How Transferable are Features in Deep Neural Networks?

1 Citation

Yosinski, Jason, et al. ”How transferable are features in deep neural networks?.” Advances in neural information processing systems. 2014.

http://papers.nips.cc/paper/5347-how-transferable-are-features-in-deep-neural-networks.pdf

2 Abstract

Most CNNs learn Gabor filters (generally applicable to vision tasks) in their first layer, but their last layer (the softmax) is highly specific. Which layers in the network are transferable and which are not? We investigate this and the key limitations to transferability: (1) specialization to the source task and (2) difficulty optimizing neurons in two layers that are co-adapted to work together and (3) the source and target tasks are too different.

3 Introduction

The first CNN layer learns Gabor filters (if not, you’ve probably got a bug). The last layer tends to be specific to the dataset. How specific or general are the layers in between? Additionally, when you transfer some layers, should you fine-tune them or leave them frozen and only train later layers?

4 Generality vs. Specificity Measured as Transfer Performance

We split the ImageNet dataset into two parts (A and B), each of which has half the classes. We do this split randomly. However, this results in similar classes in both datasets (e.g. ”labrador” in A and ”greyhound” in B), so we also experiment with a split between man-made and natural objects.

For a given split, we consider the n^{th} layer ($n \in \{1, 2, 3\}$). We train a **selffer** network BnB where we train on B, freeze the first n layers, re-initialize the rest, and train again. We also train a **transfer** network AnB where we train on A, freeze the first n layers, re-initialize the rest, and train on B. The idea is that if AnB does just as well as BnB on dataset B, then the features in layer n generalize well.

We also consider the situation where we don’t freeze layers and train the network end-to-end. This gives us BnB+ and AnB+.

5 Experimental Setup

Our base network is AlexNet.

6 Results and Discussion

Training on B alone (i.e. baseB) gives a slightly better error than state of the art. Although you'd expect the error rate to be higher because we have half as much data in each dataset, we also have half the number of classes, so the learning problem is easier.

For BnB, performance is identical to baseB if $n = 1$ or $n = 2$, indicating there's no fragile co-adaptation (i.e. neurons in adjacent layers work together, which breaks when you freeze one of the layers) in these layers. For $n = 3, 4, 5, 6$, performance is worse, indicating fragile co-adaptation. For $n = 7, 8$, performance is good, indicating no fragile co-adaptation here.

BnB+ does just as well as baseB, indicating that end-to-end fine-tuning avoids the problems caused by fragile co-adaptation above.

AnB does as well as BnB for $n = 1, 2$, indicating these features generalize well. $n = 3$ has a slight drop. $n \geq 4$ have a larger drop (smaller n suffer from co-adaptation and larger n suffer from specialization). AnB+ does better than AnB, which indicates that fine-tuning end-to-end is better than freezing the original layers.

THe AnB and AnB+ are comparatively worse when we do a man-made/natural split instead of a random split. This shows that transfer learning works better when the datasets are similar.

Previous work showed that using random filters works almost as well as learned filters. We found this does not hold on our network. Maybe it only works for shallow networks?

7 Conclusions

When transfer learning, don't use the last few layers and do end-to-end fine-tuning. You'll get better results if your target task is similar to your source task.