Current Topics in Artificial Intelligence: Depth

Chenxi Liu
Department of Statistics
University of California, Los Angeles
cxliu@ucla.edu

Abstract

This short survey discusses the role of depth in current deep learning research. Both positive and negative views towards this topic are provided and contrasted. On the positive side, the general philosophy behind depth is first presented (Section 1), followed by support from both theory (Section 2), where we highlight the efficiency and representation power of deep models, and experiments (Section 3), where we use participants of ILSVRC to illustrate the point. On the negative side, we start with the well known issue of vanishing/exploding gradient (Section 4), and finally show how and why it is possible for a deep model to be approximated by a shallow one (Section 5). This is the first of the four short surveys.

1 Pro: Hierarchical representation

There is no doubt that the world around us is hierarchical: a face consists of two eyes, two ears, a mouth, and a nose; a human consists of a face, a torso, two arms, and two legs... This holds true for language as well, which is why we have WordNet [1].

Representation learning aims at learning representations of the data that capture useful information for classifiers or predictors [2]. The artificial neural network is a natural choice towards this goal, because deep neural networks have at least one hidden layer by definition, therefore each layer captures a certain hierarchical representation between the input and output.

The stack of Restricted Boltzmann Machines is an excellent example to illustrate this hierarchical representation of data. Learning a stack of RBMs is quite essential for both deep belief nets [3] and autoencoders [4]. One RBM is simply a bipartite undirected graphical model, so a stack of RBMs is learned in a layer-by-layer fashion. Concretely, we clamp the data vector to the visible units, run contrastive divergence [5] until it converges, and then treat the hidden units as the new visible units and learn another RBM on top of that. Therefore, each layer serves as an extra level of abstraction of the raw data.

This holds true not only for RBMs, but also for convolutional neural networks. [6] showed qualitatively how neurons in different layers respond to different types of image patches. [7] took this one step further and offered quantitative analysis of both object CNNs and scene CNNs, proving that higher layer neurons correspond to higher level (e.g. object, scene) instead of lower level (e.g. texture, object part) knowledge. Recently [8] proposed to look at the activities of groups of neurons instead of a single neuron (which corresponds to the population encoding theory in cognitive science), and showed that population encoding gives better visual concepts at a particular level.

All these findings serve to show that by having a certain amount of depth, we are able to build the hierarchical representation of the input data, which is a reasonable model considering the world we live in. However, how powerful is depth from a mathematical point of view, and do we have empirical evidence to support the claim?

2 Pro: Theoretical support

While it was proven that a neural network with at least one hidden layer can represent any function to an arbitrary degree of accuracy as long as its hidden layer is permitted to have enough neurons [9], in practice we still need to take efficiency into account. That is, shallow networks can be very inefficient in terms of the number of neurons and the number of examples, and deep networks provide a more compact solution.

A naive way to see the power of depth is that suppose we have in total $m + n$ neurons. If we use all of them in one hidden layer, then between an input neuron and an output neuron we have $2(m + n)$ connections. However, if we distribute these neurons into two layers, then the number of connections between input and output neuron becomes $m + mn + n$, and of course $mn > m + n$ for $m \geq 2, n \geq 2$.

More formal work on the efficiency of deep representation include [10] [11] [12] [13]. One such example is that the parity function with d inputs requires $O(2^d)$ examples and parameters to be represented by a Gaussian SVM, $O(d^2)$ parameters for a one-hidden-layer neural network, $O(d)$ parameters and units for a multi-layer network with $O(\log_2 d)$ layers, and $O(1)$ parameters with a recurrent neural network. Another example is that boolean functions expressible by $O(\log d)$ layers of combinatorial logic with $O(d)$ neurons in each layer may require $O(2^d)$ neurons when using only 2 layers. The general lesson here is that depth could have exponential advantage.

We argue that deep architectures have more representation power than shallow ones, but how to properly define the *representation power*? Since neural networks can be seen as a non-linear partition of input space, one potential criteria is to look at the number of regions that a network could partition into.

[14] studied the multi-layer feedforward networks with rectified linear units. The main theorem states that a model with n_0 inputs and k hidden layers of widths $n_1, n_2, \ldots, n_k$ can divide the input space in

$$(\prod_{i=1}^{k-1} \lfloor \frac{n_i}{n_0} \rfloor) \sum_{i=1}^{n_0} \binom{n_k}{i} \tag{1}$$

or possibly more regions. The follow-up work [15] improved this lower bound to

$$(\prod_{i=1}^{k-1} \lfloor \frac{n_i}{n_0} \rfloor^{n_0}) \sum_{i=1}^{n_0} \binom{n_k}{i} \tag{2}$$

This result suggests that the number of linear regions is $\Omega((\frac{n}{n_0})^{(k-1)n_0} n^{n_0})$ for hidden layer width $n \geq n_0$, which grows exponentially in k and polynomially in n, proving once again the exponential advantage of depth.

3 Pro: Experimental support

Convolutional neural networks have been proved hugely successful in the past few years (for computer vision tasks in particular). Back in the 1980s, CNNs were first introduced for the classification of hand-written digits [16]. However, the recent comeback of CNNs was marked by the 2012 ImageNet classification challenge, where AlexNet [17] won by a large margin of 10%.

Now that nearly 4 years have passed, we observe that people do not come up with specific network architecture design for each individual task very often. Instead, there exists a handful of classic architectures [17, 18, 19, 20] with exactly the same building blocks (namely convolution layers, pooling layers, ReLU layers etc). The most distinctive feature between these architectures is depth, and in fact the greater the depth, the smaller the error rate.

3.1 AlexNet: 8 layers, 16.4% error rate

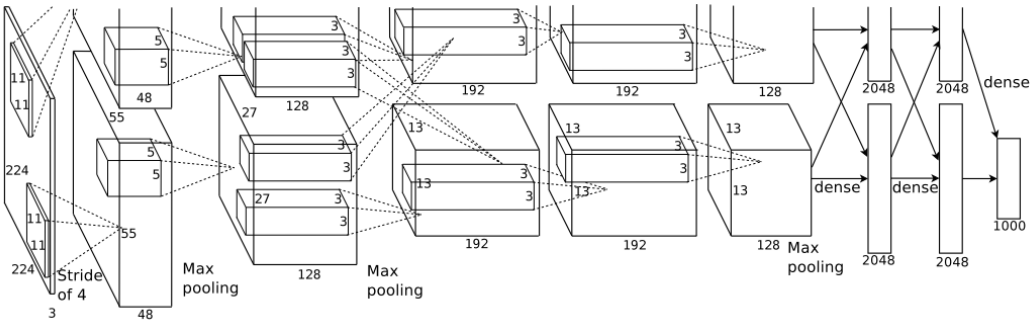

Figure 1: AlexNet Architecture

The AlexNet [17] architecture contains 8 layers. The first 5 are convolution layers followed by ReLU and max pooling, and the last 3 are fully connected layers.

3.2 VGG Net: 19 layers, 7.32% error rate

The deepest variant of VGG Net [18] has 19 weight layers. Very similar to AlexNet, the last 3 layers of VGG Net are fully connected layers, so VGG Net is essentially expanding the number of convolution layers. In addition, all convolution layers have filter size 3 by 3.

3.3 GoogLeNet: 22 layers, 6.67% error rate

While VGG Net simply seeks to expand the number of convolution layers, GoogLeNet [19] tries to come up with an interesting module called "inception". This module is essentially the concatenation of a series of convolution layers with different kernel size. The resulting architecture has 22 weight layers with 3 loss layers at different depth level (similar to the idea in [21]).

3.4 ResNet: 152 layers, 3.57% error rate

ResNet [20] is the newest winner of the ILSVRC. Instead of simply stacking convolution layers, ResNet introduces the identity mapping between layers, which in a way guarantees good initialization. The power of depth is undeniable: error rate decreases monotonically for the 34, 50, 101, 152 layer variant.

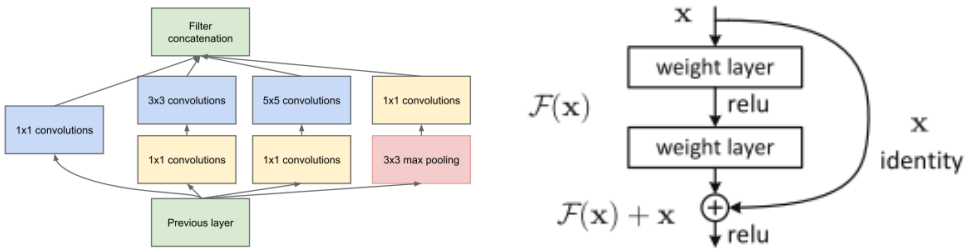

Figure 2: Left: Inception Module; Right: Residual Learning Building Block

4 Con: Deeper models are harder to train

While the models introduced in the previous section give great performance, it is also well known that it is difficult to reproduce the training from scratch. This is mainly due to the notorious vanishing/exploding gradient problem.

For example, training VGG Net [18] cannot be done in one go. Only the first 11 layers are trained together, and then other layers are fine-tuned in a layer-by-layer fashion. It is further showed in [20] that for both CIFAR-10 [22] and ImageNet [23], if we simply stack more layers then both training and test error will go up. This is obviously not overfitting, therefore the experiment suggests that as the architecture becomes too deep, more advanced architecture design or regularization methods are needed, which is their motivation behind the identity mapping between layers.

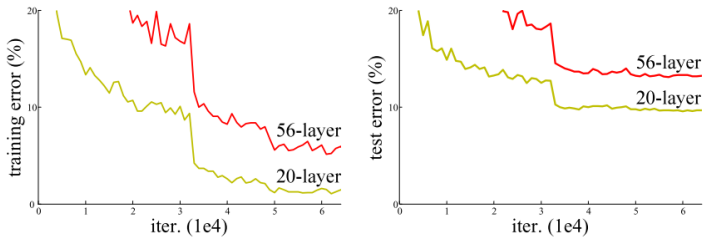

Figure 3: On CIFAR-10, both training and test error of 56 layer are higher than 20 layer

Perhaps the more commonly used example in illustraing the depth problem is Recurrent Neural Networks. The architecture of RNN is essentially one layer expanded over time, i.e. a deep neural network with shared weights across different layers. The number of time steps is the number of hidden layers, therefore the vanishing/exploding gradient problem could be quite severe.

To address this problem, people come up with architectures such as LSTM [24] and GRU [25] to enable potentially long gradient flow, and [26] do visualize that the cells are able to learn meaningful features and keep information for a long sequence. [27] [28] are recent efforts to propose LSTM-inspired models for vision tasks.

In sum, what we have seen here is that despite the growing difficulty in training deeper and deeper models, there are still players in the game of pushing the limit. If we agree that deeper models have stronger representation power, then there really is no stopping people carrying on.

5 Con: Shallow approximation of deep network

A very interesting discovery in [29] is that without hurting much performance, we can approximate a deep network with a shallow one with roughly the same number of parameters. Concretely, they did the following on both speech and vision datasets:

1. Train a deep model using standard techniques
2. Synthesize large amounts of input data and pass through the model to get input-output pair
3. Use a shallow model to approximate this mapping with L2 penalty
4. Test both deep model and shallow model on the test set to compare performance

The idea behind the paper may be more easily explained by analogy to linear regression. Imagine you have N points on a 2D plane through which you want to fit a line. If you want to recover the data exactly with a polynomial then you you will need a polynomial of degree N. However you can also fit the data with a polynomial of degree $M < N$ with L2 penalty. Obviously the reconstruction is not perfect now, but may well be a pretty good model.

We know that for classification tasks, a deep neural network serves as a non-linear partition of the input space. By synthesizing large amounts of input data and passing through the deep model, we are essentially recovering this partition as accurately as possible. The goal for the shallow model (which presumably has less representation power) is to mimic this non-linear partition with L2 loss. The experiments in this paper show that indeed the approximation is quite satisfactory, suggesting that it may be unnecessary to use a deep model for common tasks.

In fact, the presumably low representation power of the shallow network may serve as a form of regularization, and help boost the performance. This is because the original data may contain wrong labels. While the deep model may fit to this noise, the shallow model may be able to reject it. This is quite similar to the linear regression analogy.

I personally feel that a more likely explanation is the natural data in fact lie in very low dimensional space. It is well known that a face can be modeled quite well using a few principal components from PCA [30], and this notion is also supported recently in neural networks [31]. While there are fewer proofs for natural data in general, it is possible for deep neural networks to have too much representation power than we really need.

Despite this explanation, it is still quite surprising to see that deep neural networks are not showing 100% of their power. In designing a neural network architecture for a certain task, a common recipe is to keep adding layers until the performance stops growing. By that time we consider the network structure to be suitable for the task at hand and start tuning hyper-parameters more carefully. However, this paper suggests that maybe carefully selecting the number of layers is not necessary, and tuning the number of neurons in the only hidden layer should be sufficient.

A natural question to ask is: then why don't we train a shallow model directly? It turns out that training a shallow model from scratch is much more difficult than training a deep model from scratch. The reason behind this phenomenon remains an open question. In fact this is my major critique of this work: the creation of the shallow model depends on both the availability of large amounts of unlabeled data and a pre-trained deep model, and neither of the requirements is cheap.

6 Conclusion

The advantage of deeper models over shallower ones, while significant, is not absolute. This can be seen from two pairs of arguments presented in this article:

- In theory, the representation power of deeper models is stronger. However, modeling the world we live in might not need a model as complicated as the one we have now.
- Deeper and deeper models do give higher and higher performance. But the difficulty also increases, in terms of both the vanishing/exploding gradient problem and hardware constraint. As a result, the training of deeper models is harder to reproduce, because it involves more "tricks".

Since nowadays the deep learning community is still quite performance driven, people are likely to keep pushing the limit of depth. However, what I would like to see is that in addition to estimating the capacity of the *model*, we should come up with ways to estimate the capacity of the *problem* itself, and choose the model design (e.g. number of hidden layers, number of neurons in each layer) accordingly. This will eliminate the painful process of experimenting with the architecture design, and make deep learning more friendly and universal.

References

[1] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[2] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 8, pp. 1798–1828, 2013.

[3] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[4] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[5] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[6] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer vision–ECCV 2014*, pp. 818–833, Springer, 2014.

[7] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object detectors emerge in deep scene cnns," *arXiv preprint arXiv:1412.6856*, 2014.

[8] J. Wang, Z. Zhang, V. Premachandran, and A. Yuille, "Discovering internal representations from object-cnns using population encoding," *arXiv preprint arXiv:1511.06855*, 2015.

[9] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[10] J. Hastad, "Almost optimal lower bounds for small depth circuits," in *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pp. 6–20, ACM, 1986.

[11] J. Håstad and M. Goldmann, "On the power of small-depth threshold circuits," *Computational Complexity*, vol. 1, no. 2, pp. 113–129, 1991.

[12] Y. Bengio, O. Delalleau, and N. L. Roux, "The curse of highly variable functions for local kernel machines," in *Advances in neural information processing systems*, pp. 107–114, 2005.

[13] Y. Bengio, Y. LeCun, *et al.*, "Scaling learning algorithms towards ai," *Large-scale kernel machines*, vol. 34, no. 5, 2007.

[14] R. Pascanu, G. Montufar, and Y. Bengio, "On the number of response regions of deep feed forward networks with piece-wise linear activations," *arXiv preprint arXiv:1312.6098*, 2013.

[15] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio, "On the number of linear regions of deep neural networks," in *Advances in neural information processing systems*, pp. 2924–2932, 2014.

[16] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[19] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[21] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," *arXiv preprint arXiv:1409.5185*, 2014.

[22] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009.

[23] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255, IEEE, 2009.

[24] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[25] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[26] A. Karpathy, J. Johnson, and F.-F. Li, "Visualizing and understanding recurrent networks," *arXiv preprint arXiv:1506.02078*, 2015.

[27] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Advances in Neural Information Processing Systems*, pp. 2368–2376, 2015.

[28] N. Kalchbrenner, I. Danihelka, and A. Graves, "Grid long short-term memory," *arXiv preprint arXiv:1507.01526*, 2015.

[29] J. Ba and R. Caruana, "Do deep nets really need to be deep?," in *Advances in neural information processing systems*, pp. 2654–2662, 2014.

[30] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, no. 7, pp. 711–720, 1997.

[31] J. R. Gardner, M. J. Kusner, Y. Li, P. Upchurch, K. Q. Weinberger, and J. E. Hopcroft, "Deep manifold traversal: Changing labels with convolutional features," *arXiv preprint arXiv:1511.06421*, 2015.