Analysis and Implementation of Transfer Learning

Ragini Sistla and Wenbo Tian

1 Introduction

THe idea behind this proposal is to explore the existing convolutional neural networks and use these existing ones in building one of our own. Till now, we have only seen the networks where the training data and the future data belong to the same distribution or domain. What if the future data comes from a different distribution or different domain altogether?
In short, we would like to build a model on top of existing models instead of creating a model from scratch. By doing so, we can save the time of training a model fully from scratch. The motivation behind this proposal is to explore and understand "transfer learning". Many deep neural networks trained on natural images exhibit a trait which is common among most of the images. For example, detecting the edges and colors of the images which happen at the initial or low levels of the convolutional neural network would be common to any image. Hence, we can use the weights of some of the layers or can use the entire model to build a new model.

The major concern here is to decide on what to transfer or what to use and what not to use from the old models. Lot of research is going on currently in this field and we would read through the literature to understand the concerns and challenges in implementing it.

2 Plan of Action

2.1 Direction of Analysis

Here, we have following things in mind for now. Firstly, we would take the existing convolutional neural networks such as Alexnet or Lenet or Googlenet and would remove the last fully connected layer and would take the rest of the model as it is and would train the new model. All this, we are doing it under the assumption that the new data we are using to train is very much similar to the data that was used in training the old model. This can be understood in two ways; one way is either the new data would have been added as part of training data of old model itself. Another way is to think that we are saving time in training a single model for a longer time as we have a new model which is being trained on the similar data in less time.

Now, what if the new data is not similar, we would mostly take initial layer(first hidden layer) from the old network and build a network from there. If we find that the accuracy is better with our new testing data, we would go ahead and take another layer from the old network. We would do this till we get an optimal accuracy, which means when the accuracy drops, we would not take any more layers from old network.

Well, by doing so, we might end up training our new model several times and it might some times be better to have it from scratch. But we think using known weights compared to random weights would be always better.

We would also need to think on the learning rates and fine tuning of the new network. If the training data set of the new model is not totally different from the training data set of the old model, we would keep the learning rate low and fine tune the weights as we are expecting the model to be similar to the old network. Major concern here is how to fine tune the weights if we have the training data set of the new model totally different from the training data set of the old model.

2.2 Research Question

We need to figure out which layers of the old network should be used as part of our new model. For this, we need to understand what happens at each layer of the network.

2.3 Expected Experiments

Experiment 1:

1. Firstly, we would collect the data which falls into the same domain as that of the training data set of the old model. We have not yet decided on the dataset. We should select the size of the data set in such a way that we avoid overfitting.
2. We would take one of the convolutional neural network either Alexnet or Lenet.
3. Remove the last layer, and train the new model by fine tuning the weights.
4. Calculate the execution time in training the new model. Also, while fine tuning we would apply several optimizers and momentum techniques and check the execution time.
5. Check the accuracy of the testing data set on the new model.
6. We would use tensor flow framework to build our new model

7. Accuracy should be as high as old model and execution time should be lower than the old model.

Experiment 2:

1. Collect data set of a completely different domain.
2. Take layer one of old network where usually the edge detection happens which is not specific to any dataset.
3. Train the new model and see the accuracy
4. Take few more layers from the old network, may be layers till 2nd pooling and repeat the process.
5. Find the execution time and accuracy of the newly trained network.

2.4 Project Members and Task Allocation

Ragini:

1. Read through the current reaserch going on in transfer learning.
2. Read through the reasearch papers to understand the views on how transfers can be done.
3. Analyse what happens at each convolutional layer of old network (Alexnet or Lenet).
4. Conduct Experiment 2

Wenbo:

1. Read through the research papers to understand the challenges in implementing transfer learning
2. Analyse on various paramters such as learning rate, optimizers, regularizers, activation functions, batch normalization etc. Also, would check if any pre-processing is required on the collected data set.
3. Conduct Experiment 1

2.5 Literature Survey

The eye disease Diabetic Retinopathy is a common cause for eye loss. Sreening diabetic patients using fluorescin angiography images can potentially reduce the risk of blindness. Current trends in the research have shown that using transfer learning in neural networks, the images of the eyes were recogined with an accuracy of about 90% and could improve the DR prediction. **This was achieved by having a traning data set of only 140 images**. This shows that even with small sample set we are getting an accuracy of 90% by using transfer learning in neural netwroks.

2.6 Thoughts

Caffe library has a Model Zoo where people share their respective network model weights. We feel this is a good place to find the initial weights for our model if our dataset is not going to be similar as that of ALexnet or Lenet.

References

[1] Yosinski, Jason, et al. "How transferable are features in deep neural networks?" *Advances in neural information processing systems*. 2014.

[2] Sharif Razavian, Ali, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. "CNN features off-the-shelf: an astounding baseline for recognition." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 806-813. 2014

[3] Pan, Sinno Jialin, and Qiang Yang. "A survey on transfer learning." *IEEE Transactions on knowledge and data engineering* 22, no. 10 (2010): 1345-1359.

[4] Mesnil, Grégoire, et al. "Unsupervised and Transfer Learning Challenge: a Deep Learning Approach." *ICML Unsupervised and Transfer Learning* 27 (2012): 97-110.

[5] Donahue, Jeff, et al. "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition." *Icml*. Vol. 32. 2014.