Evaluation of Fully-Connected Layers in Convolutional Neural Networks

Rhys Lee, Yong Jae Lee
University of California
Davis

Abstract

This paper analyzes the utility of the most basic components of Convolutional Neural Networks, the Fully Connected layers, in comparison to the convolutional layers. I analyzed the Fully Connected layers from the implementation perspective and the biological perspective. Through the analysis I tried to reveal some of the common beliefs that I deem questionable, though some might have taken them for granted. The argument I am trying to establish in this paper is that the cost of Fully Connected Layers can be greater than their actual utility, unless we stop taking them for granted, and start actively exploring the balances between the trade-offs of efficiency, over-fitting, and performance in each specific network. To reduce the cost of the Fully Connected in layers, I proposed the approach of making neural networks deeper, and thinner, to effectively reduce the implementation cost of Fully Connected Layers and chance for over-fitting. To support my arguments, I designed two sets of experiments each reducing over 90% of the size of the state of art convolutional neural network at a marginal cost on performance.

1. Introduction

We might be taking for granted the utility of Fully-Connecter Layers in neural networks. Despite enormous amount of papers have been published on Convolutional Neural Network (CNN), contributing to the development of models with higher accuracy and better applicability, few papers could be found to elaborate on one of its most fundamental component, the Fully Connected Layer (FC Layer).

Yet, the cost of FC Layers in the most common CNNs is not negligible at all: they usually render the entire CNN bulky, prone to over-fitting, slow to train, and etc. To address the elephant in the room, I attempt to provide a more comprehensively evaluation of the FC Layers through this paper, not only to evaluate the effectiveness of their implementation, but also their metaphorical representation of its biological counterparts, in order to better put things into perspective and to justify certain design choices we make in CNNs.

1.1. Fully-Connected Layers from an implementation perspective

In CNN, FC layer is a special type of convolutional layer, where the size of its receptive fields is equal to the size of the input space which it samples from. Thus, each output of a FC layer is a linear combination of all the inputs in the input space.

An FC layer with N as its input and M as its output is usually implemented with an N-by-M matrix of adjustable/learned weights and a matrix of bias with the same size as M. Often the size of M is not very large, however the size of N can be quite variable.

Often FC Layers occupy a large portion of the total parameters in a CNN, thus rendering the network very bulky and prone to over-fitting [2]. For example, in the Alex Net (Krizhevsky et al., 2012), according to Krizhevsky et al. (2012) there are in total about 60 million parameters in the network. Given the architecture of their CNN, it's easy to calculate that at least 50,550,784 parameters are in the FC Layers. That is to say at least 5/6 of the parameters in Alex Net is in FC layers.

To combat the over-fitting, some have introduced methods such as the drop-out method to remedy such bulkiness problem within FC layers. However, most of these methods seem reluctant to actually reduce the number of parameters in FC layers. So this brings up the question: is FC layer really worth its cost?

1.2. Why FC Layers?

FC layers appear in almost every CNN implementation, with few exceptions such as the Fully Convolutional Neural Network [7] for pixel-wise semantic segmentation. Such fact strongly implies that FC layer is regarded by most as an indispensable component for CNNs. A possible side effect of such reality is to discourage people from mindfully thinking about questions such as whether there are better alternatives than adding an FC Layer, how FC layer is going to help with the CNN to achieve its specific goals, and how

to modify the FC layers to balance the tradeoffs and better pursue its purposes.

The important step towards a potentially better solution is to better understand the current model, its utility and its shortcomings.

1.3. The utility of FC Layers

Non-linearity

First, it is generally believed that FC layers can add to the non-linearity of the network. There are lots of cases where two FC Layers are used in sequence in a single network in order to add more linearity, in which case Sigmoid function or ReLU must be implemented in between the two FC Layers, otherwise the two FC Layers will be equivalent to one FC Layer except with more parameters to tune. This is simple to understand: the FC Layer output is the linear function ***f*** of all the inputs; When we apply another FC layer ***g*** directly to the output of the previous FC Layer, taking ***f(x)*** as input, the result of the two layers becomes ***g(f(x))*** which is also a linear function of the original input x, and thus equivalent to applying just one FC Layer ***u(x)***.

In comparison, the convolutional layers add non-linearity partly from spatial discrimination. By repetitiously applying convolution on local regions of the input space and neglecting data outside of the region, each output of the convolution layer is non-linear to the whole input. There are other ways to introduce more linearity with convolution layers, such as adding pooling layers.

Invariance

FC layers can also induce invariance to the model, which helps CNN to learn useful invariance within each class of inputs. However, due to its receptive field size, FC layer is not as discriminative on spatial information. Thus, we can say that FC layers help introduce global invariance, whereas convolution layers introduce invariance on local patterns.

Consequently, FC layers' outputs will lose the explicit spatial information encoded in the outputs from convolutional layers, whereas the convolution layers' output will lose the exact local patterns.

Transition

The most intuitive function of FC Layer is that they provide a transition of formats, usually from the spatial discriminative inputs to the outputs with implicit correlation to spatial information encodings. Such utility is quite useful since the number of spatial location categories of the input usually don't match the number of desired outputs, such as the number of different classes. Directly correlating the spatial discriminative outputs of a convolution layer to each class output is just equivalently creating a FC layer. Even if they do match, how good the performance will be by taking each input directly as the class output is unknown.

1.4. The problems of FC Layers

Over-fitting

The most prominent problem of FC layers is their tendency to over-fit. As we have showed with the example of the Alex Net, most parameters in some CNN architectures are contributed by the FC layers. More parameters to train can mean more computation, but not necessarily longer time to converge, which depends also on the architecture, though models with more parameters usually take more time to train than smaller ones.

More importantly, more parameters can mean that the model will make its decisions accounting for more details and relations. This may not be a good thing when, in reality, only a few key features are actually relevant for a certain decision. In such cases, the model that has too many parameters have so many extra parameters to train, thus the network will eventually have to look for relations that don't actually exist but justifiable enough in the limited amount of training data. In such situations, the model with more parameters might show a higher accuracy than smaller models on the same training and validation sets, but are hardly generalizable outside of the training data.

Such over-fitting problems sometimes can appear as a slight increase of the performance on the limited amount of data, but at the cost of a lot more parameters and potentially a more severe lack of generalizability.

Therefore, I am skeptical about increasing the total number of parameters of a model by many times just to get a marginal increase in performance. However, in many papers researchers showed preference on such marginal accuracy because the over-fitting problem is much subtler to detect/discuss, meanwhile any increase in performance no matter how marginal seem to add to the credibility of the paper.

The trade-off

Intuitively, if we can reduce the total parameters of a model by a major portion (maybe above 50%, the higher more powerful the argument) without significant loss in the performance, then we can argue, in some degree, that the model shows problems of over-fitting.

To reduce the size of a model, the most straight forward and efficient way may be to **shrink the size of the FC layers** which occupy most of the parameters in many models. We may better justify our reduction of FC Layers by referring to our biological neural networks.

1.5. The biological perspective

Artificial Neural Networks are inspired by our biological neural networks. There are about 100 billion neurons in an average human brain. Each neuron can be connected to up to 10,000 other neurons through synapses. Biological neural networks are very complex to be accurately described through mathematical models.

In a biological neuron, the most relevant components are the soma, the dendrites and the axon. Each neuron's characteristics of activation is highly unique, influenced by the structure of its dendrites, the relative locations where axons from other neuron communicate to this neuron's dendrite and etc. To describe the behavior of the activation of a specific neuron, there are ways such as neuron encoding, describing P (response | stimulus) on a range of inputs, which is comparable to the convolution operation we apply to inputs with weighted filters in a CNN.

However, just to describe the basic membrane of a neuron we need mathematically heavy models such as the Hodgkin-Huxley equation for the ion channel opening, which is very expensive to simulate in full detail for Artificial Neural Networks, so we simplify the model and at the same time we discard many very important characteristics such as the temporal characteristics.

The point is, CNN is not very accurate model of biological neural networks. The frontal part of our visual pathway from retina to LGN is probably the closest thing the CNN can get to. In this sense, we can find similarities between the retina cells and the convolution layers, both of which have only feed-forward connectivity and are specialized in discriminating the spatial information of the inputs (which can probably tell us why CNN is so good at image classification).

The convolution filter is very similar to the neuron groups in the retina which, structured in very specific spatial combinations, react only to certain very specific patterns. The complicated connectivity structures between these neurons are modeled with weights and bias, which are almost good enough models since we don't consider much about temporal characteristics of stimulus for visual tasks. In such comparison we can justify the significance of convolution layers in CNN.

However, the fully connected layers are not so conveniently justifiable since there is not really any counterpart close enough to them in our biological neural network. Instead, FC layers are more of a compromise between computational viability and the very complicated visual pathway we try to learn after. This doesn't mean that the FC layers can't be better at the job, but the necessity of FC layers just aren't justified by our powerful biological neural network.

Therefore, when I design my experiment to shrink the network model, as I choose between reducing the convolution layers and reducing the FC layers, I feel more justifiable to shrink the FC layers.

2. Related works

The experiments in this paper are conducted on two datasets, the MNIST [3] by LeCun et al, and the CIFAR-10 with a conv net model [4] by Alex Krizhevsky.

The training and testing are done entirely with caffe [5].

There are a lot more questions to ask about the basic components in convolutional neural networks. One way is to examine what kind of invariance the neural network has learnt by fooling the neural network with synthetic images as input [6]. The fact that CNNs can be fooled by adding noise unnoticeable to our biological brain means that there are more differences between the way CNNs classify images and how biological neural network does than what we currently understand. Therefore, as we develop better and better models of neural network there is still always room for us to learn more about the very basic neural network components.

3.1 Technical Approach

The most straight forward way to reduce the number of parameters of a FC layer is to reduce the size of its input, which can be either the feature map from a convolution layer or the output from another FC layer. Empirically, the most parameters in FC layers are generated between the FC layer and a convolution layer.

One way to reduce the size of the feature map of the last convolution layer is to add extra convolutional layer after it, and use certain configuration of kernels, paddings, and strides to mindfully reduce the size of the output size.

3.2 Experiments Setup

Experiment set I

The experiment set one consists of two very similar neural networks trained on MNIST: the first neural network contains 2 convolutional layers each followed by a pooling layer, 2 FC layers with ReLU and a softmax-with-loss layer; the second neural network contains 3 convolutional layers each followed by a pooling layer, and a reshape layer that transform the output of the last convolution layer into a suitable format for the softmax-with-loss layer.

The second network has an extra convolution layer before the FC layers and retains the other configurations of the first network (in total, 3 conv layers and 2 FC layers).

Experiment set II

Since MNIST is a small dataset, the state-of-art networks on MNIST don't have very many parameters to begin with. In order to make the experiments more generalizable, I designed a second experiment set on CIFAR-10 which has substantially more data, more parameters in the networks, and the data type are quite different from those of MNIST (object images vs black-white handwritings).

Set two contains two neural networks trained on CIFAR-10: the first neural network contains 3 convolution layers, 1 FC layer and a softmax-with-loss layer; the second neural network contains 4 convolution layers (an extra convolution layer before the FC layers), 1 FC layer, and a softmax-with-loss layer.

Both experiments sets are trained until fully converge and are tested on validation sets.

4.1 Experiment results

All networks are trained until no significant improvements can be achieved. For each network, the number of parameters are calculated and shown in the tables to indicate the over-fitting problem caused by FC layers.

MNIST	Accuracy	Error rate	loss	# para
2conv 2fc	99.09	0.91	0.0292	431,080
3conv 2fc	98.39	1.61	0.0116	30,080

Table 1: Data from the experiment set 1.

CIFAR-10	Accuracy	Error rate	loss	# para
3conv 1fc	75.35	24.65	0.7535	4,198,244
4conv 1fc	73.83	26.7	0.8509	266,084

Table 2: Data from the experiment set 2.

4.2 Experiment Analysis

In the two experiments, the sizes of the models are reduced by 93.04% and 93.66% respectively, whereas the accuracy only decrease by 0.7 and 2.05 respectively.

Notice that in both experiments, I reduced parameters only from the FC layers, despite the fact there is also an increase of parameters generated by the extra convolution layer, which is largely insignificant compared to the number of parameters reduced from FC layers. Such results provide good confidence to support my argument that these models have some degree of over-fitting problems due to the over-sized FC layers.

The slight decrease in performance could be a sign of the reduction of the previous over-fitting on the training data. If that's true, then the decrease in performance could be partly from the right classifications enabled by false details that don't reflect reality, thus such reduction in performance could theoretically induce more generalizability for the model on data outside of the training set.

For the networks on MNIST, most of the error rate is due to some very terrible handwritings in the dataset that I can hardly recognize myself. Having more parameters might allow the network to over-fit on certain details to enable it to recognize those hardly recognizable digits but such improvement is of little value for real world problems, since people with terrible handwritings don't necessarily write the digits in similar ways and the network will have to learn every terrible handwriting to get actual increase over 99%.

For the networks on CIFAR-10, the task to classify objects is apparently harder than classifying digits in smaller black-white images, thus the network only achieved around 75% accuracy. In the second network, the decrease in performance due to reduction of parameters is more significant than experiment set I, which might indicate that the optimal number of parameters might be actually higher than the one I reduced to. Nonetheless, the reduction of 93.6% parameters only results in a 2 percent decrease in accuracy still strongly illustrates that the FC layers of the first CIFAR-10 network have over-fitting problems.

5. Conclusions

As the experiments proved that the over-fitting problem is pretty common even in some state-of-the-art networks. The methodology of my experiments directly pinned the over-fitting problems to the oversized FC Layers.

Empirically, sometimes more parameters might give a marginal increase in the performance, but at the same time we should be alert that the increase in performance might be from falsely included details that are only justifiable in the training data. Especially when we can reduce the parameters of the models by over a half without any significant decrease in the performance.

There is a trade-off between size and accuracy, but we should also be alert about the potential trade-off between generalizability and performance on a specific dataset.

Anyway, this experiment can also be a reminder to researchers to be skeptical about the origin of the improvements in performance.

References

[1] Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet classification with deep convolutional neural net- works. In *NIPS*, pp. 1106–1114, 2012.

[2] "Convolutional neural network." *Wikipedia: The Free Encyclopedia*. Wikimedia Foundation, Inc. 10 Nov 2015. Web. 21 Nov. 2015. URL: https://en.wikipedia.org/wiki/Convolutional_neural_network

[3] Y. LeCun. Gradient-Based Learning Applied to Document Recognition, Proceedings of the IEEE, 86(11):2278-2324, November 1998

[4] A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images, 2009. Web.21 Nov.2015. URL: http://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

[5] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell. Caffe: Convolutional Architecture for Fast Feature Embedding. Web. 21Nov. 2015 URL: http://arxiv.org/abs/1408.5093

[6] A. Nguyen, J. Yosinski, J. Clune. Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images. CVPR 2015.

[7] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.