Notes on Residual Networks

Artur Back de Luca

February 12, 2019

Introduction

This is the compilation of my personal notes on Residual Networks. They cover introductory aspects such as the success of deep learning models, the intrinsic challenge of learning that prevented such models to reach popularity in earlier decades and its solutions; the degradation problem and finally the structure and explanation of residual networks.

1 Success of deep learning

The success of deep learning models resides on the ability to derive complex features hierarchies from low-level inputs. This is achieved by substantially coupling non-linear operations throughout layers which whose parameters, in turn, rearrange, gradually adapting to a target function. Features of an input can be progressively exploited as the number of layers in a Neural Network increase, establishing network depth as a determinant factor of success in models, especially in intricate tasks such as image recognition.

Figure 1: Compositional representation of features throughout layers: from pixels to gradients and edges. Source: Goodfellow et al. (2016, pg. 6)

Research outlines from He et al. (2015, pg. 2) and Bengio (2009, pg. 6) indicate performance improvement upon an increase in network depth assisted by techniques which will be later here commented such as max-pooling, batch normalization, and initialization methods[1]. Yet, as the reader may conjecture, the absolute number of layers in a network is not the main issue, rather the relative size to how many layers are necessary to effectively represent the target function. As Bengio (2009, pg. 9) describes:

> "More precisely, functions that can be compactly represented by a depth k architecture might require an exponential number of computational elements to be represented by a depth $k-1$ architecture. Since the number of computational elements one can afford depends on the number of training examples available to tune or select them, the consequences are not just computational but also statistical: poor generalization may be expected when using an insufficiently deep architecture for representing some functions."

In this statement, Bengio (2009) drives an analogy of deep learning architectures and logic circuits, supported by the work of Hastad (1986) and Yao (1985) which imply that logic architecture limited in-depth presents an exponential number of components. As an illustration, consider the calculation of the parity function, defined as:

$$f : \{0,1\}^n \rightarrow \{0,1\} \; s.t. \; f(x) = \left(\sum_{i=1}^{n} x_i \right) \; mod \, 2$$

[1] More unmentioned techniques are applied such as dynamic learning rates, dropout and appropriate activation functions, such as the ReLu.

The work developed by Yao (1985) suggests that the number of logical components for a depth-limited architecture of 2 layers is of exponential order of 2^{n-1}, in oposition of depth unlimited architectures which in turn can derive less complex orders, such as the balanced tree structure, of $O(N \log N)$ Bengio and LeCun (2007).

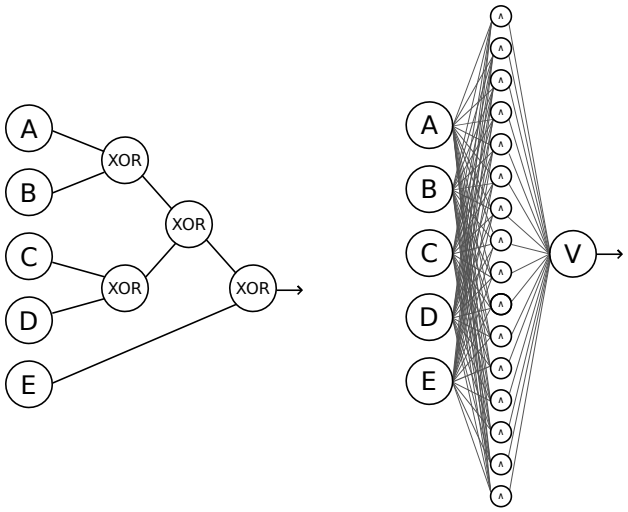

Figure 2: Distinct architectures to represent the parity function. A balanced tree structure (*left)* with 3 layers and complexity of $O(N \log N)$. A DNF (Disjunctive normal form) structure (*right*), with complexity of $O(2^N)$.

Thus, ideally, a function represented by a model may have a minimum number of layers which are necessary to properly generalize its behavior. As one may assume, complex tasks, in this case, specified as functions, would require deeper networks. And supporting evidence from recent breakthroughs in Machine Learning are associated with the recent capability of stacking an increasing amount of layers, which will be further discussed.

2 Why deep learning was not popularly employed

Despite the mentioned advantages over shallow networks, deep architectures were not extensively researched in literature as today. This was due to the difficulty of training neural networks with more than two layers, generating poor generalization usually obtained by a random initialization of parameters (Bengio et al. (2007)). In these situations, weights would lie very far from local optima, requiring many training iterations to readjust. Moreover, in deep configurations, the backpropagation technique may become troublesome as a result of exploding or vanishing gradients. As the cost function is progressively derived in terms of numerically large parameters, these adjustments will likely overshoot, hamper-

ing training. Conversely, small contributions propagated through many layers may cause virtually no effect at all:

Consider the following node output and activation:

$$z_i = W_i a_{i-1} + b_i$$

$$a_{i+1} = g\left(W_i a_{i-1} + b_i\right)$$

The derivative of the loss function $\mathcal{L}$ in a network with l layers in terms of a weight $W_n$can be written as:

$$\frac{\partial \mathcal{L}}{\partial W_n} = \frac{\partial z_n}{\partial W_n}\left[\prod_{i=n}^{l-1} \frac{\partial a_{i+1}}{\partial z_i}\frac{\partial z_{i+1}}{\partial a_{i+1}}\right]\frac{\partial a_l}{\partial z_l}\frac{\partial \mathcal{L}}{\partial a_l}$$

Defining a linear activation function $g(z_i) = z_i$ and $W_i = W$, we obtain:

$$\frac{\partial \mathcal{L}}{\partial W_n} = \frac{\partial z_n}{\partial W_n}\left[\prod_{i=n}^{l-1} \cancel{\frac{\partial a_{i+1}}{\partial z_i}}\overset{1}{\cancel{\frac{\partial z_{i+1}}{\partial a_{i+1}}}}\right]^{W}\frac{\partial a_l}{\partial z_l}\frac{\partial \mathcal{L}}{\partial a_l}$$

Thus:

$$\frac{\partial \mathcal{L}}{\partial W_n} = a_{n-1}W^{n-l}\frac{\partial \mathcal{L}}{\partial a_l}$$

If the elements in W greater than one, and with a sufficiently large $n - l$ value, $\frac{\partial \mathcal{L}}{\partial W_n}$ will tend to infinity, i.e. *exploding*. Conversely, for values less than one, the derivative tends to zero, i.e. *vanishing*.

For the case of early neural network development, a typical initialization framework presented in Erhan et al. (2009) (as cited in Bradley (2009, pg. 24)) would have parameters randomly sampled from a uniform distribution centered in zero of $[-1/\sqrt{k}; 1/\sqrt{k}]$ where k is the size of the previous layer of a fully-connected network. For deep architectures, many nodes would likely be initialized very close to zero, leading to the previously mentioned vanishing gradient problem.

This all changed in 2006, when Geoffrey Hinton, presented a solution to the training challenge utilizing the unsupervised algorithm RBM (Restricted Boltzmann machine) to precondition the weights one layer at a time Bengio (2009, pg. 4). The technique was used in the conception of a deep belief network, what in time surpassed performance ever seen using deep architectures, reaching state of the art results and unleashing groundbreaking effects led by research on new techniques but also deeper and novel architectures.

Next topics:

3 - Techniques to improve the increasing amount of layers in deep networks
4 - The degradation problem
5 - Residual Networks
Conclusion

References

Bengio, Y. (2009). Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. (2007). Greedy Layer-Wise Training of Deep Networks. page 13.

Bengio, Y. and LeCun, Y. (2007). Scaling Learning Algorithms towards AI. page 41.

Bradley, D. M. (2009). Learning In Modular Systems.

Erhan, D., Manzagol, P.-A., Bengio, Y., Bengio, S., and Vincent, P. (2009). The Difï¬¬*cultyofTrainingDeepArchitecturesandtheEffectofUnsupervisedPre−Training.page* 8.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Hastad, J. (1986). Almost Optimal Lower Bounds for Small Depth Circuits. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, STOC '86, pages 6–20, New York, NY, USA. ACM. event-place: Berkeley, California, USA.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep Residual Learning for Image Recognition. *arXiv:1512.03385*.

Yao, A. C. (1985). Separating the polynomial-time hierarchy by oracles. In *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, pages 1–10.