Learning Distillation Model for Image Data

Dean Chao-Kai Chung   
Department of Computer Science  
University of AucklandAuckland, New Zealand

This project aims to perform learning distillation, in order to train up a ‘student’ model, that has been fed with pre-trained ‘teacher’ model. It helps improve model performance, all while keeping the expenses of training a model to a minimal by transferring the soft output probabilities of the teacher model, into the student model. This was done through training the teacher model on a few epochs, then freezing the weights to prevent real-time updating during student learning. The results showed good performance with the student model, whilst reducing time and resources spent in training the models. Displaying how effective and efficient knowledge distillation can be.

Keywords—PyTorch, Deep Learning, Knowledge Distillation, Student-Teacher Model, Image Classification

# Introduction

Image Classification models have been notorious for expending a lot of time and resources to train, due to the amount of data that represents a singular image. You can’t have an accurate model if it’s not complex or time consuming. However, ‘Knowledge Distillation’ is a technique, that transfers knowledge from a large pretrained ‘teacher’ model to a smaller ‘student’ model. KD allows the student model to learn and predict correct labels, with the help of the teacher’s soft prediction patterns, improving overall performance while using less computational costs. Our objective is to implement KD on PyTorch, and visualize the teacher and student model’s performance on their loss and accuracy.

The rest of this report is outlined as such: II. Reviews of literature that has done existing work on KD and Image Classification. III. Detailing my approach, such as preprocessing, model training, KD process, hyperparameter choices. IV. Our results of teacher-student model performance in accuracy and loss. V. Discussions on results, reflection of findings, and potential improvements/limitations. Lastly, VI. Is our conclusion

# Literature Review

## The Core Foundation of Knowledge Distillation

Knowledge distillation was first introduced by Hinton et al. [1],

## Feature-Based and Intermediate Representation KD

Romero et al. [2] extended the KD framework with Fit

## Attention-Based Knowledge Transfer Zagoruyko and Komodakis [3]

## Deep Networks Urban et al. [4]

## Studies on Distillation Strategies Heo et al. [5]

## Online Knowledge Distillation with Diverse Peers Chen et al. [6]

## What is the relevance of Literature Review to This Project

# Methodology

1. Dataset and Preprocessing

The dataset used for this project was a 10-class image dataset, containing RGB images of varying sizes. Images are loaded into memory, handling both unstructured data (test), and structured data (train/validation). Once test data has been extracted into a list of images, we then extract both the list of images, and the class labels, which are named accordingly by their subdirectories. Image format has also been formatted to BGR (Height x Width x Channel).

Images are then resized into 224x224 squares. This is done by setting the shortest side of an image to 224, correct the latter size by the short side’s new scale, and cut off excess of the latter size from the middle of the image, to preserve the important areas of the image, which is most often from the centre.

Images are then all divided by 255, to keep all RGB channel values ranging from 0 to 1, as most ML models operate best when known numerals are between 0 and 1. I have converted then into float32, and transposed image data from NHWC, to NCHW (1, 224, 224, 3) to (1, 3, 224, 224).

For each 3 colour channels of an image, I computed their mean and standard deviation, which is then all saved into a ‘.pt’ file for later use, along with the labels.

1. Teacher Training (Resnet 34)

We used a pretrained ResNet34 architecture to be trained on our dataset. It’s final layer has been modified to output 10 classes. SGD was utilized as optimizer, as it usually works well with convoluted neural networks, and don’t overfit as easily as Adam optimizers. Learning rate was set at 0.1, often a good starting point. Epochs at 5, as this architecture is large and complex, making it expensive to train for many epochs. Batch size set as 64, as larger would take up too much memory, and any smaller reduces the model’s ability to learn correlation between other classes. Model was trained on cross entropy loss, which is standard for multi-categorical classification.

After training for 5 epochs, the model was able to achieve 54.1% accuracy on validation dataset.

# Results

--

##### References

--

**IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove template text from your paper may result in your paper not being published.**