Compact representations and pruning in residual networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

We show that residual networks encode their input signals in the transient dynamics of the neurons in each layer. These representations are similar for inputs from the same class, and distinct for inputs from different classes. Based on the neural transient dynamics, we provide a sufficient criterion to determine the depth of such networks during training. This criterion is based on the convergence of the neural dynamics in the last two successive layers of the residual block. This method compresses the depth of the network and removes unnecessary deep layers.

1 Introduction

Residual networks (Resnets) [1] have been more successful in classification tasks in comparison with many other standard methods. This success is attributed to the skip connections between layers that facilitate the propagation of the gradient throughout the network, and in practice allow very deep networks to undergo a successful training. Apart from mitigating the gradient problem in deep networks, the skip connections introduce a dependency between variables in different layers that can be seen as a system state. This novelty provides an opportunity for interesting theoretical analysis of their functioning, and has been the underlying pillar for some interesting analysis of such networks from a dynamical system point of view [2, 3, 4, 5, 6, 7, 8].

Some studies on Resnets have focused on tracking the features layer by layer [9, 10], and have challenged the idea that deeper layers in neural networks build up abstract features that are different than those formed in lower layers. One supporting evidence for this challenge comes from lesion studies on Resnets [11] and Highway networks [12] which show that after the network is trained, perturbing the weights in the deep layers does not have a fundamental effect on the network performance, and therefore, does not bring the performance to chance level. However, changing the weights which are closer to initial layers, have a more damaging effect. Empirical studies in [9, 10] suggest an alternative explanation for feature formation in deep layers; that is, successive layers estimate the same features which, along the depth of the network, are more refined, and yield an estimate with smaller standard deviation than earlier layers. Our approach in this paper is similar to the latter studies, however, to understand the classification mechanism in Resnets, we focus on the role of the intrinsic dynamics of the residuals over different layers of the network. Our study supports the idea in [9, 10] by showing that features in different layers of a Resnet are formed by the transient dynamics of residuals that may converge to their steady state values if they are stable. Based on this finding, we suggest an algorithm that estimates the depth of the network adaptively during training.

2 Neural dynamics in Resnets

We consider a dense Resnet with N input dimensions, and arbitrary T layers with exactly N neurons at each layer. In this network, the activity of neuron i at layer t is represented as $y_i(t)$, and the activity of all neurons in the same layer is represented by the vector $\mathbf{y}(t)$. After the integration of the output from layer $t-1$, the output of layer t is represented by $\mathbf{x}(t)$. The components of these residuals $\mathbf{y}(t)$ are calculated based on a linear function of $\mathbf{x}(t)$, i.e. $z_i(t) = \sum_{i=1}^{N} w_{ij}(t)x_j(t)$ followed by a nonlinear function $f(z_i)$. Any hidden layer t represents a sample of the dynamical states $\mathbf{x}$ after t steps. This implies that the network at different layers calculates samples of $\mathbf{x}(t)$. Input data is considered as the initial condition of the system, and is depicted by $\mathbf{x}(0)$. Interpreting the network as a dynamical system which evolves throughout the layers, the dynamics of neural activations are $\mathbf{x}(t+1) = \mathbf{x}(t) + \mathbf{y}(t+1)$, where $\mathbf{y}(t)$ is the output of the neurons, and in the rest of the paper, they are called "residuals". This equation implies a difference equation for the variable $\mathbf{x}(t)$, that is $\mathbf{x}(t+1) - \mathbf{x}(t) = \mathbf{y}(t+1)$. The left side of this equation resembles the forward Euler method of derivative of a continuous system, when the discretization step is equal to 1. This approximates a continuous system with dynamics that follow $\dot{x}_i(t) = y_i(t)$. The latter equation implies $x_i(t) = \int_0^t y_i(\tau)d\tau + x_i(0)$ where $x_i(0)$ stands for the input data that neuron i receives. In other words, $\mathbf{x}(t)$ sums up the input data as well as the activities of the neurons (residuals) over the layers. This signal feeds the next block in the network, or the classifier in the output layer.

To study the properties of the neural activities in each layer that shape the cumulative signal $\mathbf{x}(t)$, we considered a 784 dimensional network with the MNIST dataset as inputs. After training a $15-$layer Resnet using the back-propagation algorithm, we studied the dynamics of the residuals (the signal of neural activities from the input layer up to the last layer). We observed rapid changes in the first initial layers of the network, and more steady behavior close to the final layers (figure 1A, B, see also [13] for more details on the layer-dependent dynamics of the residuals and their fixed points). The dynamics of the residuals do not change significantly after the 6th layer (figure 1A), and the standard deviation of the trajectories for different samples approaches zero thereafter (figure 1B). This implies that different trajectories for each sample converge to the same fixed point.

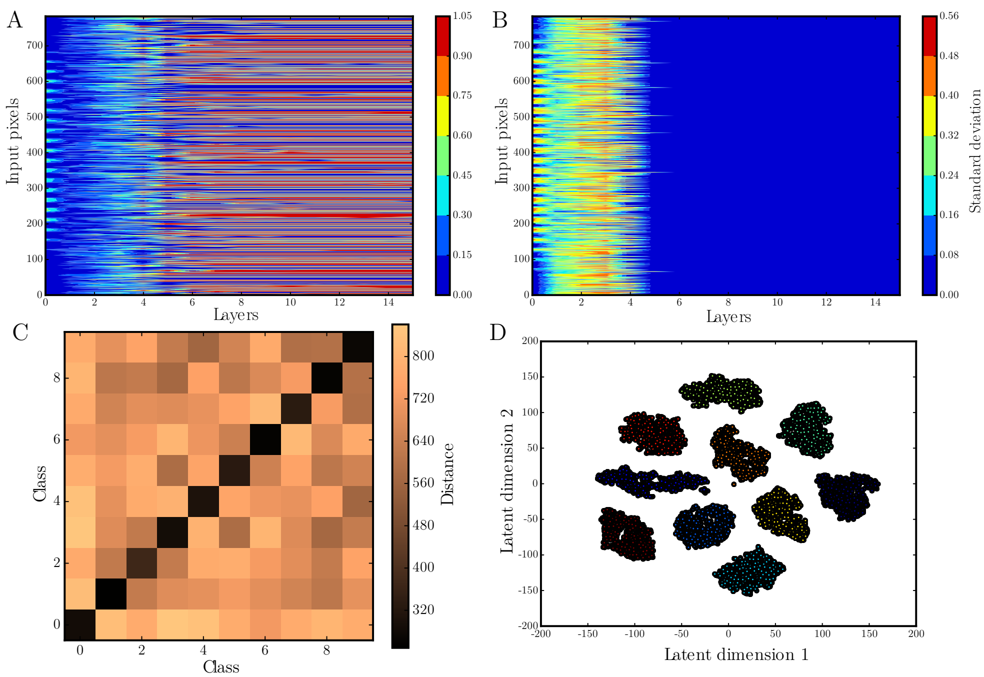

Figure 1: A: Mean of the residuals for 200 samples of each class from the MNIST dataset as a function of network depth (extracted from [13]). B: Standard deviation of the residuals (extracted from [13]). C: Distance matrix for 200 samples for each class, normalized by the number of samples. D: Dimensionality reduction of the cumulative signals for each class, at the final layer of the network, before the classifier.

In order to show the similarities and differences of the internal dynamics corresponding to each class, we compared the l_2 distance between neural trajectories of all neurons for 200 examples from each class. As illustrated in figure 1C, trajectories within each class have a smaller l_2 distance compared with trajectories for samples from different classes. This shows that the network specifies different forms of neural trajectories for inputs from each class, that are distinct from each other. Moreover, in order to generalize the classification performance for inputs from the same class, this difference for those trajectories is less, meaning that inputs from the same class are more similar. To compare the cumulative signal $\mathbf{x}(t)$ among different classes, at the final hidden layer of the Resnet before the classifier, we employed the t-SNE algorithm [14] to visualize the properties of these high dimensional signals in a two-dimensional plot. As depicted in figure 1D, signals corresponding to different classes are mapped and clustered in different regions of the two-dimensional space. This reflects how Resnets encode inputs in different and distinct forms of their neural transient (layer-dependent) dynamics (even though residuals for different classes converge to identical values in this particular example).

3 Compressing network's depth

In the MNIST network example, we observed that the neural trajectories (residuals) show less variabilities at deeper layers, some of which have already converged to their steady state values (figure 1A, B). Under this condition, the cumulative signal $\mathbf{x}(t)$ receives similar components in successive deep layers, and hence no new information about the variabilities in neural trajectories are encoded in this signal. In other words, almost always constant numbers (due to convergence) are added to this signal without providing new information about the input-induced neural transitions. Therefore, after achieving this state of neural transient dynamics, it seems viable to stop the forward propagation of information, and cut the extra layers of the network without losing much information in the cumulative signal. This provides the basis for an algorithm that adaptively sets the depth of the network considering the difference between neural activities of the last two successive layers of the network. The algorithm as such is the following:

```
while loss function is not minimum do
    for each epoch of the training data do
        residuals of the last hidden layer in the block → r1
        residuals of the second last hidden layer in the block → r2
        if l1 norm for r1 − r2 < threshold then
            remove the layer corresponding to r1
        end
    end
end
```

which compares the l_1 norm of the difference between the neural activities of the last two layers [13]. If this difference is less than a threshold, the activities of the neurons could be considered approximately identical. If this condition is fulfilled, the last layer of the network is removed. To compress the network as much as possible, this algorithm was applied on the MNIST network with shared weights between layers [6], and resulted in a 5-layer network (for threshold = 0.01) without any significant changes in the classification accuracy. Note that the convergence criterion for the residuals is a sufficient condition to remove the last hidden layer, not a necessary condition. This means that even shallower networks might still be able to classify the inputs with the same accuracy, however, our algorithm does not provide the necessary conditions to achieve the same level of accuracy for those cases. This algorithm is more efficient than fully training shallower networks first, and then adding extra layers to compensate for a high-value loss function and retraining the network.

4 Conclusion

In this letter, we showed the importance of neural transient dynamics on input classification in Resnets. It was demonstrated here that the cumulative signal of the residuals has a distinct state for each input class, in a high dimensional state space of the neural network. Also, the neural trajectories across layers are similar for inputs of the same class while different for inputs from two distinguished classes. This form of input representation in the transient dynamics of neural trajectories in Resnets can potentially underlie more efficient methods of training in the future. Based on the convergence of the neural dynamics to a steady state, we proposed an algorithm that determines the sufficient number of layers for the network during training. This can be considered as network depth compression without losing the classification accuracy significantly.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[2] Marco Ciccone, Marco Gallieri, Jonathan Masci, Christian Osendorfer, and Faustino Gomez. NAIS-NET: Stable Deep Networks from Non-Autonomous Differential Equations. *arXiv*, 2018.

[3] Bo Chang, Lili Meng, Eldad Haber, Frederick Tung, and David Begert. Multi-level Residual Networks From Dynamical Systems View. *ICLR*, pages 1–14, 2018.

[4] Eldad Haber and Lars Ruthotto. Stable Architectures for Deep Neural Networks. *arXiv*, 2017.

[5] Yiping Lu, Aoxiao Zhong, Quanzheng Li, Massachusetts General Hospital, and Bin Dong. Beyond Finite Layer Neural Networks: Bridging Deep Architectures and Numerical Differential Equations. *arxiv*, pages 1–15, 2017.

[6] Qianli Liao and Tomaso Poggio. Bridging the Gaps Between Residual Learning, Recurrent Neural Networks and Visual Cortex. *arXiv*, (047):1–16, 2016.

[7] Lars Ruthotto and Eldad Haber. Deep Neural Networks motivated by Partial Differential Equations. *arXiv*, pages 1–7, 2018.

[8] Pratik Chaudhari, Adam Oberman, Stanley Osher, Stefano Soatto, and Guillaume Carlier. Deep Relaxation: partial differential equations for optimizing deep neural networks. *Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, PMLR*, 2017.

[9] Klaus Greff, Rupesh K. Srivastava, and Jürgen Schmidhuber. Highway and Residual Networks learn Unrolled Iterative Estimation. *ICLR*, (2015):1–14, 2017.

[10] Brian Chu, Daylen Yang, and Ravi Tadinada. Visualizing Residual Networks. *arxiv*, 2017.

[11] Andreas Veit, Michael Wilber, Serge Belongie, and Cornell Tech. Residual Networks Behave Like Ensembles of Relatively Shallow Networks. *NIPS*, pages 1–9, 2016.

[12] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training Very Deep Networks. In *NIPS*, pages 1–9, 2015.

[13] Anonymous. Residual Networks classify inputs based on their neural transient dynamics. *ICLR 2018 Submission*, (2018):1–11, 2018.

[14] Laurens Van Der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.