Better techniques for transfer learning

Ranti Dev Sharma rds004@ucsd.edu
Under Guidance of Dr Gary Cottrell

It is not yet clear whether brain does backpropagation or something similar to that, but one thing is clear: brain does adjusts its weights over time, while learning new concepts. In this paper we explore how to decide which layer weights should be changed. In other terms when we need to learn new pattern or new task, which layer should be trained. Since training of weights is both time consuming and energy consuming process, my hypothesis is that brain does something similar to save time and energy.

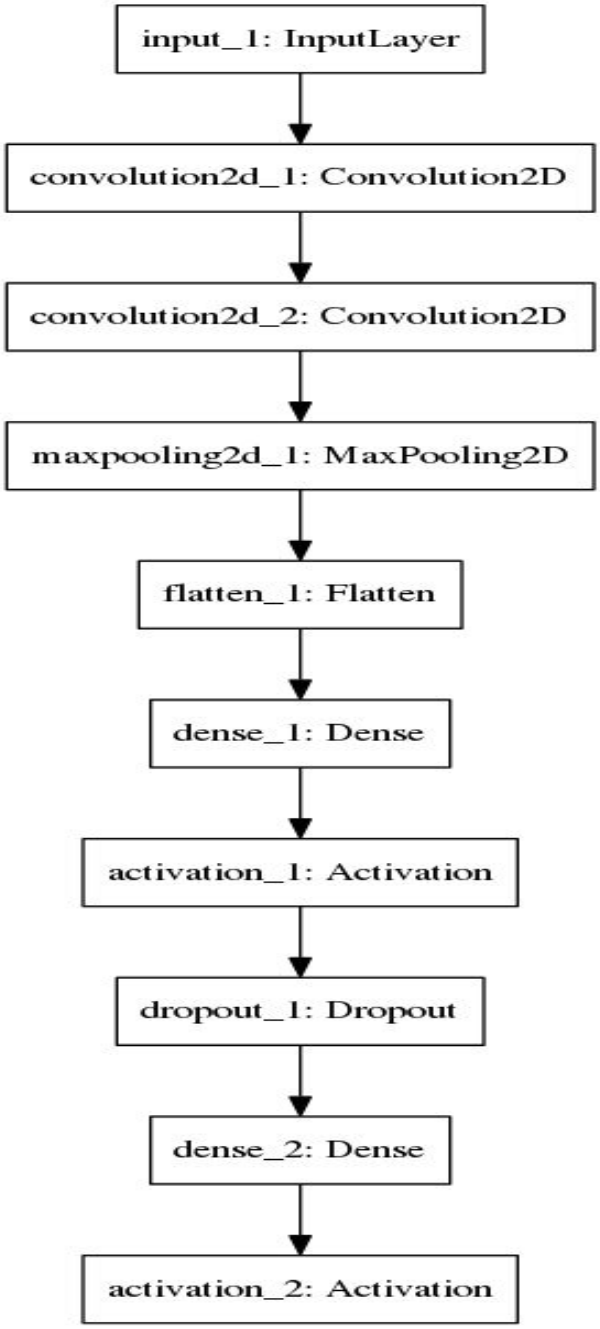
The above mentioned methodology can have many practical applications. As deep learning is progressing, networks are getting deeper. For example Resnet has more than 50 layers and we don't know how many layers future would bring us. In the current state of art methods to train our network on some new set of (unknown) patterns, we train topmost fully connected layers or train some top k layers (where k is some small integer) or maybe train the complete end-to-end network sometimes. It may be possible that for some unknown patterns network won't require training at all because it has already learnt all necessary filters to classify it. Out of all these methods fully trained network is better but is time and energy consuming.

A simple approach to make learning more efficient could be devised by looking at layer wise activations produced by both known and unknown patterns to network. Basically we propose to do two experiments to know the state of layerwise activations for both known and unknown patterns.The details of two approaches (clustering and reconstruction) are mentioned below:

1) **Clustering on activations**: In this experiment we observed activations produced by both known and unknown patterns layer wise. As these activations are higher dimensional spaces we observe them using t-sne. As lower layers learn most basic filters like edges and curves, so even for unknown classes they won't need any retraining. But since upper layers learn more complex patterns specific to problem, hence those learnt features may not understand unknown patterns and might produce activations which are more spread. But clustering methods doesn't quite give quantification of diverseness and also I don't think our brains are capable of doing this to learn which layers need more training.
2) **Reconstruction on activations**: In this approach, we used two basic techniques - one is gradient based and other is encoder-decoder structure based. Firstly we tried gradient based technique to reconstruct image from it's activations from different layers. We compared the image reconstruction loss of both known and unknown patterns layerwise to formalize which layers required more training. Further we tried encoder-decoder type methods for reconstruction.

By doing these experiments we gained more in depth knowledge of what these layers are doing and why they are doing. Intuitively, Reconstruction based approach should give better results because it is similar to the way humans reconstruct images from our memory.

Structure of neural network : We are using simple two layer convolutional neural network. We trained the network for digits (0-4) and we call them known patterns. We call digits 5-9 as unknown patterns.

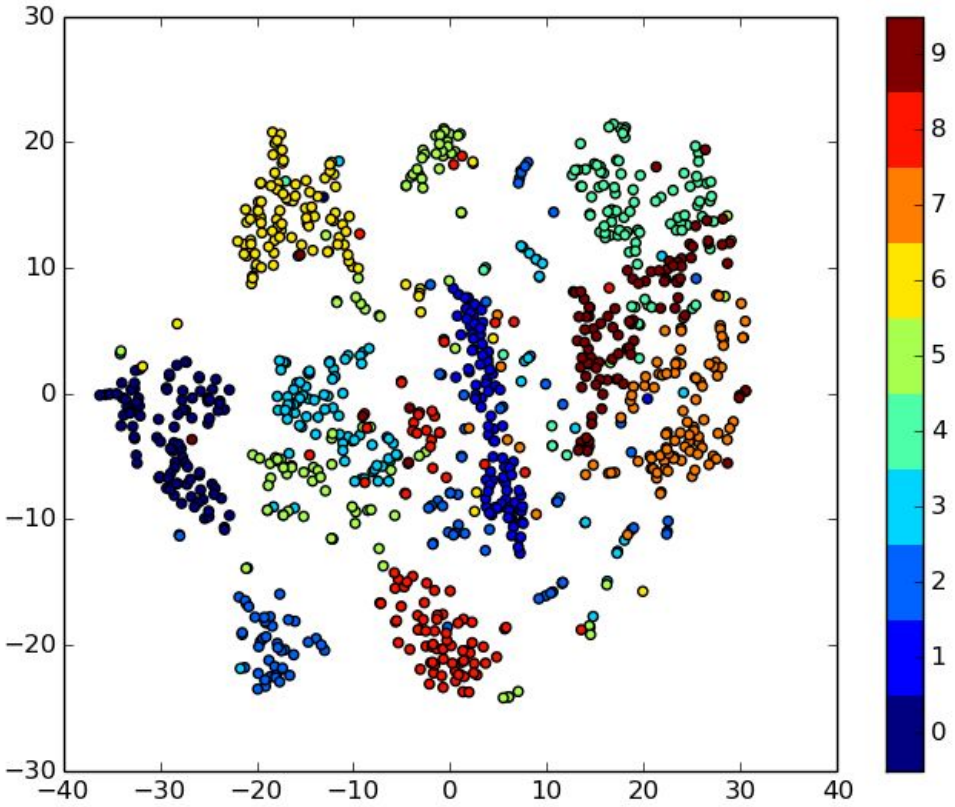


Experiment 1: Clustering based method:

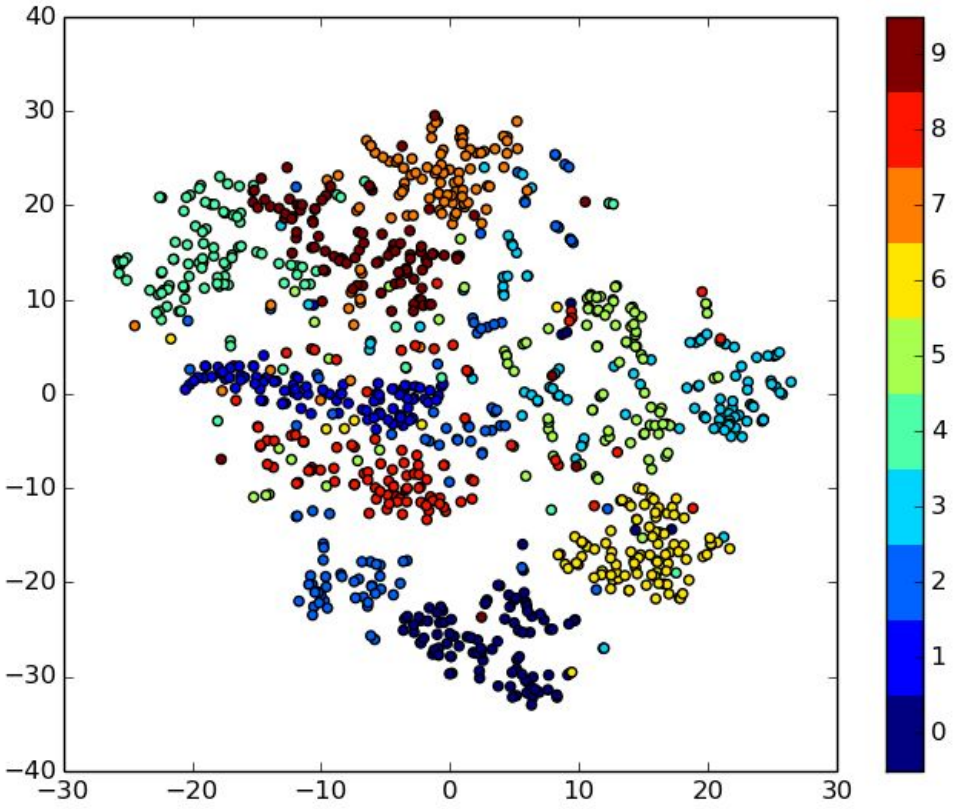
We trained our two layer CNN over digits from 0-4 and obtained activation at CNN layer 1 and layer 2 for digits 0-9 with this network. We use t-SNE to reduce the dimensionality of activations layer to 2D space to visualize the activations more clearly.

T-sne results:

Convolutional Layer 1 activations t-sne result:



Convolutional Layer 2 activations t-sne result:



Discussion: For known patterns activations are pretty much clustered but for unknown patterns activations are more spread. Clustering results show that as we go deep layer wise for unknown patterns their clustering is becoming more spreaded. One particular result is that for layer 2 activations of digit 8, digit 8 is highly spread whereas for digit 9 and digit 7 they are still clustered. One more important thing to see here is that somehow network is associating digit 8 with known pattern digit 0 which are somewhat similar in shape.

Experiment 2: Reconstruction using Gradient based techniques:

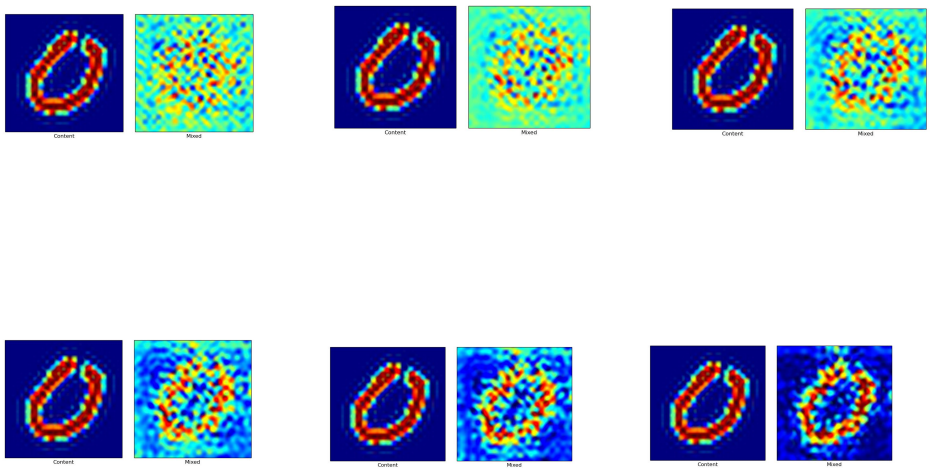
This is really easy scheme. Here let's say we wanted to reconstruct pattern 'p'. We will get activations of this pattern at layer 'l'. After that we will input a random image into same network and obtain its activation at layer 'l'. Then we will take MSE loss of this new image and pattern 'p' at layer 'l'. Once we have loss we will get derivative of this layer with respect to new image and change values of new image. One important thing to note here is that gradient based reconstruction method is not possible in brain whereas autoencoder based methods may be possible in brain. Using that gradient we update new image. The results obtained through gradient based reconstruction technique is shown below.

Digit	Layer 1 reconstruction error	Layer 2 reconstruction error	Layer 2 error Without any training
0(known)	8018.10351562	10714.22363281	11414.13769531
1(known)	5083.41992188	7549.57910156	8240.32128906
2(known)	7755.01660156	10549.84765625	10980.546875
3(known)	7366.31933594	10222.41210938	10769.54296875
4(known)	6782.44726562	9637.2109375	10000.38964844
5(unknown)	6271.58544922	8856.26660156	9862.296875
6(unknown)	7448.06396484	10270.82226562	10581.93261719
7(unknown)	6603.953125	9235.93261719	9618.20898438
8(unknown)	8119.54541016	10832.08398438	11293.93554688
9(unknown)	6839.11328125	9510.26074219	10074.81542969

Discussion:
These results clearly show that as depth is increasing error is increasing, But these results does not differentiate between known and unknown patterns as reconstruction error for fully random layer 2 and trained layer are similar.This similarity shows that this gradient based method is not

a good choice. One thing is clear that gradient based reconstruction is not giving us the results using which we can formalize layer.

Gradient based reconstruction for fully random 2 layer CNN at layer 2:



VGG-19 architecture based reconstruction error:
After observing these results, we experimented with two networks. First, VGG-19 style architecture trained on images and the other network with architecture same as VGG-19 but with random weights. For these networks I observed that I see some differences in results only after 4th layer. For layers 1-3 results of fully trained and fully random architecture is same. Whereas when depth became 16, randomly initialized VGG-19 network gave very bad results as compared to pre trained weights initialized VGG-19 network. These results show that these

gradient based method will give better results only with highly deep networks whereas our MNIST based architecture is just 3 layer deep. Results are shown below.

Layer 1 reconstruction for Random weights VGG-19 network:

Layer 2 reconstruction for Random weights VGG-19 network:

Layer 16 reconstruction for random weights VGG-19 network:

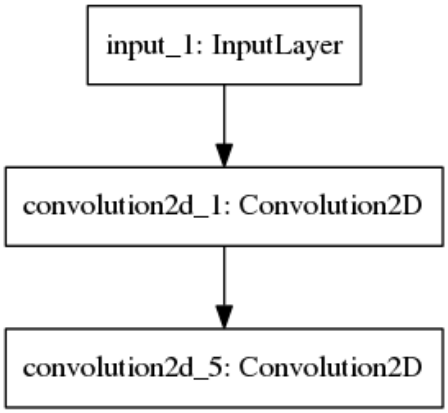
Layer 16 reconstruction for pretrained VGG-19 network:

Experiment 3: Reconstruction using Autoencoder based models:

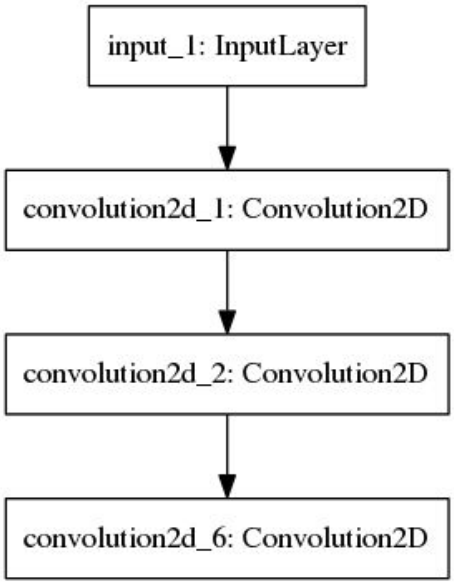
Autoencoder based reconstruction is very simple. What we do this for each convolutional layer in classifier we train a decoder based network for reconstruction. Important point is that classifier and decoder network both are only trained on known patterns. For given MNIST two layer convolutional neural network I trained two decoders. While training decoder part, all layers of classifier are fixed.

Autoencoder structure:

Autoencoder 1:



Autoencoder 2:



By doing experiments on autoencoder network we get following results.

Digit	Layer 1(reconstruction error)	Layer 2(reconstruction error)
0(known)	76.24341583	144.20317078
1(known)	37.76148605	71.99356079
2(known)	89.99630737	175.07717896
3(known)	87.36468506	167.58268738
4(known)	84.15545654	157.71461487
5(unknown)	88.88010406	184.75756836
6(unknown)	76.63885498	170.90098572
7(unknown)	63.3898735	118.9193573
8(unknown)	98.47270966	213.88096619
9(unknown)	71.3926239	143.69009399

Discussion:
These results clearly show that as depth is increasing reconstruction error is also increasing. As you can see in layer 1, reconstruction error of digits 3 and 4 is greater than digit 6, whereas in layer 2 it is vice versa. Which clearly shows that layer 1 does not need further retraining but layer 2 may need some. But these results also show that for digits 5,6 and 8 we might require retraining of layer 2 as reconstruction error is higher than known patterns. But for digits 7 and 9 no retraining is required as their reconstruction error is showing that features learn't while classifying 0-4 digits is enough for classifying digit 7 and digit 9. On the other hand, high reconstruction error for digit 8 shows that digit 8 will certainly need more retraining. Note that Higher reconstruction error for digit 5,6 and 8 is consistent with clustering results obtained by t-SNE experiment.

Conclusion: By performing all these above experiments we obtained some intuition on features learnt by layers. We also tried to formalize the training required for transfer learning for unknown patterns. In all above experiments, Autoencoder based networks gave best results. Also autoencoder results align with observations obtained by t-SNE clustering experiments. In all we can propose that using autoencoder based reconstruction we can formalize transfer learning for unknown patterns. Our results show that for some unknown patterns we may not need retraining at all whereas for some other unknown pattern we may require further retraining of convolutional layers. Our results also show that we might need to retrain upper layers or lower layers depending upon cases . As MNIST dataset is really simple dataset to get more authentic results in future ImageNet dataset should also be tried. Also Gradient based reconstruction based methods may also perform well on that big networks for ImageNet.