Log-Dense Networks

Xiwei Cheng
1155141642@link.cuhk.edu.hk

Yunrui Guan
1155141425@link.cuhk.edu.hk

Abstract

Recent work has shown the positive impact of shorter connections between far apart layers on the accuracy of convolutional neural networks. The dense convolutional network (DenseNet) connects each layer to every other layer in a feed-forward fashion. Despite DenseNet's good performance, it has been criticised for its memory occupation and complexity – an L-layer DenseNet contains $\mathcal{O}(L^2)$ layer connections. In this project, we introduce the log-dense networks (LogDenseNet), in which the connection from the $(\ell - j)$-th layer to the ℓ-th layer is governed by a Bernoulli$\left(\log\left(\frac{j+1}{j}\right)\right)$ random variable. LogDenseNet significantly lightens the density of DenseNet – an L-layer LogDenseNet contains $\mathcal{O}(L \log L)$ layer connections in expectation. In the meanwhile, it inherits the advantages of DenseNet – it has strong gradient flow and allows deep supervision, thus alleviates the vanishing-gradient problem as well as avoids degradation or overfitting. The performance of LogDenseNet is similar to DenseNet of the same number of layers, while has significantly less complexity.

1 Introduction

Over the years, convolutional neural networks (CNNs) have became the dominant approach in machine learning for object detection and image classification (1; 2). The improvements of computing devices have enabled the training of increasingly deep CNN, which is able to decompose higher-dimensional features (3; 4).

However, naively deepening CNNs may cause the problem of vanishing gradient: as information about the input or gradient passes through many layers, it may vanish when reaching the end of the network (2; 5). To solve this problem, residual learning framework (ResNet) (6) applies a scheme that sums up the output and input of the previous layer as the input for the next layer. Such operation allows both the feature and its gradient to travel through the network via bypassing paths. DenseNet (2) makes further use of the intuition behind ResNets. It concatenates all previous features to be the input of next layer. Such scheme builds connections between all layers, thus the features of one layer can be adopted by every later layers, and the parameters can be optimized via the gradient of every later layers.

In this project, we propose log-densely connected convolutional networks (LogDenseNets) based on DenseNets. As illustrated in Figure 1 (2), different from connecting every two layers in DenseNets, in LogDenseNets the connection from the $(\ell - j)^{th}$ layer to the ℓ^{th} layer is governed by a Bernoulli$\left(\log\left(\frac{j+1}{j}\right)\right)$ random variable.

One major advantage of LogDenseNet is that, comparing with DenseNet it greatly reduces the number of parameters – from $\mathcal{O}(L^2)$ in DenseNet to $\mathcal{O}(L \log L)$ – as well as maintains strong information and gradient flow. The intuition behind the advantage is that LogDenseNet cuts off a significant number of "redundant and inefficient" connections in DenseNet and keeps "reasonable" amount of far apart connections. In a LogDenseNet with depth less than 1000, with a high probability the length of the shortest path from any two layers is at most 5, which means the information and gradient flow between any two layers is transmitted within 5 intermediate steps.

We evaluate LogDenseNet on the benchmark data set CIFAR-10. Our model presents comparable accuracy to DenseNet of the same depth while with much less layer connections.

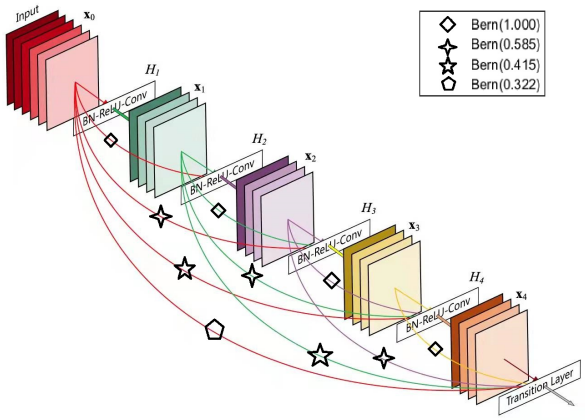


Figure 1: In LogDenseNet, each layer takes one preceding feature-map with certain probability.

2 Log-Dense Networks

DenseNets. Here we briefly introduce DenseNets. Consider a CNN that consists of L layers, with input data image x_0. Denote the transformation of i^{th} layer by H_i. Furthermore, let x_i denote the output of i^{th} layer. DenseNets applied a scheme that directly connect each layer to all layers that follows. That is, the output at i^{th} layer is $x_i = H_i([x_0, x_1, ..., x_{i-1}])$.

An advantage of DenseNets is that the gradient can flow directly from each layer to all its previous layers, thus forming a strong gradient flow. However, the memory occupation and complexity of DenseNets is extremely high.

Log-Dense Connectivity. Motivated by DenseNets, we propose Log-Dense Networks model. The connection from the $(\ell - j)$-th layer to the ℓ-th layer is governed by a Bernoulli$\left(\log\left(\frac{j+1}{j}\right)\right)$ random variable. The following algorithm describes the Log-Dense connectivity.

Algorithm 1 Algorithm to generate the input of ℓ-th layer

Initialize $input \leftarrow$ an empty list
for $j \leftarrow 1$ **to** ℓ **do**
  $rand \leftarrow$ a random number in $[0, 1)$
  if $rand < \log_2\left(\frac{j+1}{j}\right)$ **then**
    $input \leftarrow \text{cat}(\mathbf{x}_{\ell-j}, input)$
  end if
end for

Implementation Details. As illustrated in Figure 2, we used the same implementation as in the DenseNets (2) except the connectivity. Firstly, we perform a convolution with 16 output channels. Each layer contains a composite function of batch normalization (BN), a rectified linear unit (ReLU) and a 3×3 convolution (Conv) using zero padding, whose number of output channels is denoted by growth rate k. The layers are divided into three dense blocks of same size. The feature map between two adjacent blocks goes through 1×1 convolution followed by 2×2 average pooling. After the last block, we use a global average pooling, followed by a softmax classifier.

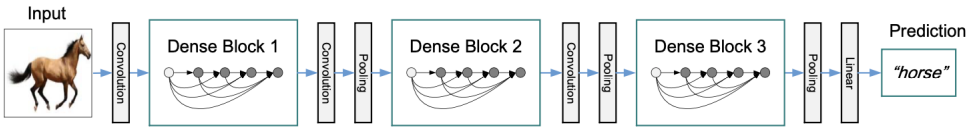


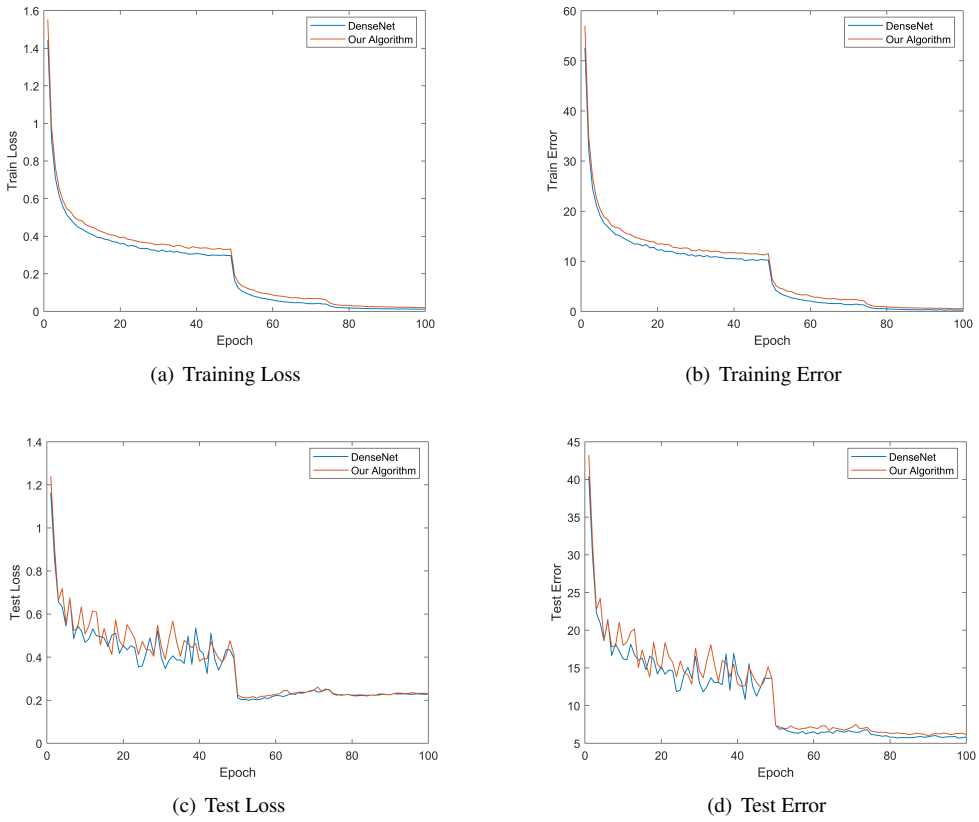
Figure 2: A LogDenseNet with three dense blocks.

3 Experiment

To evaluate the effectiveness of LogDenseNet, we perform experiment on the benchmark data set CIFAR-10 and compare with DenseNet.

We train both LogDenseNet and DenseNet with depth $L = 100$ and growth rate $k = 12$ using batch size 40 for 100 epoches. The initial learning rate is set to 0.1 and decays to 0.01 and 0.001 at the 50-th and 75-th epoch respectively. It is worth noting that the number of parameters in LogDenseNet is only 8.1×10^4 while it is 8.0×10^5 in DenseNet.

As can be seen from the results demonstrated in Figure 3, LogDenseNet performs similar to DenseNet in terms of all of training loss, training error, test loss and test error. The final test errors are both below 6.0%. The results imply that LogDenseNet achieves competitive performance to DenseNet while requiring much less parameters.



(a) Training Loss

(b) Training Error

(c) Test Loss

(d) Test Error

Figure 3: Comparison between DenseNets and Log-Dense Networks.

4 Discussion

Probabilistic Connection Model. The intuition of the proposed probabilistic connection model comes from one heat map in DenseNet (2), which is demonstrated in Figure 4. In a DenseNet with

40 layers, each pixel of the heat map evaluates the average absolute filter weight between two layers. More precisely, the color of pixel (s, ℓ) indicates the $L1$ norm of the parameters of the connection from the s-th layer to the ℓ-th layer. A pixel closer to red represents stronger dependency between two layers. It can be seen that in general the dependency between closer layers tends to be strong, while the dependency between far-apart layers tends to be weak.

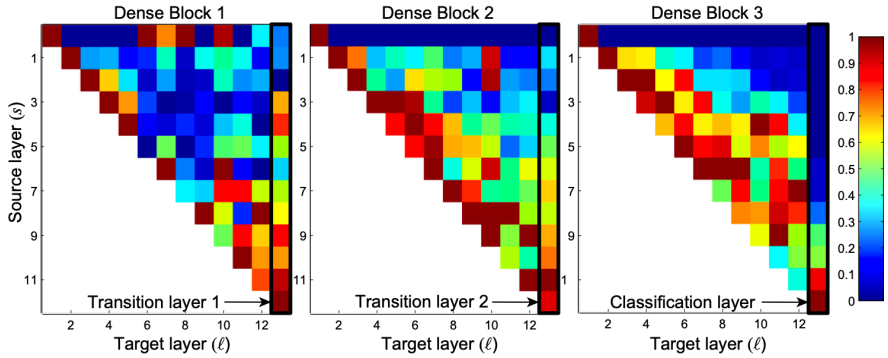


Figure 4: Each pixel of the heat map evaluates the average absolute filter weight between two layers.

One direct approach to reduce the parameters is to cut-off "redundant" connections while keep "effective" connections. Without the dependency being known a priori, due to the intuition gained from the heat map, a closer connection shall be more possible to be kept. In the meanwhile, we still want to keep a reasonable amount of connections in order to maintain strong information and gradient flow.

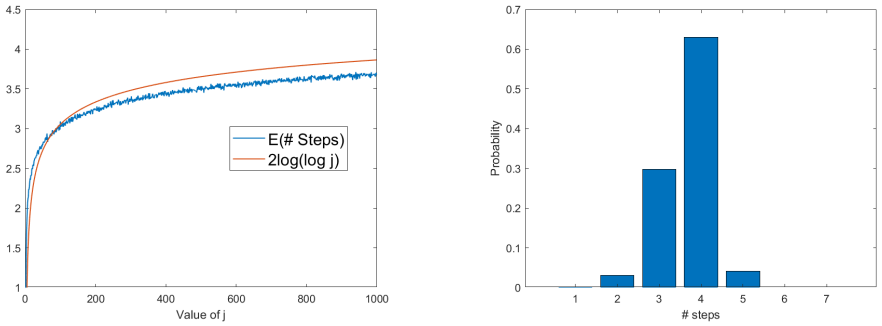
Thus, instead of connecting every two layers we govern the connection between two layers by a Bernoulli random variable. The probability that the ℓ-th layer is connected to $(\ell - j)$-th layer is $\log_2\left(\frac{j+1}{j}\right)$. Here, the connection means that the output of $(\ell - j)$-th layer is used as input of ℓ-th layer, otherwise it is not concatenated.

In such connection model, the expected number of connections to ℓ-th layer is $\sum_{j=1}^{\ell-1} \log_2\left(\frac{j+1}{j}\right) = \log_2 \ell$. Hence we can estimate that the number of connections in a L layers network is $\mathcal{O}\left(L \log_2 L\right)$. Recalling that in DenseNets the number of connections is $\frac{L(L+1)}{2}$, such model significantly reduce the connections from quadratic to almost linear.

Dense Connections. Here we discuss the density of connections in the LogDenseNet model. For $m > n$, Define the absolute distance between two layers m and n to be $m - n$ and define the number of steps between two layers m and n to be the length of shortest path between them. The number of steps represents the minimum number of intermediate layers for the m-th layer to supervise the n-th layer. We show that although a large amount of connections are reduced, the information and gradient between layers are still kept.

Figure 5 illustrates the relationship between absolute distance, denoted by j, and the number of steps. The left-hand figure plots the expected number of steps for different values of j, and the right-hand figure plots the probability distribution of the number of steps when $j = 1000$. We can see that the expectation is in general bounded by $2\log(\log j)$. And for two layers of distance 1000, with probability more than 95% the number of steps is no greater than 4, and with probability almost 100% the number of steps is no greater than 5.

It implies that in a LogDenseNet with less than 1000 layers, with high probability the information and gradient flow between any two layers is transmitted within 5 intermediate steps. Such deep supervision is able to avoid vanishing-gradient problem or degradation.



(a) Expected number of steps.

(b) Probability distribution when j = 1000.

Figure 5: Relationship between distance and steps by Monte Carlo simulation.

5 Conclusion

In conclusion, LogDenseNet admits a probabilistic-layer-connection model, which provides strong information and gradient flow. It greatly reduces the number of parameters while obtaining similar performance to DenseNet. The complexity of LogDenseNet is $O(L \log L)$, which is near linear comparing to the quadratic complexity of DenseNet.

References

[1] 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.

[2] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[3] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," *arXiv preprint arXiv:1507.06228*, 2015.

[4] 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*, 2015, pp. 1–9.

[5] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *European conference on computer vision*. Springer, 2016, pp. 646–661.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.