Facing the Depth with Skip Connections

Michaël Diatta
University of Fribourg, Switzerland,
michael.diatta@unifr.ch

9 June 2017

Abstract

Deep neural networks (DNNs) show superior performances compared to standard shallow neural networks (SNNs) in image processing tasks, such as image restoration or recognition. Despite this, the deeper are neural networks, the harder they are to train. In this paper, we show how residual learning and his theoretical history make it possible to cope depth related problems like the vanishing gradient. Then we show how residual learning has inspired the elaboration of skip connections which directly influence the performance of image restoration tasks.

1 Introduction

A neural network (NN) consists of a number of simple interconnected processing units. The purpose of these units is to process the information to achieve a desired behavior. To perform it, units called "neurons" need to endorse a series of causal computational step, which depend on the pattern that their connections express. In this paper we are concerned by a problem related to depth in DNNs and it's corollary effects for image processing tasks [14].

The depth of a neural network (NN), given a fixed topology, is define by the length of the causal computational chain [15, 6]. If the number of steps is less than 10 steps, deep learning experts tend to speak about SNNs, until 30 steps about DNNs and after that about very DNNs[15, 6, 17, 13]. But there is no clear consensus on the demarcation.

In 1991, in his diploma thesis, Hochreiter [9] discovers one of the fundamental Deep Learning problems regarding the Gradient Descent, today known as "vanishing gradient problem". He shows that layers in some DNN learn at different speeds. In the vanishing gradient problem, the later layers learn well but the earlier ones are stuck during the training. Indeed during the training, standard activation functions, like sigmoid or tanh, accumulate the back-propagated error signals in such way that the gradient tends to get rapidly smaller. This problem also lead to a degradation of the training accuracy of the model. To avoid this phenomena and its corollary effects [14] on the flow of informations, one solution is to think about how the network architecture and activation functions influence the learning of the gradient descent during the back-propagation [3, 5].

2 Take inspiration from ResNet

The main purpose behind the construction of residual networks (ResNets) can be summarized by this question: knowing that, for the same number of iteration step, a SNN with a small number of layers has a smaller training error than a SNN with an higher number of layers [8, 7, 16], how to take into account the benefits of this prperty, while maintaining a deep model? The held solution can be explained in two parts, first by a modular stacking and secondly by the residual learning.

2.1 Modular stacking

The modular stacking consists of using a series of modules, composed by two layers, stacked one after the other. By using this shallow architecture of two layers it is possible to contain the mapping function between the input and the desired output in a building block. The function which performs the operation is called an underline mapping function, denoted $H(x)$ [8]. By making an hypothesis we assume that this function can approximate other complicated functions like the mapping of a restoration image task.

2.2 Residual learning

The residual learning consists of replacing $H(x)$ by a residual mapping function. The residual mapping function is performed by doing an identity mapping $I(x) = x$ using a shortcut connection between the input and the output of the building block. There are two advantages of this technique. First, it is a viable solution to the vanishing gradient problem. Because each input passes into the next module with the residual, which contributes to decrease the disappearing of the gradient, this type of learning leads to having a low training error [8, 11]. The second advantage is that the optimization of the residual mapping is easier to do than the optimization of the underline mapping because there is no extra parameters, like hidden function or dropout, which are needed to train the ResNet [8, 10, 16].

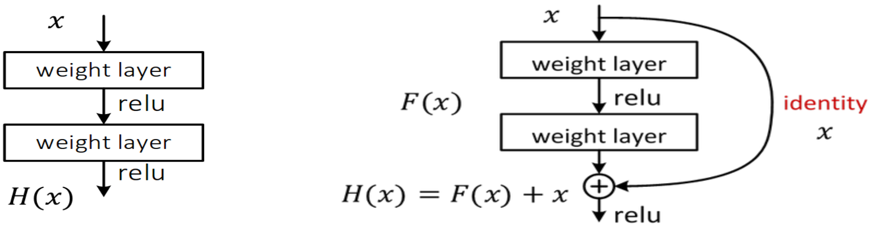
Suppose a given input x, x gets through the modular building block: a weighted convolutional layer followed by a rectified linear unit function (ReLU), followed by another weighted convolutional layer and another ReLU. The output expresses the underline function: $H(x)$, like in figure 1a. If $H(x)$ approximates a complicated function, then $H(x)$ approximates a residual function expressed by $F(x) := H(x) - x$ [8]. So in the modular residual building block, $H(x)$ is replaced by $F(x)$ after the first layer. After the second layer the residual function becomes: $F(x) + x$, as shown in figure 1b. This expresses a shortcut element-wise addition due to an identity mapping: $I(x)$. This operation allows to map the weight of the layers closer to identity, by adjusting the small fluctuations of the weights at each building block steps.[8].

2.3 Genesis of ResNets

Finally, we must specify that ResNets architecture is a lot accountable of Reccurent Neural Networks using Long Short Term Memory cells as unit (RNN-LSTMs) [10] and Highway Networks [16], regarding the method used to face the vanishing gradient problem with success [4, 17]. A question remains: are those two models really distinct? As Jürgen Schmidhuber says [1], ResNets and therefore residual learning techniques

[1] See: Feed-forward LSTM without gates

seem just to be ”a special case” of Highway Networks [16] which are themselves a feed-forward RNN-LSTM [1] versions. Indeed, the main difference consists of the layers structure. A ResNet has convolutional layers without gates. A Highway Network has recurrent layers where each component is a LSTM cell unit with three gates. But each building block of a ResNet computes the same non-linear functions with fixed parameters. For a layer of an Highway Network the mapping function can be expressed in this way: $g(x) * x + t(x) * F(x)$ [16] and the residual function of a ResNet as: $x + F(x)$ where $g(x) = 1$ and $t(x) = 1$. The main difference consist of initializing the two functions $g(x)$ and $t(x)$ with a fixed value: 1.



(a) Underline function

(b) Residual function

Figure 1: Modular building block for residual learning found in [8][2].

3 Skip connections in RED-Net

As we said, DNNs show superior performances compared to standard SNNs, in many domains such as image processing tasks, which include features extraction, noise removal, super-resolution or image restoration tasks [2, 12, 11, 6].

3.1 General RED-Net architecture

The authors of the paper, [14] which serves us as a theoretical basis for our reflection, propose to recover a clean image as an output, given a corrupted image as an input. The purpose is to use one single new model to handle denoising or super-resolution image problems. They name their model *Very Deep Residual Encoder-Decoder network* (RED-Net) [14]. The architecture consists of symmetrical convolutional layers used as an encoder and deconvolutional layers used as a decoder as shown in Figure 2. The specificity of the model is the skip-layers connections between the convolutional and the deconvolutional layers. This technique allows to handle gradient vanishing problem due to the very deep network and gives a better performance to catch images details. The model is widely inspired by ResNets architecture [14, 8].

3.2 Structure of skip connections

Skip connections are structered is such way that each two layers in the encoder part, which consist of a convolutional layer followed by a ReLU function layer, there is a shortcut connection. The shortcut connection is an identity mapping function. The shortcut connection begin in the encoder part and end in a latent block layer in the decoder part as shown in Figure 2. The decoder is composed of a chain of two layers which consist of a deconvolutional layer followed by a ReLU function layer. After this two layers there is a latent block. This latent block receives the output of the previous ReLU function which can be seen as a residual function. The residual function is added by a point-wise addition operation to the corresponding shortcut identity mapping that comes from the encoder part, achieving the residual learning. The residual learning is followed by a ReLU function layer. This operation close the latent block operations. As we can see, the skip connections work as a residual learning function in their fundamental principles.

3.3 The importance of the information flow

The authors [14] compare the performance of the skip connections of their RED-Net model to the ResNet for restoration and super-resolution image task. The results show a better performance for the RED-Net skip connections structure. In the first section of this paper we explain that the performance for a given task depends also of the architecture of the network. The purpose of restoration/super-resolution task is to learn a feature map for grasping image details in such way that the information flow is preserved. For doing this they use an auto-encoder where the architecture is symmetrical. What they do in the encoder part is done in the inverse way in the decoder part. Consequently, doing a residual learning between the two symmetrical parts make more sense than doing a residual learning in a stacked way as in [8]. Because at each encoded step we want to keep the features map which must ideally correspond to the reconstructed features map in each decoded step in an element-wise correspondence. This technique allow to preserve the information encoded and to reuse it when the image is reconstructed.

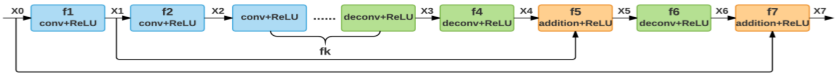


Figure 2: A representation of how two skip connections work in RED-Net, f_k express the hidden architecture, the figure comes from [14].

4 Conclusion

To face the gradient vanishing problem in deep learning and his corollary consequences, we briefly have shown how a network architecture and optimization functions can influence the performance of a DNN. By exposing the purpose of residual learning we have shown how skip connections in RED-Net solve this major deep learning problem. The authors of the paper claim that their skip connections are ”very different from the ones proposed in paper [16] and [8]”. Indeed we have shown first that ResNets networks are a special case of Highway Networks. Secondly in the last section we have shown that skip connections work as a special case of residual learning. The authors also claim that the two papers are only concerned by the optimization of the learning error when the gradient is back-propagated. But it has been demonstrated that the performance of a DNN for a given task increase when the flow of information is managed by shortcut connections, using the respective two techniques. By making a research effort in the literature, we can understand the genesis of the techniques used. This allows us to weigh the degree of inovation of a given architecture.

References

[1] Colah's blog: Understanding lstm networks. http://colah.github.io/posts/2015-08-Understanding-LSTMs/. Accessed: 2017-05-17.

[2] Pierre Baldi and Yves Chauvin. Neural networks for fingerprint recognition. *Neural Computation*, 5(3):402–418, 1993.

[3] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *Trans. Neur. Netw.*, 5(2):157–166, March 1994.

[4] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[5] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256, 2010.

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[7] Kaiming He and Jian Sun. Convolutional neural networks at constrained time cost. *CoRR*, abs/1412.1710, 2014.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[9] Sepp Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. *Master's thesis, Institut fur Informatik, Technische Universitat, Munchen*, 1991.

[10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, NIPS'12, pages 1097–1105, USA, 2012. Curran Associates Inc.

[12] B Boser Le Cun, JS Denker, D Henderson, RE Howard, W Hubbard, and LD Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems*. Citeseer, 1990.

[13] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[14] Xiao-Jiao Mao, Chunhua Shen, and Yu-Bin Yang. Image restoration using very deep fully convolutional encoder-decoder networks with symmetric skip connections. *CoRR*, abs/1603.09056, 2016.

[15] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *CoRR*, abs/1404.7828, 2014.

[16] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *CoRR*, abs/1505.00387, 2015.

[17] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. *CoRR*, abs/1507.06228, 2015.