Pattern Recognition Letters 103 (2018) 53–59

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Reducing parameter number in residual networks by sharing weights

Alexandre Boulch*

ONERA – the French Aerospace Lab, FR-91761 Palaiseau, France

ARTICLE INFO

Article history:
Received 6 March 2017
Available online 10 January 2018

MSC:
41A05
41A10
65D05
65D17

ABSTRACT

Deep residual networks have reached the state of the art in many image processing tasks such image classification. However, the cost for a gain in accuracy in terms of depth and memory is prohibitive as it requires a higher number of residual blocks, up to double the initial value. To tackle this problem, we propose in this paper a way to reduce the redundant information of the networks. We share the weights of convolutional layers between residual blocks operating at the same spatial scale. The signal flows multiple times in the same convolutional layer. The resulting architecture, called ShaResNet, contains block specific layers and shared layers. These ShaResNet are trained exactly in the same fashion as the commonly used residual networks. We show, on the one hand, that they are almost as efficient as their sequential counterparts while involving less parameters, and on the other hand that they are more efficient than a residual network with the same number of parameters. For example, a 152-layer-deep residual network can be reduced to 106 convolutional layers, i.e. a parameter gain of 39%, while loosing less than 0.2% accuracy on ImageNet.

1. Introduction

Convolutional neural networks (CNNs) are now widely used for image processing tasks from classification [24] and object detection [9,31] to semantic segmentation [1,2]. Their utilisation even generalizes to other fields where data can be represented as tensors like in point cloud processing [4] or 3D shape style identification [25]. Today's network architectures still carry a strong inheritance of the CNN early stage designs. They are based on stacking convolutional, activation and dimensionality reduction layers. Over the past years, the progress in image processing tasks went together with a gradual increase in the number of layers, from AlexNet [21] to residual networks [14] (ResNets) that may contain up to hundreds of convolutions.

Practical use of such networks may be challenging when using low memory system, such as autonomous vehicles, both for optimization and inference. Moreover, from a biological point of view, a higher number of stacked layers leads to networks further from the original underlying idea of neural networks: biological brain mimicry. According to the current knowledge of the brain, cerebral cortex is composed of a low number of layers where the neurons are highly connected. Moreover the signal is also allowed to recursively go through the same neurons. In that sense recurrent neural networks are much closer to the brain structure but more difficult to optimize [3,16].

Looking more closely at the repetition of residual blocks in ResNets, it could somehow be interpreted as an unwrapped recurrent neural networks. This constatation raises questions such as *"how similar are the weights of the blocks?"*, *"do the same parts of the blocks operate similar operations?"* and in the later case *"is it possible to reduce the parameter number of a residual network?"*. Driven by these observations and questions, we present a new network architecture based on residual networks where part of the convolutions share weights, called *ShaResNets*. It results in a great decrease of the number of network parameters, from 25% to 45% depending on the size of the original architecture. Our networks also present a better ratio performances over parameter number while downgrading the absolute performance by less than 1%.

The paper is organized as follows: Section 2 presents the related work on convolutional neural networks (CNNs); the ShaResNets are presented in Section 3 and finally, in Section 4, we expose our experimentations on classification datasets CIFAR 10 and 100 and ILSVRC ImageNet.

2. Related work

CNNs were introduced in [23] for hand written digits recognition. They became over the past years one of the most enthusiastic field of deep learning [22]. The CNNs are usually built using a common framework. They contains many convolutional layers and operate a gradual spatial dimension reduction using convolutional

* Corresponding author.
E-mail address: alexandre.boulch@onera.fr

https://doi.org/10.1016/j.patrec.2018.01.006
0167-8655/

strides or pooling layers [6,18]. This structure naturally integrates low/mid/high level features along with a dimension compression before ending with a classifier, commonly a perceptron [32], multilayered or not, i.e. one or more fully connected layers.

Looking at the evolution CNNs, the depth appears to be a key feature. On challenging image processing tasks, an increase of performance is often related to a deeper network. As an example, AlexNet [21] has 5 convolutions while VGG16 and VGG19 [34] have respectively 16 and 19 convolutional layers and more recently, in [17], the authors train a 1200 layer deep network.

Variations in the LeNet structure have also been used to improve convergence. In a Network in Network (NiN) [26], convolutions are mapped with a multilayer perceptron (1×1 convolutions), which prevent overfitting and improved accuracy on datasets such as CIFAR [20]. GoogLeNet [38] introduced a multiscale approach using the inception module, composed of parallel convolutions with different kernel sizes.

Optimizing such deep architectures can face practical problems such as overfitting or vanishing or exploding gradients. To overcome these issues, several solutions have been proposed such as enhance optimizers [37], dropout [36] applying a random reduction of the number of connection in fully connected layers or on convolutional layers [41], intelligent initialization strategies [10] or training sub-networks with stochastic depth [17].

Residual networks [14] achieved the state of the art in many recognition tasks including ImageNet [33] and COCO [27]. They proved to be easier to optimize. One of the particularities of these networks is to be very deep, up to hundreds of residual layers. More recently, the authors of [41] introduced wide residual networks which reduce the depth compared to usual resnets by using wider convolutional blocks.

To balance the increasing size of CNNs, various work consider reducing the weight number of the network to reduce memory size and or testing speed. We can distinguish three categories. The first kind of approaches consists in statically modifying the architecture to get lighter networks, e.g. the replacement of the fully connected layers by average pooling [14,38], the replacement and factorization of convolutions [39] or the weights constrained to be binary [8,30]. This work follows this approach as we modify the internal structure of the residual networks to make it lighter. The second category regroups works that dynamically modify the network at training. Among them, [7] alternate between regularization and neuron deactivation (forcing weights to zero) to progressively reduce the number of neurons. In Hashnets [5] uses a hash function to regroup connections that will share the same weights. Finally the third category post process the network to compress it. Some consider weight pruning [29,35] and sparsification [28]. Other try to compress the data via weight matrix factorization [11,19,40]. Han et al. [13] remove redundant connections and allow weight sharing, As for dynamical network modifications, these compression methods may also apply to our proposed architectures and add a compression step to our optimized ResNet architecture.

3. Sharing residual networks

ShaResNets are based on residual networks architectures in which we force the residual blocks in the same stage, i.e. between two spatial dimension reduction, to share the weights of one convolution. In this section, we first present the residual architectures we based our work on, and then, detail the sharing process.

3.1. Residual networks

The residual networks basic without [14] or with pre-activation [15] or wide [41] are a sequential stack of residual blocks, with

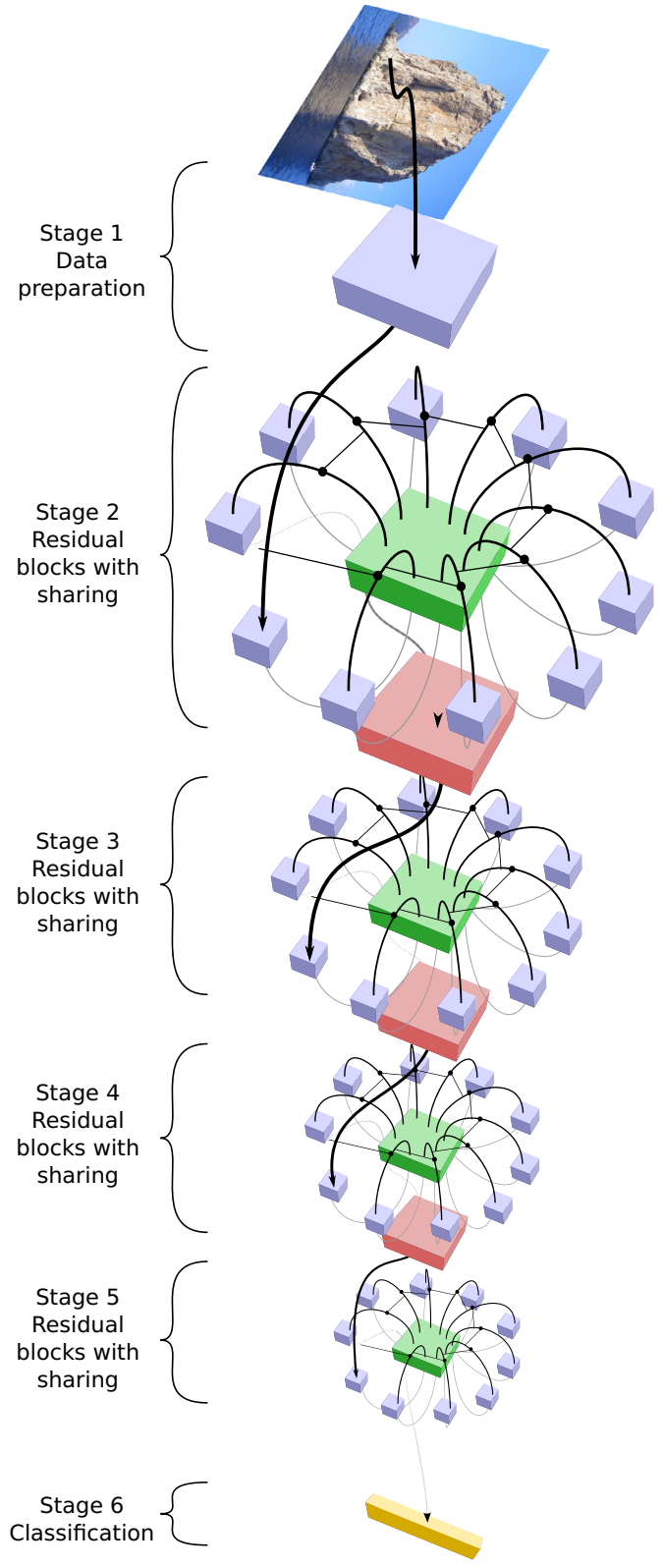

Fig. 1. 3D representation of a residual network with shared convolution. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)