of many new hyperparameters. There are two claims mentioned in the paper. The first claim is that online distillation allows us to use additional parallelism to adapt to very large datasets and double the speed of distillation. The second claim is that online distillation is a cost-effective method that can improve the accuracy of a model. This online distillation method is named "codistillation". The characteristic of codistillation is that each node in the distributed environment can be a teacher and student for each other. They extract knowledge from each other to improve the model performance of other nodes. The specific algorithm is as follows:



Algorithm 1 Codistillation

```
Input loss function $\phi$(label, prediction)
Input distillation loss function $\psi$(aggregated_label, prediction)
Input prediction function $F(\theta,$ input)
Input learning rate $\eta$
for n_burn_in steps do
    for $\theta_i$ in model_set do
        y, f = get_train_example()
        $\theta_i = \theta_i - \eta\nabla_{\theta_i}\{\phi(y, F(\theta_i, f))\}$
    end for
end for
while not converged do
    for $\theta_i$ in model_set do
        y, f = get_train_example()
        $\theta_i = \theta_i - \eta\nabla_{\theta_i}\{\phi(y, F(\theta_i, f)) + \psi(\{\frac{1}{N-1}\sum_{j\neq i} F(\theta_j, f)\}, F(\theta_i, f))\}$
    end for
end while
```

4. Other KD methods

4.1. Use early-stopping teacher regularization [3]

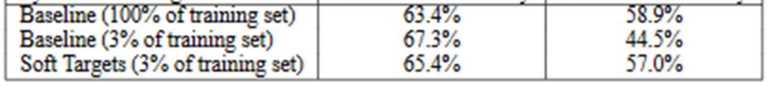
In the 'On the Efficacy of Knowledge Distillation' paper, KD soft labels are still used, but one regularization is added to the model. The authors claim that it is not necessarily the fact that a teacher model with better performance can distill a better student model. However, the authors did not actually supply a distillation strategy in the paper. According to the article, the distillation should be stopped when the training loss curve is close to convergence. This method is called early-stopping teacher regularization.

4.2. Use dataset with noise to train the student models [4]

In a paper published in 2020, Qizhe Xie and other authors proposed a method of training the student model using datasets with noise. KD soft labels are still used. First ImageNet is used to train the teacher model. Then pseudo labels for JFT dataset by the trained teacher model were generated. Finally, the JFT dataset with pseudo labels and ImageNet were used to train the student model. While training the student model, use the following tricks were used: (1) add model noise (dropout 0.5, stochastic depth 0.8 and so on) to improve the robustness and generalization of the model (2) filter images with low confidence from the teacher model (3) data balance: the number of images in different classes should be balanced (4) use soft labels as output in the teacher model.

5. Results

Table 1 shows that using soft targets can reduce overfitting compared to baseline training accuracy. However, the accuracy of the test set improves.



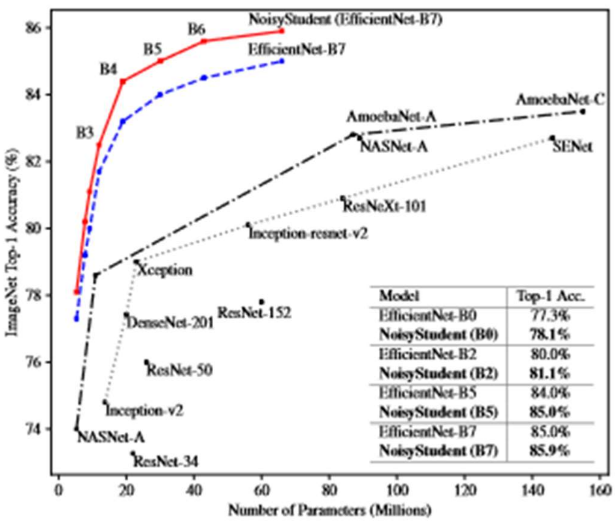
System & training set	Train Frame Accuracy	Test Frame Accuracy
Baseline (100% of training set)	63.4%	58.9%
Baseline (3% of training set)	67.3%	44.5%
Soft Targets (3% of training set)	65.4%	57.0%

Table 1: Improvement for using soft labels [1]

Teacher	Top-1 Error (%, Test)	CE (Train)	KD (Train)	KD (Test)
ResNet18	30.57	0.146	2.916	3.358
ResNet18 (ES KD)	29.01	0.123	2.234	2.491
ResNet34	30.79	0.145	1.357	1.503
ResNet34 (ES KD)	29.16	0.123	2.359	2.582
ResNet50	30.95	0.146	1.553	1.721
ResNet50 (ES KD)	29.35	0.124	2.659	2.940

Table 2: The early stopping effect

Table 2 shows that KD using early-stopping prevents the student model from suffering from degraded performance when classifying on ImageNet.



Model	Top-1 Acc.
EfficientNet-B0	77.3%
NoisyStudent (B0)	**78.1%**
EfficientNet-B2	80.0%
NoisyStudent (B2)	**81.1%**
EfficientNet-B5	84.0%
NoisyStudent (B5)	**85.0%**
EfficientNet-B7	85.0%
NoisyStudent (B7)	**85.9%**

Figure 3: The improvement by adding noisy to train the student model

Figure 3 shows adding noise in the student models can improvement performance. This improvement occurs across all model sizes for EfficientNet.

6. Discussion

Does knowledge distillation really work? There is no doubt that it is useful because it often improves the generalization performance of student models. However, Samuel Stanton et al. [5] in their paper notices that obtaining a fidelity student model is often difficult. The authors have done many experiments and the results show that many student models do not learn the knowledge sufficiently well from the teacher model. They also indicated that optimization of knowledge distillation sometimes converges at the suboptimal. How to find an optimal solution becomes the most important topic during building a good KD model.

How to choose a suitable distillation temperature is also a question that needs to be considered. If the requirement is to let the student model learn more information from negative labels, giving a higher temperature value is a suitable choice. If reducing the interference of negative labels in the student model is the goal, a lower temperature value should be used. However, the selection of temperature is also related to the size of the student model. If the student model contains fewer parameters, a low temperature value should be given because the model with a smaller number of parameters cannot learn all the knowledge. Temperature is one of the hyperparameters that need to be tuned. Building an efficient method to tune the temperature may help improve the optimization.

References

[1] G. Hinton, O. Vinyals, and J. Dean, “Distilling the Knowledge in a Neural Network,” 2015, doi: 10.48550/ARXIV.1503.02531.

[2] R. Anil, G. Pereyra, A. Passos, R. Ormandi, G. E. Dahl, and G. E. Hinton, “Large scale distributed neural network training through online distillation,” 2018, doi: 10.48550/ARXIV.1804.03235.

[3] J. H. Cho and B. Hariharan, “On the Efficacy of Knowledge Distillation,” 2019, doi: 10.48550/ARXIV.1910.01348.

[4] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, “Self-training with Noisy Student improves ImageNet classification,” 2019, doi: 10.48550/ARXIV.1911.04252.

[5] S. Stanton, P. Izmailov, P. Kirichenko, A. A. Alemi, and A. G. Wilson, “Does Knowledge Distillation Really Work?,” 2021, doi: 10.48550/ARXIV.2106.05945.