Reproduction of SNOW paper

Deep Learning with Multiple Tasks 2020/2021

Wiktor Daniec, Paweł Goliszewski, Konrad Wójtowicz

1 Introduction

Creating models which will be able to handle multiple tasks is current object of research in machine learning field. People usually do not need thousands of examples (like many ML model do) to learn how to classify objects into some number of classes of object - they usually use information learned through entire life and adapt quickly to new tasks. Idea is to use 'knowledge' from model trained on the other task to learn faster how to manage other task. Authors of paper [1] tackle the problem of image classification. They use pretrained model with smaller model connected to it to perform classification task on other classes. Our goal was to try to reproduce their results.

2 SNOW model architecture

This model uses 2 models connected via operation called Channel Pooling which will be described later. One model (called source model) is ResNet50 pretrained on ImageNet dataset with frozen weights. It is connected after each convolutional layer with second model (called delta model) which is smaller version of Resnet (in our experiments channels reduced by 8) with randomly initialized and trainable weights. We put image tensor to both model inputs, and we use only the output of the delta model into consideration.

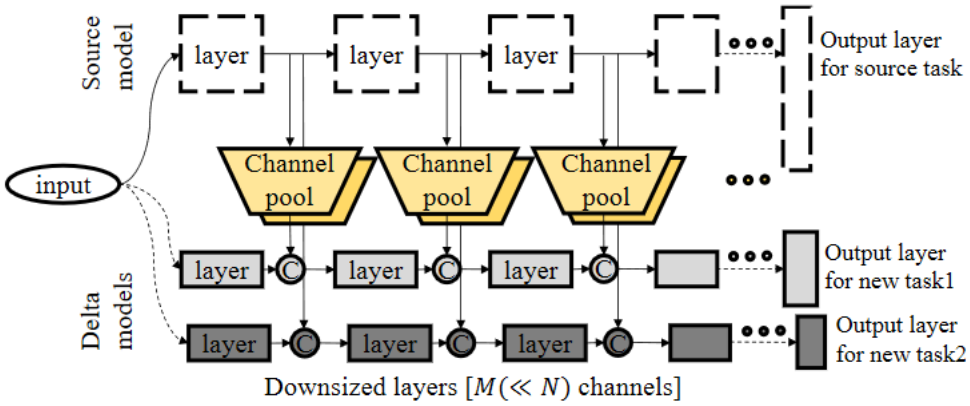

Figure 1: Illustration of SNOW model

2.1 Channel pooling operation

Channel pooling (CP) operation behaves differently during training stage and the evaluation stage.
CP layer contains weight vector w of size N where N is also number of input tensor layers. If it is training time we sample vector size N from normal distribution of mean 0 and standard deviation σ, and add it to the weights vector obtaining $rand_v$ vector. We take indexes of the M biggest values of $rand_v$. We take M channels of the input given by those indexes multiplied by w element-wisely. We concatenate the result to the corresponding input int the delta model.

3 Transfer learning

We use some standard transfer learning techniques for comparison. Those techniques are:

- full fine-tuning (FT) - We use pretrained ResNet50, and we modify last layer to adjust number of classes to a new task. Then we train network with all trainable parameters on a new task.
- feature extraction (FE) - Similarly as above, but weights of first 3 layers are frozen.

Algorithm 1: Channel Pooling pseudocode

w: channel pooling weight ($N \times 1$ vector);
x: input tensor (with N channels);
Function `ChannelPoolingFroward(`x`):`
  if *training* **then**
    $rand_v = w + \mathcal{N}(0, \sigma)$;
    $\_, idx = \text{topK}(rand_v, M)$;
    $selected\_weights = w[idx]$
  else
    $selected\_weights, idx = \text{topK}(rand_v, M)$;
  end
  $x = x[idx]$;
  $x = x \bigotimes selected\_weights$;
return x

- finial layer only (FO) - Similarly as above, but only last linear layer is not frozen.

4 Experiments

4.1 Datasets

4.2 Transfer learning

4.3 SNOW

4.4 Throughput

4.5 Visualization of filters

5 Conslusions

References

[1] https://openreview.net/pdf?id=rJxtgJBKDr
Chungkuk Yoo, Bumsoo Kang, Minsik Cho, SNOW: Subscribing to

Knowledge via Channel Pooling for Transfer & Lifelong Learning of Convolutional Neural Networks, ICLR 2020