Deep Residual Learning for Image Recognition

Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun
Microsoft Research
{kahe, v-xiangz, v-shren, jiansun}@microsoft.com

Abstract

Deeper neural networks are more difficult to train. We present a residual learning framework to ease the training of networks that are substantially deeper than those used previously. We explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. We provide comprehensive empirical evidence showing that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. On the ImageNet dataset we evaluate residual nets with a depth of up to 152 layers—8× deeper than VGG nets [40] but still having lower complexity. An ensemble of these residual nets achieves 3.57% error on the ImageNet test *set. This result won the 1st place on the ILSVRC 2015 classification task. We also present analysis on CIFAR-10 with 100 and 1000 layers.*

The depth of representations is of central importance for many visual recognition tasks. Solely due to our extremely deep representations, we obtain a 28% relative improvement on the COCO object detection dataset. Deep residual nets are foundations of our submissions to ILSVRC & COCO 2015 competitions[1]*, where we also won the 1st places on the tasks of ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation.*

1. Introduction

Deep convolutional neural networks [22, 21] have led to a series of breakthroughs for image classification [21, 49, 39]. Deep networks naturally integrate low/mid/high-level features [49] and classifiers in an end-to-end multi-layer fashion, and the "levels" of features can be enriched by the number of stacked layers (depth). Recent evidence [40, 43] reveals that network depth is of crucial importance, and the leading results [40, 43, 12, 16] on the challenging ImageNet dataset [35] all exploit "very deep" [40] models, with a depth of sixteen [40] to thirty [16]. Many other non-trivial visual recognition tasks [7, 11, 6, 32, 27] have also greatly benefited from very deep models.

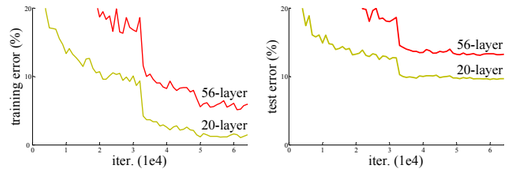


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

Driven by the significance of depth, a question arises: *Is learning better networks as easy as stacking more layers?* An obstacle to answering this question was the notorious problem of vanishing/exploding gradients [14, 1, 8], which hamper convergence from the beginning. This problem, however, has been largely addressed by normalized initialization [23, 8, 36, 12] and intermediate normalization layers [16], which enable networks with tens of layers to start converging for stochastic gradient descent (SGD) with back-propagation [22].

When deeper networks are able to start converging, a *degradation* problem has been exposed: with the network depth increasing, accuracy gets saturated (which might be unsurprising) and then degrades rapidly. Unexpectedly, such degradation is *not caused by overfitting*, and adding more layers to a suitably deep model leads to *higher training error*, as reported in [10, 41] and thoroughly verified by our experiments. Fig. 1 shows a typical example.

The degradation (of training accuracy) indicates that not all systems are similarly easy to optimize. Let us consider a shallower architecture and its deeper counterpart that adds more layers onto it. There exists a solution *by construction* to the deeper model: the added layers are *identity* mapping, and the other layers are copied from the learned shallower model. The existence of this constructed solution indicates that a deeper model should produce no higher training error than its shallower counterpart. But experiments show that our current solvers on hand are unable to find solutions that

[1] http://image-net.org/challenges/LSVRC/2015/ and http://mscoco.org/dataset/#detections-challenge2015.

This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the version available on IEEE Xplore.

Học tập dư thừa sâu để nhận dạng hình ảnh

Khai Minh Hà Tương Vũ Trương Thiếu Thanh Nhâm Kiến Tôn
Nghiên cứu của Microsoft
{kahe, v-xiangz, v-shren, jiansun}@microsoft.com

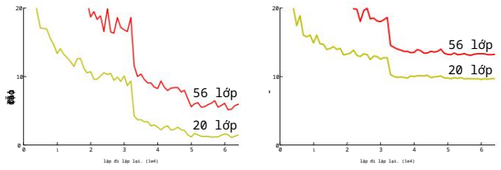
trừu tượng

Mạng lưới thần kinh sâu hơn khó đào tạo hơn. Chúng tôi trình bày một khuôn khổ học tập còn lại để giảm bớt việc đào tạo của các mạng sâu hơn đáng kể so với các mạng được sử dụng trước đó. Chúng tôi định dạng lại một cách rõ ràng các lớp dưới dạng các hàm dư đang học có tham chiếu đến các đầu vào của lớp, thay vì học các hàm không được tham chiếu. Chúng tôi cung cấp bằng chứng thực nghiệm toàn diện cho thấy rằng những dư lượng này mạng dễ tối ưu hóa hơn và có thể đạt được độ chính xác từ độ sâu tăng lên đáng kể. Trên tập dữ liệu ImageNet, chúng tôi đánh giá các lưới dư có độ sâu lên tới 152 lớp—8× sâu hơn lưới VGG [40] nhưng vẫn có độ phức tạp thấp hơn. Một tập hợp các lưới dư này đạt được sai số 3,57% trên bộ thử nghiệm ImageNet. Kết quả này đã giành vị trí số 1 trên bảng xếp hạng Nhiệm vụ phân loại ILSVRC 2015. Chúng tôi cũ ng trình bày phân tích trên CIFAR-10 với 100 và 1000 lớp.

Chiều sâu của sự thể hiện là điều quan trọng nhất cho nhiều nhiệm vụ nhận dạng hình ảnh. Chỉ nhờ vào các biểu diễn cực kỳ sâu sắc của chúng tôi, chúng tôi đã đạt được mức cải thiện tương đối 28% trên tập dữ liệu phát hiện đối tượng COCO. Sâu mạng dư là nền tảng cho việc chúng tôi đệ trình lên ILSVRC & Cuộc thi COCO 20151 , nơi chúng tôi cũ ng đã giành giải nhất đặt vào các nhiệm vụ phát hiện ImageNet, bản địa hóa ImageNet, phát hiện COCO và phân đoạn COCO.

1. Giới thiệu

Mạng lưới thần kinh tích chập sâu [22, 21] đã dẫn đến hàng loạt đột phá về phân loại hình ảnh [21, 49, 39]. Mạng sâu tích hợp một cách tự nhiên các tính năng cấp thấp/trung bình/cao [49] và các bộ phân loại theo kiểu nhiều lớp từ đầu đến cuối và "cấp độ" của các tính năng có thể được làm phong phú theo số lớp xếp chồng lên nhau (độ sâu). Bằng chứng gần đây [40, 43] tiết lộ rằng độ sâu mạng có tầm quan trọng đặc biệt, và những kết quả dẫn đầu [40, 43, 12, 16] về thử thách Tập dữ liệu ImageNet [35] đều khai thác các mô hình "rất sâu" [40], với độ sâu từ mười sáu [40] đến ba mươi [16]. Nhiều nhiệm vụ nhận dạng hình ảnh không tầm thường khác [7, 11, 6, 32, 27] cũ ng có



Hình 1. Lỗi huấn luyện (trái) và lỗi kiểm tra (phải) trên CIFAR-10 với mạng "đơn giản" 20 lớp và 56 lớp. Mạng lưới sâu hơn có lỗi huấn luyện cao hơn và do đó lỗi kiểm tra. Hiện tượng tương tự trên ImageNet được trình bày trong Hình 4.

được hưởng lợi rất nhiều từ các mô hình rất sâu sắc.

Bị thúc đẩy bởi tầm quan trọng của chiều sâu, một câu hỏi được đặt ra: Liệu học các mạng tốt hơn dễ dàng như xếp chồng nhiều lớp hơn? Trở ngại cho việc trả lời câu hỏi này là câu chuyện khét tiếng vấn đề biến mất/bùng nổ gradient [14, 1, 8], trong đó cản trở sự hội tụ ngay từ đầu. Vấn đề này, tuy nhiên, phần lớn đã được giải quyết bằng các lớp khởi tạo chuẩn hóa [23, 8, 36, 12] và các lớp chuẩn hóa trung gian [16], cho phép các mạng có hàng chục lớp bắt đầu hội tụ để giảm độ dốc ngẫu nhiên (SGD) với lan truyền ngược [22].

Khi các mạng sâu hơn có thể bắt đầu hội tụ, một vấn đề suy thoái đã được bộc lộ: với mạng tăng độ sâu, độ chính xác trở nên bão hòa (có thể không có gì đáng ngạc nhiên) và sau đó xuống cấp nhanh chóng. Thật bất ngờ, sự xuống cấp như vậy không phải do trang bị quá mức và việc thêm nhiều lớp hơn cho một mô hình có độ sâu phù hợp sẽ dẫn đến lỗi đào tạo cao hơn, như đã báo cáo trong [10, 41] và được xác minh kỹ lưỡng bởi thí nghiệm của chúng tôi. Hình 1 cho thấy một ví dụ điển hình.

Sự suy giảm (độ chính xác trong huấn luyện) chỉ ra rằng không tất cả các hệ thống đều dễ dàng tối ưu hóa như nhau. Chúng ta hãy xem xét một kiến trúc nông hơn và kiến trúc tương ứng sâu hơn của nó bổ sung thêm nhiều lớp hơn trên đó. Đã có giải pháp bằng cách xây dựng sang mô hình sâu hơn: các lớp được thêm vào là ánh xạ nhận dạng, và các lớp khác được sao chép từ phần nông hơn đã học người mẫu. Sự tồn tại của giải pháp được xây dựng này cho thấy rằng một mô hình sâu hơn sẽ không tạo ra lỗi huấn luyện cao hơn hơn so với đối tác nông hơn của nó. Nhưng thí nghiệm cho thấy rằng những người giải quyết hiện tại của chúng tôi không thể tìm ra giải pháp

1http://image-net.org/challenges/LSVRC/2015/ và http://mscoco.org/dataset/# detections-challenge2015.

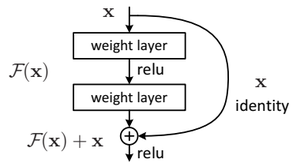


Figure 2. Residual learning: a building block.

are comparably good or better than the constructed solution (or unable to do so in feasible time).

In this paper, we address the degradation problem by introducing a *deep residual learning* framework. Instead of hoping each few stacked layers directly fit a desired underlying mapping, we explicitly let these layers fit a residual mapping. Formally, denoting the desired underlying mapping as $\mathcal{H}(\mathbf{x})$, we let the stacked nonlinear layers fit another mapping of $\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$. The original mapping is recast into $\mathcal{F}(\mathbf{x})+\mathbf{x}$. We hypothesize that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping. To the extreme, if an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers.

The formulation of $\mathcal{F}(\mathbf{x})+\mathbf{x}$ can be realized by feedforward neural networks with "shortcut connections" (Fig. 2). Shortcut connections [2, 33, 48] are those skipping one or more layers. In our case, the shortcut connections simply perform *identity* mapping, and their outputs are added to the outputs of the stacked layers (Fig. 2). Identity shortcut connections add neither extra parameter nor computational complexity. The entire network can still be trained end-to-end by SGD with backpropagation, and can be easily implemented using common libraries (*e.g.*, Caffe [19]) without modifying the solvers.

We present comprehensive experiments on ImageNet [35] to show the degradation problem and evaluate our method. We show that: 1) Our extremely deep residual nets are easy to optimize, but the counterpart "plain" nets (that simply stack layers) exhibit higher training error when the depth increases; 2) Our deep residual nets can easily enjoy accuracy gains from greatly increased depth, producing results substantially better than previous networks.

Similar phenomena are also shown on the CIFAR-10 set [20], suggesting that the optimization difficulties and the effects of our method are not just akin to a particular dataset. We present successfully trained models on this dataset with over 100 layers, and explore models with over 1000 layers.

On the ImageNet classification dataset [35], we obtain excellent results by extremely deep residual nets. Our 152-layer residual net is the deepest network ever presented on ImageNet, while still having lower complexity than VGG nets [40]. Our ensemble has **3.57%** top-5 error on the ImageNet *test* set, and *won the 1st place in the ILSVRC 2015 classification competition*. The extremely deep representations also have excellent generalization performance on other recognition tasks, and lead us to further *win the 1st places on: ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation* in ILSVRC & COCO 2015 competitions. This strong evidence shows that the residual learning principle is generic, and we expect that it is applicable in other vision and non-vision problems.

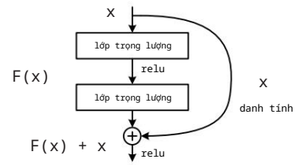
2. Related Work

Residual Representations. In image recognition, VLAD [18] is a representation that encodes by the residual vectors with respect to a dictionary, and Fisher Vector [30] can be formulated as a probabilistic version [18] of VLAD. Both of them are powerful shallow representations for image retrieval and classification [4, 47]. For vector quantization, encoding residual vectors [17] is shown to be more effective than encoding original vectors.

In low-level vision and computer graphics, for solving Partial Differential Equations (PDEs), the widely used Multigrid method [3] reformulates the system as subproblems at multiple scales, where each subproblem is responsible for the residual solution between a coarser and a finer scale. An alternative to Multigrid is hierarchical basis preconditioning [44, 45], which relies on variables that represent residual vectors between two scales. It has been shown [3, 44, 45] that these solvers converge much faster than standard solvers that are unaware of the residual nature of the solutions. These methods suggest that a good reformulation or preconditioning can simplify the optimization.

Shortcut Connections. Practices and theories that lead to shortcut connections [2, 33, 48] have been studied for a long time. An early practice of training multi-layer perceptrons (MLPs) is to add a linear layer connected from the network input to the output [33, 48]. In [43, 24], a few intermediate layers are directly connected to auxiliary classifiers for addressing vanishing/exploding gradients. The papers of [38, 37, 31, 46] propose methods for centering layer responses, gradients, and propagated errors, implemented by shortcut connections. In [43], an "inception" layer is composed of a shortcut branch and a few deeper branches.

Concurrent with our work, "highway networks" [41, 42] present shortcut connections with gating functions [15]. These gates are data-dependent and have parameters, in contrast to our identity shortcuts that are parameter-free. When a gated shortcut is "closed" (approaching zero), the layers in highway networks represent *non-residual* functions. On the contrary, our formulation always learns residual functions; our identity shortcuts are never closed, and all information is always passed through, with additional residual functions to be learned. In addition, high-



Hình 2. Học tập còn lại: một khối xây dựng.

tương đương tốt hoặc tốt hơn giải pháp được xây dựng
(hoặc không thể thực hiện được trong thời gian khả thi).

Trong bài báo này, chúng tôi giải quyết vấn đề suy thoái bằng cách giới thiệu một khuôn khổ học tập còn lại sâu sắc. Thay vì hy vọng mỗi lớp xếp chồng lên nhau sẽ phù hợp trực tiếp với một ánh xạ cơ bản mong muốn, chúng tôi rõ ràng để các lớp này khớp với ánh xạ dư. Về mặt hình thức, biểu thị mong muốn ánh xạ cơ bản là H(x), chúng ta để phi tuyến xếp chồng lên nhau các lớp phù hợp với một ánh xạ khác của F(x) := H(x) x. Ánh xạ ban đầu được viết lại thành F(x)+x. Chúng tôi đưa ra giả thuyết rằng việc tối ưu hóa ánh xạ dư dễ dàng hơn là tối ưu hóa ánh xạ ban đầu, không được tham chiếu. Đến mức cực đoan, nếu một ánh xạ nhận dạng là tối ưu, sẽ dễ dàng hơn để đẩy phần dư bằng 0 để phù hợp với ánh xạ nhận dạng theo ngăn xếp của các lớp phi tuyến.

Công thức của F(x) +x có thể được thực hiện bằng các mạng nơron cấp dữ liệu với "kết nối phím tắt" (Hình 2). Các kết nối tắt [2, 33, 48] là những kết nối bỏ qua một hoặc nhiều lớp hơn. Trong trường hợp của chúng tôi, các kết nối phím tắt chỉ đơn giản là thực hiện ánh xạ nhận dạng và kết quả đầu ra của chúng được thêm vào đầu ra của các lớp xếp chồng lên nhau (Hình 2). Các kết nối rút gọn nhận dạng không thêm tham số bổ sung cũ ng như độ phức tạp tính toán. Toàn bộ mạng vẫn có thể được đào tạo end-to-end bởi SGD với khả năng lan truyền ngược và có thể được triển khai dễ dàng bằng các thư viện phổ biến (ví dụ: Caffe [19]) mà không sửa đổi bộ giải.

Chúng tôi trình bày các thử nghiệm toàn diện trên ImageNet [35] để chỉ ra vấn đề suy thoái và đánh giá phương pháp. Chúng tôi chứng minh rằng: 1) Lưới dư cực kỳ sâu của chúng tôi rất dễ tối ưu hóa, nhưng các mạng "đơn giản" tương ứng (đó chỉ đơn giản là xếp chồng các lớp) thể hiện lỗi huấn luyện cao hơn khi độ sâu tăng lên; 2) Lưới dư sâu của chúng tôi có thể dễ dàng tận hưởng độ chính xác đạt được nhờ độ sâu tăng lên đáng kể, tạo ra kết quả tốt hơn đáng kể so với các mạng trước đó.

Hiện tượng tương tự cũ ng được thể hiện trên bộ CIFAR-10 [20], gợi ý rằng những khó khăn tối ưu hóa và tác dụng của phương pháp của chúng tôi không chỉ giống với một tập dữ liệu cụ thể. Chúng tôi trình bày các mô hình được đào tạo thành công trên tập dữ liệu này với hơn 100 lớp và khám phá các mô hình với hơn 1000 lớp.

Trên tập dữ liệu phân loại ImageNet [35], chúng tôi thu được kết quả tuyệt vời nhờ lưới dư cực kỳ sâu. 152- của chúng tôi mạng dư lớp là mạng sâu nhất từng được trình bày trên ImageNet, trong khi vẫn có độ phức tạp thấp hơn VGG lưới [40]. Nhóm của chúng tôi có 3,57% lỗi trong top 5 trên Bộ thử nghiệm ImageNet và giành vị trí số 1 trong ILSVRC Cuộc thi xếp hạng năm 2015 Những sự phản ánh cực kỳ sâu sắc cũ ng có hiệu suất khái quát hóa tuyệt vời vào các nhiệm vụ công nhận khác và giúp chúng tôi tiếp tục giành được Vị trí số 1 về: Phát hiện ImageNet, bản địa hóa ImageNet, Phát hiện COCO và phân đoạn COCO trong ILSVRC & Cuộc thi COCO 2015 Bằng chứng mạnh mẽ này cho thấy nguyên tắc học tập còn lại là chung chung và chúng tôi mong đợi rằng nó có thể áp dụng được trong các vấn đề về tầm nhìn và không tầm nhìn khác.

2. Công việc liên quan

Đại diện dư. Trong nhận dạng hình ảnh, VLAD [18] là biểu diễn được mã hóa bằng các vectơ dư đối với một từ điển, và Fisher Vector [30] có thể được xây dựng dưới dạng phiên bản xác suất [18] của VLAD. Cả hai trong số đó là các biểu diễn nông mạnh mẽ để truy xuất và phân loại hình ảnh [4, 47]. Để lượng tử hóa vector, mã hóa các vectơ dư [17] được chứng minh là hiệu quả hơn mã hóa các vectơ gốc.

Trong đồ họa máy tính và thị giác cấp thấp, để giải các phương trình vi phân từng phần (PDE), phương pháp được sử dụng rộng rãi Phương pháp đa lưới [3] xây dựng lại hệ thống thành các bài toán con ở nhiều tỷ lệ, trong đó mỗi bài toán con chịu trách nhiệm về giải pháp dư giữa một bài toán thô hơn và một bài toán mịn hơn tỉ lệ. Một giải pháp thay thế cho Multigrid là tiền điều hòa cơ sở phân cấp [44, 45], dựa trên các biến đại diện cho việc gửi vectơ dư giữa hai thang đo. Nó đã được hiển thị [3, 44, 45] rằng những bộ giải này hội tụ nhanh hơn nhiều so với những bộ giải tiêu chuẩn không biết về bản chất dư của các giải pháp. Những phương pháp này gợi ý rằng một sự cải cách tốt hoặc điều kiện tiên quyết có thể đơn giản hóa việc tối ưu hóa.

Kết nối lối tắt. Thực tiễn và lý thuyết dẫn đến kết nối phím tắt [2, 33, 48] đã được nghiên cứu từ lâu thời gian. Thực hành ban đầu về đào tạo các perceptron nhiều lớp (MLP) là thêm một lớp tuyến tính được kết nối từ mạng đầu vào thành đầu ra [33, 48]. Trong [43, 24], một số lớp trung gian được kết nối trực tiếp với các bộ phân loại phụ trợ để giải quyết các gradient biến mất/nổ. Những trang giấy của [38, 37, 31, 46] đề xuất các phương pháp phản hồi lại lớp trung tâm, độ dốc và lỗi lan truyền, được thực hiện bởi kết nối phím tắt. Trong [43], lớp "khởi động" bao gồm một nhánh tắt và một vài nhánh sâu hơn.

Đồng thời với công việc của chúng tôi, "mạng lưới đường cao tốc" [41, 42] trình bày các kết nối phím tắt với chức năng gấp [15]. Các cổng này phụ thuộc vào dữ liệu và có các tham số, trong tương phản với các phím tắt nhận dạng không có tham số của chúng tôi. Khi một lối tắt có cổng được "đóng" (gần bằng 0), các lớp trong mạng lưới đường cao tốc biểu diễn các chức năng không dư. Ngược lại, công thức của chúng tôi luôn học hỏi hàm dư; lối tắt nhận dạng của chúng ta không bao giờ bị đóng, và tất cả thông tin luôn được truyền qua, với các hàm dư bổ sung cần được học. Ngoài ra, cao

way networks have not demonstrated accuracy gains with extremely increased depth (*e.g.*, over 100 layers).

3. Deep Residual Learning

3.1. Residual Learning

Let us consider $\mathcal{H}(\mathbf{x})$ as an underlying mapping to be fit by a few stacked layers (not necessarily the entire net), with $\mathbf{x}$ denoting the inputs to the first of these layers. If one hypothesizes that multiple nonlinear layers can asymptotically approximate complicated functions[2], then it is equivalent to hypothesize that they can asymptotically approximate the residual functions, *i.e.*, $\mathcal{H}(\mathbf{x}) - \mathbf{x}$ (assuming that the input and output are of the same dimensions). So rather than expect stacked layers to approximate $\mathcal{H}(\mathbf{x})$, we explicitly let these layers approximate a residual function $\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$. The original function thus becomes $\mathcal{F}(\mathbf{x})+\mathbf{x}$. Although both forms should be able to asymptotically approximate the desired functions (as hypothesized), the ease of learning might be different.

This reformulation is motivated by the counterintuitive phenomena about the degradation problem (Fig. 1, left). As we discussed in the introduction, if the added layers can be constructed as identity mappings, a deeper model should have training error no greater than its shallower counterpart. The degradation problem suggests that the solvers might have difficulties in approximating identity mappings by multiple nonlinear layers. With the residual learning reformulation, if identity mappings are optimal, the solvers may simply drive the weights of the multiple nonlinear layers toward zero to approach identity mappings.

In real cases, it is unlikely that identity mappings are optimal, but our reformulation may help to precondition the problem. If the optimal function is closer to an identity mapping than to a zero mapping, it should be easier for the solver to find the perturbations with reference to an identity mapping, than to learn the function as a new one. We show by experiments (Fig. 7) that the learned residual functions in general have small responses, suggesting that identity mappings provide reasonable preconditioning.

3.2. Identity Mapping by Shortcuts

We adopt residual learning to every few stacked layers. A building block is shown in Fig. 2. Formally, in this paper we consider a building block defined as:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}. \tag{1}$$

Here $\mathbf{x}$ and $\mathbf{y}$ are the input and output vectors of the layers considered. The function $\mathcal{F}(\mathbf{x}, \{W_i\})$ represents the residual mapping to be learned. For the example in Fig. 2 that has two layers, $\mathcal{F} = W_2\sigma(W_1\mathbf{x})$ in which σ denotes ReLU [29] and the biases are omitted for simplifying notations. The operation $\mathcal{F} + \mathbf{x}$ is performed by a shortcut connection and element-wise addition. We adopt the second nonlinearity after the addition (*i.e.*, $\sigma(\mathbf{y})$, see Fig. 2).

The shortcut connections in Eqn.(1) introduce neither extra parameter nor computation complexity. This is not only attractive in practice but also important in our comparisons between plain and residual networks. We can fairly compare plain/residual networks that simultaneously have the same number of parameters, depth, width, and computational cost (except for the negligible element-wise addition).

The dimensions of $\mathbf{x}$ and $\mathcal{F}$ must be equal in Eqn.(1). If this is not the case (*e.g.*, when changing the input/output channels), we can perform a linear projection W_s by the shortcut connections to match the dimensions:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + W_s\mathbf{x}. \tag{2}$$

We can also use a square matrix W_s in Eqn.(1). But we will show by experiments that the identity mapping is sufficient for addressing the degradation problem and is economical, and thus W_s is only used when matching dimensions.

The form of the residual function $\mathcal{F}$ is flexible. Experiments in this paper involve a function $\mathcal{F}$ that has two or three layers (Fig. 5), while more layers are possible. But if $\mathcal{F}$ has only a single layer, Eqn.(1) is similar to a linear layer: $\mathbf{y} = W_1\mathbf{x} + \mathbf{x}$, for which we have not observed advantages.

We also note that although the above notations are about fully-connected layers for simplicity, they are applicable to convolutional layers. The function $\mathcal{F}(\mathbf{x}, \{W_i\})$ can represent multiple convolutional layers. The element-wise addition is performed on two feature maps, channel by channel.

3.3. Network Architectures

We have tested various plain/residual nets, and have observed consistent phenomena. To provide instances for discussion, we describe two models for ImageNet as follows.

Plain Network. Our plain baselines (Fig. 3, middle) are mainly inspired by the philosophy of VGG nets [40] (Fig. 3, left). The convolutional layers mostly have 3×3 filters and follow two simple design rules: (i) for the same output feature map size, the layers have the same number of filters; and (ii) if the feature map size is halved, the number of filters is doubled so as to preserve the time complexity per layer. We perform downsampling directly by convolutional layers that have a stride of 2. The network ends with a global average pooling layer and a 1000-way fully-connected layer with softmax. The total number of weighted layers is 34 in Fig. 3 (middle).

It is worth noticing that our model has *fewer* filters and *lower* complexity than VGG nets [40] (Fig. 3, left). Our 34-layer baseline has 3.6 billion FLOPs (multiply-adds), which is only 18% of VGG-19 (19.6 billion FLOPs).

[2]This hypothesis, however, is still an open question. See [28].

cách mạng chưa chứng minh được độ chính xác đạt được với
độ sâu cực kỳ tăng (ví dụ: hơn 100 lớp).

3. Học tập sâu còn sót lại

3.1. Học tập còn lại

Chúng ta hãy coi H(x) là một ánh xạ cơ bản
vừa với một vài lớp xếp chồng lên nhau (không nhất thiết phải là toàn bộ lưới),
với x biểu thị đầu vào của lớp đầu tiên trong số các lớp này. Nếu một
đưa ra giả thuyết rằng nhiều lớp phi tuyến có thể xấp xỉ tiệm cận
các hàm phức tạp2 , thì tương đương
với việc đưa ra giả thuyết rằng chúng có thể xấp xỉ tiệm cận các
hàm số dư, tức là H(x) x (giả sử rằng
đầu vào và đầu ra có cùng kích thước). Vì thế
thay vì mong đợi các lớp xếp chồng lên nhau sẽ xấp xỉ H(x), chúng ta
rõ ràng để các lớp này gần đúng với hàm dư
F(x) := H(x) x. Hàm ban đầu do đó trở thành
F(x)+x. Mặc dù cả hai dạng đều có thể tiệm cận gần đúng các hàm mong
muốn (như giả thuyết),
sự dễ dàng của việc học có thể khác nhau.

Sự cải cách này được thúc đẩy bởi sự phản trực giác
hiện tượng về vấn đề suy thoái (Hình 1, bên trái). BẰNG
chúng ta đã thảo luận trong phần giới thiệu, liệu các lớp được thêm vào có thể
được xây dựng dưới dạng ánh xạ nhận dạng, một mô hình sâu hơn sẽ
có lỗi huấn luyện không lớn hơn phần đối tác nông hơn của nó. Vấn
đề suy thoái cho thấy rằng người giải quyết
có thể gặp khó khăn trong việc ánh xạ nhận dạng gần đúng
bởi nhiều lớp phi tuyến. Với việc xây dựng lại việc học phần dư,
nếu ánh xạ nhận dạng là tối ưu thì người giải
có thể chỉ cần đẩy trọng số của nhiều lớp phi tuyến về 0 để tiếp
cận ánh xạ nhận dạng.

Trong các trường hợp thực tế, ánh xạ nhận dạng không chắc là tối ưu,
nhưng việc cải cách của chúng tôi có thể giúp tạo điều kiện tiên quyết cho
vấn đề. Nếu hàm tối ưu gần với danh tính hơn
ánh xạ hơn so với ánh xạ 0, nó sẽ dễ dàng hơn cho
người giải quyết để tìm ra những nhiễu loạn có liên quan đến một danh tính
lập bản đồ, hơn là tìm hiểu chức năng như một chức năng mới. Chúng tôi biểu diễn
bằng các thí nghiệm (Hình 7) rằng các hàm số dư đã học trong
chung có những phản hồi nhỏ, cho thấy rằng ping bản đồ nhận dạng
cung cấp điều kiện tiên quyết hợp lý.

3.2. Ánh xạ nhận dạng bằng phím tắt

Chúng tôi áp dụng phương pháp học còn lại cho mỗi vài lớp xếp chồng lên nhau.
Một khối xây dựng được hiển thị trong Hình 2. Về mặt hình thức, trong bài báo này
chúng tôi xem xét một khối xây dựng được xác định là:

y = F(x, {Wi}) + x. (1)

Ở đây x và y là vectơ đầu vào và đầu ra của các lớp được xem xét.
Hàm F(x, {Wi}) biểu thị
ánh xạ dư cần học. Ví dụ trong Hình 2
có hai lớp, F = W2σ(W1x) trong đó σ biểu thị

2Tuy nhiên, giả thuyết này vẫn là một câu hỏi mở. Xem [28].

ReLU [29] và các độ lệch được bỏ qua để đơn giản hóa việc không xác
định. Thao tác F + x được thực hiện bằng phím tắt
kết nối và bổ sung phần tử khôn ngoan. Chúng tôi chấp nhận tính phi
tuyến thứ hai sau phép cộng (tức là, σ(y), xem Hình 2).

Các kết nối phím tắt trong phương trình (1) không đưa ra tham số
bổ sung cũ ng như độ phức tạp tính toán. Đây không chỉ là
hấp dẫn trong thực tế nhưng cũ ng quan trọng trong so sánh của chúng tôi
giữa mạng đơn giản và mạng dư. Chúng ta có thể so sánh khá rõ ràng
các mạng đơn giản/dư đồng thời có
cùng số tham số, độ sâu, chiều rộng và chi phí tính toán (ngoại trừ
việc bổ sung phần tử không đáng kể).

Kích thước của x và F phải bằng nhau trong phương trình (1).
Nếu không đúng như vậy (ví dụ: khi thay đổi đầu vào/đầu ra
kênh), chúng ta có thể thực hiện phép chiếu tuyến tính Ws bằng
kết nối phím tắt để phù hợp với kích thước:

y = F(x, {Wi}) + Wsx. (2)

Chúng ta cũ ng có thể sử dụng ma trận vuông Ws trong phương trình (1). Nhưng chúng tôi sẽ
cho thấy bằng các thí nghiệm rằng ánh xạ nhận dạng là đủ
để giải quyết vấn đề suy thoái và mang lại hiệu quả kinh tế,
và do đó Ws chỉ được sử dụng khi khớp các kích thước.

Dạng của hàm dư F là linh hoạt. Các thí nghiệm trong bài báo này
liên quan đến hàm F có hai hoặc
ba lớp (Hình 5), trong khi có thể có nhiều lớp hơn. Nhưng nếu
F chỉ có một lớp duy nhất, Phương trình (1) tương tự như một lớp tuyến tính:
y = W1x + x, mà chúng tôi chưa quan sát thấy lợi thế.

Chúng tôi cũ ng lưu ý rằng mặc dù các ký hiệu trên là về
các lớp được kết nối đầy đủ để đơn giản, chúng có thể áp dụng cho
các lớp chập. Hàm F(x, {Wi}) có thể biểu diễn việc gửi nhiều lớp
chập. Việc bổ sung theo từng phần tử được thực hiện trên hai bản đồ
đặc trưng, theo từng kênh.

3.3. Kiến trúc mạng

Chúng tôi đã thử nghiệm nhiều loại lưới trơn/dư khác nhau và
quan sát thấy hiện tượng nhất quán. Để cung cấp các trường hợp thảo
luận, chúng tôi mô tả hai mô hình cho ImageNet như sau.

Mạng đơn giản. Đường cơ sở đơn giản của chúng tôi (Hình 3, ở giữa) là
chủ yếu lấy cảm hứng từ triết lý của lưới VGG [40] (Hình 3,
bên trái). Các lớp chập hầu hết có bộ lọc 3×3 và
tuân theo hai quy tắc thiết kế đơn giản: (i) cho cùng một đầu ra
kích thước bản đồ đặc trưng, các lớp có cùng số lượng bộ lọc; và
(ii) nếu kích thước bản đồ đặc trưng giảm một nửa thì số lượng bộ
lọc sẽ tăng gấp đôi để duy trì độ phức tạp thời gian trên mỗi lớp.
Chúng tôi thực hiện downsampling trực tiếp bằng cách
các lớp chập có bước tiến là 2. Mạng
kết thúc bằng lớp tổng hợp trung bình toàn cầu và 1000 chiều
lớp được kết nối đầy đủ với softmax. Tổng số
các lớp có trọng số là 34 trong Hình 3 (giữa).

Điều đáng chú ý là mô hình của chúng tôi có ít bộ lọc hơn và
độ phức tạp thấp hơn lưới VGG [40] (Hình 3, bên trái). 34- của chúng tôi
đường cơ sở của lớp có 3,6 tỷ FLOP (nhân lên), trong đó
chỉ bằng 18% VGG-19 (19,6 tỷ FLOP).

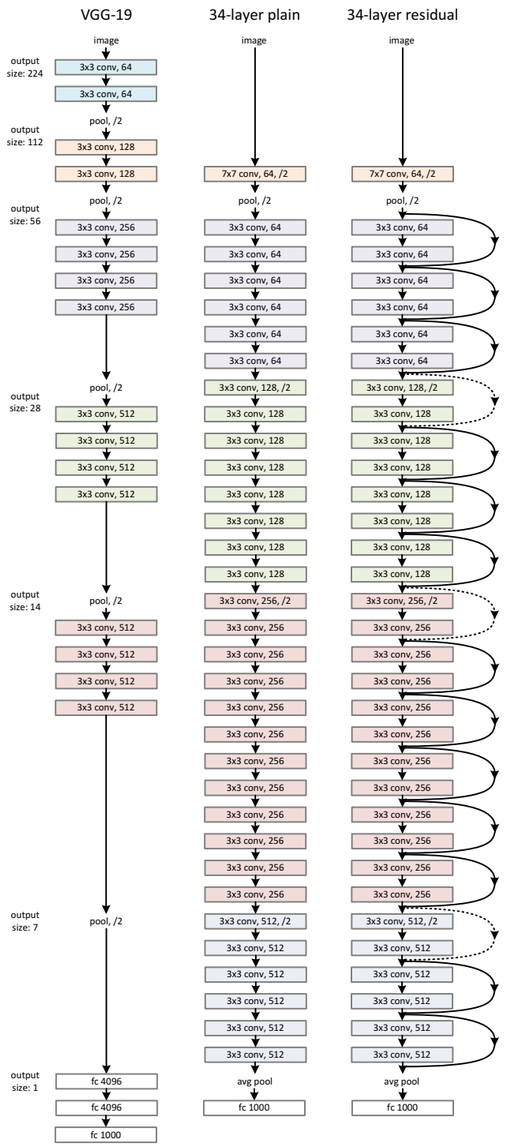


Figure 3. Example network architectures for ImageNet. **Left**: the VGG-19 model [40] (19.6 billion FLOPs) as a reference. **Middle**: a plain network with 34 parameter layers (3.6 billion FLOPs). **Right**: a residual network with 34 parameter layers (3.6 billion FLOPs). The dotted shortcuts increase dimensions. **Table 1** shows more details and other variants.

Residual Network. Based on the above plain network, we insert shortcut connections (Fig. 3, right) which turn the network into its counterpart residual version. The identity shortcuts (Eqn.(1)) can be directly used when the input and output are of the same dimensions (solid line shortcuts in Fig. 3). When the dimensions increase (dotted line shortcuts in Fig. 3), we consider two options: (A) The shortcut still performs identity mapping, with extra zero entries padded for increasing dimensions. This option introduces no extra parameter; (B) The projection shortcut in Eqn.(2) is used to match dimensions (done by 1×1 convolutions). For both options, when the shortcuts go across feature maps of two sizes, they are performed with a stride of 2.

3.4. Implementation

Our implementation for ImageNet follows the practice in [21, 40]. The image is resized with its shorter side randomly sampled in $[256, 480]$ for scale augmentation [40]. A 224×224 crop is randomly sampled from an image or its horizontal flip, with the per-pixel mean subtracted [21]. The standard color augmentation in [21] is used. We adopt batch normalization (BN) [16] right after each convolution and before activation, following [16]. We initialize the weights as in [12] and train all plain/residual nets from scratch. We use SGD with a mini-batch size of 256. The learning rate starts from 0.1 and is divided by 10 when the error plateaus, and the models are trained for up to 60×10^4 iterations. We use a weight decay of 0.0001 and a momentum of 0.9. We do not use dropout [13], following the practice in [16].

In testing, for comparison studies we adopt the standard 10-crop testing [21]. For best results, we adopt the fully-convolutional form as in [40, 12], and average the scores at multiple scales (images are resized such that the shorter side is in $\{224, 256, 384, 480, 640\}$).

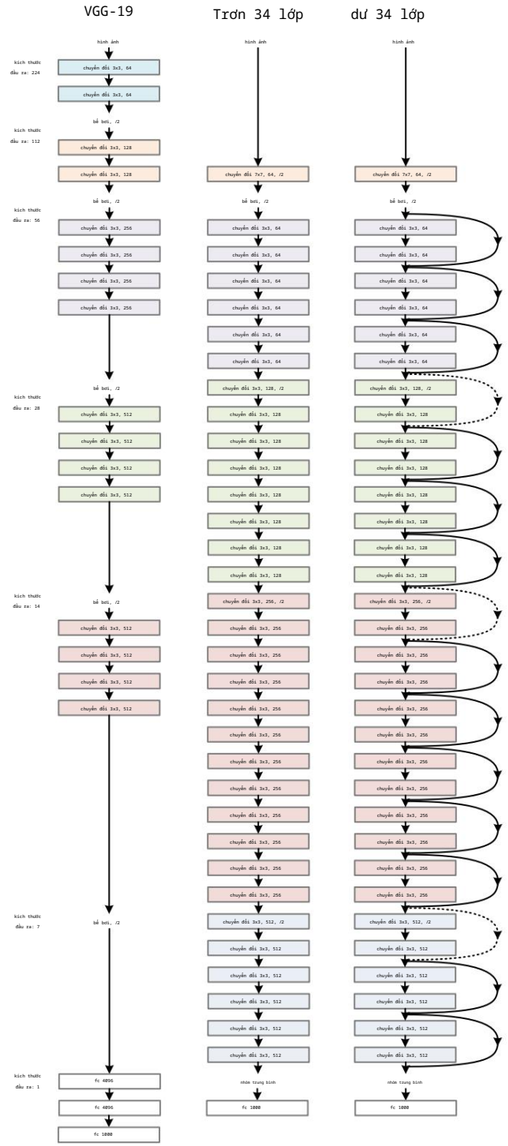
4. Experiments

4.1. ImageNet Classification

We evaluate our method on the ImageNet 2012 classification dataset [35] that consists of 1000 classes. The models are trained on the 1.28 million training images, and evaluated on the 50k validation images. We also obtain a final result on the 100k test images, reported by the test server. We evaluate both top-1 and top-5 error rates.

Plain Networks. We first evaluate 18-layer and 34-layer plain nets. The 34-layer plain net is in Fig. 3 (middle). The 18-layer plain net is of a similar form. See Table 1 for detailed architectures.

The results in Table 2 show that the deeper 34-layer plain net has higher validation error than the shallower 18-layer plain net. To reveal the reasons, in Fig. 4 (left) we compare their training/validation errors during the training procedure. We have observed the degradation problem - the



Hình 3. Ví dụ về kiến trúc mạng cho ImageNet. Bên trái: mô hình VGG-19 [40] (19,6 tỷ FLOP) làm tham chiếu. Ở giữa: một mạng đơn giản với 34 lớp tham số (3,6 tỷ FLOP).
Bên phải: mạng dư có 34 lớp tham số (3,6 tỷ FLOP). Các phím tắt chấm tăng kích thước. Bảng 1 cho thấy nhiều chi tiết hơn và các biến thể khác.

Mạng dư. Dựa trên mạng đơn giản ở trên, chúng tôi chèn các kết nối phím tắt (Hình 3, bên phải) để biến mạng thành phiên bản còn lại đối tác của nó. Các phím tắt nhận dạng (Eqn.(1)) có thể được sử dụng trực tiếp khi đầu vào và đầu ra có cùng kích thước (các phím tắt đường liền nét trong Hình 3). Khi kích thước tăng (các phím tắt đường chấm trong Hình 3), chúng tôi xem xét hai tùy chọn: (A) Phím tắt vẫn thực hiện ánh xạ nhận dạng, với các mục nhập số 0 bổ sung được đệm để tăng kích thước. Tùy chọn này không giới thiệu thêm tham số nào; (B) Phím tắt phép chiếu trong phương trình (2) được sử dụng để khớp với các kích thước (được thực hiện bằng phép cuộn 1×1). Đối với cả hai tùy chọn, khi các phím tắt đi qua các bản đồ đặc điểm có hai kích thước, chúng sẽ được thực hiện với bước tiến là 2.

3.4. Thực hiện

Việc triển khai ImageNet của chúng tôi tuân theo thông lệ trong [21, 40]. Hình ảnh được thay đổi kích thước với cạnh ngắn hơn được lấy mẫu ngẫu nhiên trong [256, 480] để tăng tỷ lệ [40].
Phần cắt 224×224 được lấy mẫu ngẫu nhiên từ một hình ảnh hoặc ảnh lật ngang của nó, với giá trị trung bình trên mỗi pixel bị trừ đi [21]. Việc tăng màu tiêu chuẩn trong [21] được sử dụng. Chúng tôi áp dụng chuẩn hóa hàng loạt (BN) [16] ngay sau mỗi lần tích chập và trước khi kích hoạt, theo sau [16]. Chúng tôi khởi tạo các trọng số như trong [12] và huấn luyện tất cả các lưới trơn/dư từ đầu. Chúng tôi sử dụng SGD với kích thước lô nhỏ là 256. Tốc độ học bắt đầu từ 0,1 và được chia cho 10 khi lỗi ổn định và các mô hình được huấn luyện cho số lần lặp lên tới 60 × 104 . Chúng tôi sử dụng độ giảm trọng lượng là 0,0001 và động lượng là 0,9. Chúng tôi không sử dụng dropout [13], theo thực tế trong [16].

Trong thử nghiệm, đối với các nghiên cứu so sánh, chúng tôi áp dụng thử nghiệm tiêu chuẩn 10 vụ [21]. Để có kết quả tốt nhất, chúng tôi áp dụng dạng tích chập đầy đủ như trong [40, 12] và tính điểm trung bình ở nhiều tỷ lệ (hình ảnh được thay đổi kích thước sao cho cạnh ngắn hơn là {224, 256, 384, 480, 640}).

4. Thí nghiệm

4.1. Phân loại ImageNet

Chúng tôi đánh giá phương pháp của mình trên bộ dữ liệu phân loại ImageNet 2012 [35] bao gồm 1000 lớp. Các mô hình được đào tạo trên 1,28 triệu hình ảnh đào tạo và được đánh giá dựa trên 50 nghìn hình ảnh xác thực. Chúng tôi cũ ng thu được kết quả cuối cùng trên 100k hình ảnh thử nghiệm do máy chủ thử nghiệm báo cáo.
Chúng tôi đánh giá cả tỷ lệ lỗi top 1 và top 5.

Mạng đơn giản. Đầu tiên chúng tôi đánh giá lưới trơn 18 lớp và 34 lớp. Lưới trơn 34 lớp như trong Hình 3 (giữa). Lưới trơn 18 lớp có dạng tương tự. Xem Bảng 1 để biết kiến trúc chi tiết.

Kết quả trong Bảng 2 cho thấy lưới trơn 34 lớp sâu hơn có lỗi xác thực cao hơn lưới trơn 18 lớp nông hơn. Để tiết lộ lý do, trong Hình 4 (trái), chúng tôi so sánh các lỗi đào tạo/xác thực của họ trong quá trình đào tạo. Chúng tôi đã quan sát thấy vấn đề xuống cấp -

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Table 1. Architectures for ImageNet. Building blocks are shown in brackets (see also Fig. 5), with the numbers of blocks stacked. Down-sampling is performed by conv3_1, conv4_1, and conv5_1 with a stride of 2.

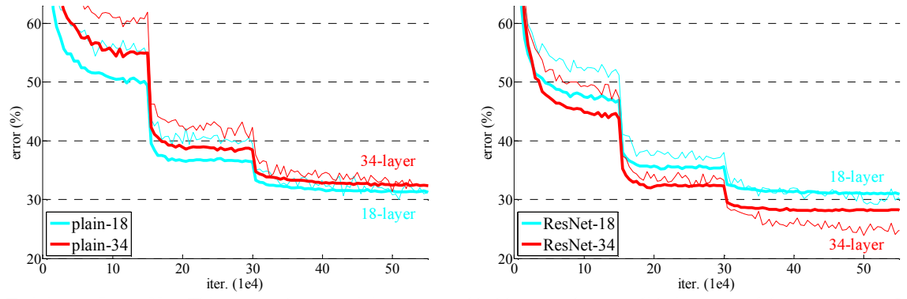


Figure 4. Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

	plain	ResNet
18 layers	27.94	27.88
34 layers	28.54	**25.03**

Table 2. Top-1 error (%, 10-crop testing) on ImageNet validation. Here the ResNets have no extra parameter compared to their plain counterparts. Fig. 4 shows the training procedures.

34-layer plain net has higher *training* error throughout the whole training procedure, even though the solution space of the 18-layer plain network is a subspace of that of the 34-layer one.

We argue that this optimization difficulty is *unlikely* to be caused by vanishing gradients. These plain networks are trained with BN [16], which ensures forward propagated signals to have non-zero variances. We also verify that the backward propagated gradients exhibit healthy norms with BN. So neither forward nor backward signals vanish. In fact, the 34-layer plain net is still able to achieve competitive accuracy (Table 3), suggesting that the solver works to some extent. We conjecture that the deep plain nets may have exponentially low convergence rates, which impact the reducing of the training error[3]. The reason for such optimization difficulties will be studied in the future.

Residual Networks. Next we evaluate 18-layer and 34-layer residual nets (*ResNets*). The baseline architectures are the same as the above plain nets, expect that a shortcut connection is added to each pair of 3×3 filters as in Fig. 3 (right). In the first comparison (Table 2 and Fig. 4 right), we use identity mapping for all shortcuts and zero-padding for increasing dimensions (option A). So they have *no extra parameter* compared to the plain counterparts.

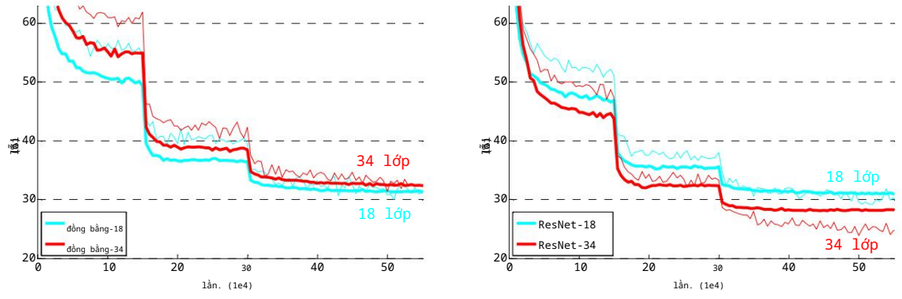
We have three major observations from Table 2 and Fig. 4. First, the situation is reversed with residual learning – the 34-layer ResNet is better than the 18-layer ResNet (by 2.8%). More importantly, the 34-layer ResNet exhibits considerably lower training error and is generalizable to the validation data. This indicates that the degradation problem is well addressed in this setting and we manage to obtain accuracy gains from increased depth.

Second, compared to its plain counterpart, the 34-layer

[3]We have experimented with more training iterations (3×) and still observed the degradation problem, suggesting that this problem cannot be feasibly addressed by simply using more iterations.

tên lớp kích thước đầu ra	18 lớp	34 lớp	50 lớp	101 lớp	152 lớp
conv1 112×112	7×7, 64, sải bước 2				
	Bể bơi tối đa 3 × 3, sải chân 2				
đối số2 x 56×56	3×3, 64 ×2 3×3, 64	3×3, 64 ×3 3×3, 64	1×1, 64 3×3, 64 1×1, 256 ×3	1×1, 64 3×3, 64 1×1, 256 ×3	1×1, 64 3×3, 64 1×1, 256 ×3
đối đối3 x 28×28	3×3, 128 ×2 3×3, 128	3×3, 128 ×4 3×3, 128	1×1, 128 3×3, 128 1×1, 512 ×4	1×1, 128 3×3, 128 1×1, 512 ×4	1×1, 128 3×3, 128 1×1, 512 ×8
conv4_x 14×14	3×3, 256 ×2 3×3, 256	3×3, 256 ×6 3×3, 256	1×1, 256 3×3, 256 1×1, 1024 ×6	1×1, 256 3×3, 256 1×1, 1024 ×23	1×1, 256 3×3, 256 1×1, 1024 ×36
chuyển đổi5 x 7×7	3×3, 512 ×2 3×3, 512	3×3, 512 ×3 3×3, 512	1×1, 512 3×3, 512 1×1, 2048 ×3	1×1, 512 3×3, 512 1×1, 2048 ×3	1×1, 512 3×3, 512 1×1, 2048 ×3
1×1	hồ bơi trung bình, 1000-d fc, softmax				
FLOP	1,8×109	3,6×109	3,8×109	7,6×109	11,3×109

Bảng 1. Kiến trúc cho ImageNet. Các khối xây dựng được hiển thị trong ngoặc (xem thêm Hình 5), với số khối xếp chồng lên nhau. Lấy mẫu xuống được thực hiện bởi conv3 1, conv4 1 và conv5 1 với bước nhảy là 2.



Hình 4. Đào tạo trên ImageNet. Các đường cong mỏng biểu thị lỗi huấn luyện và các đường cong đậm biểu thị lỗi xác thực của các phần trung tâm. Trái: trơn mạng 18 và 34 lớp. Phải: ResNets gồm 18 và 34 lớp. Trong biểu đồ này, các mạng dư không có tham số bổ sung so với đối tác đơn giản của họ.

	đơn giản	ResNet
18 lớp 34	27,94	27,88
lớp	28,54	25.03

Bảng 2. Lỗi Top-1 (%, thử nghiệm 10 vụ) khi xác thực ImageNet. Ở đây ResNets không có tham số bổ sung so với thông số đơn giản của chúng đối tác. Hình 4 thể hiện các quy trình đào tạo.

Lưới trơn 34 lớp có lỗi đào tạo cao hơn trong suốt toàn bộ quy trình đào tạo, mặc dù không gian giải pháp của mạng đơn giản 18 lớp là không gian con của mạng 34 lớp một.

Chúng tôi lập luận rằng khó khăn tối ưu hóa này khó có thể xảy ra được gây ra bởi độ dốc biến mất. Những mạng đơn giản này là được huấn luyện với BN [16], đảm bảo việc truyền về phía trước tín hiệu có phương sai khác 0. Chúng tôi cũ ng xác minh rằng độ dốc lan truyền ngược thể hiện các tiêu chuẩn lành mạnh với BN. Vì vậy, cả tín hiệu tiến và lùi đều không biến mất. TRONG trên thực tế, mạng trơn 34 lớp vẫn có thể đạt được độ chính xác cạnh tranh (Bảng 3), cho thấy rằng bộ giải hoạt động đến một mức độ nào. Chúng tôi phỏng đoán rằng các lưới đồng bằng sâu có thể có tốc độ hội tụ thấp theo cấp số nhân, điều này ảnh hưởng đến giảm lỗi đào tạo3 · Sở dĩ có sự tối ưu như vậy những khó khăn trong việc thu thập sẽ được nghiên cứu trong tương lai.

Mạng dư. Tiếp theo chúng tôi đánh giá 18 lớp và 34-lưới dư lớp (ResNets). Các kiến trúc cơ bản giống như các lưới đơn giản ở trên, hy vọng rằng có một phím tắt kết nối được thêm vào từng cặp bộ lọc 3×3 như trong Hình 3 (Phải). Trong lần so sánh đầu tiên (Bảng 2 và Hình 4 bên phải), chúng tôi sử dụng ánh xạ nhận dạng cho tất cả các phím tắt và khoảng đệm bằng 0 để tăng kích thước (tùy chọn A). Vì vậy họ không có thêm tham số so với các đối tác đơn giản.

Chúng tôi có ba quan sát chính từ Bảng 2 và Hình 4. Đầu tiên, tình huống được đảo ngược với việc học còn lại – ResNet 34 lớp tốt hơn ResNet 18 lớp (tăng 2,8%). Quan trọng hơn, triển lãm ResNet 34 lớp lỗi huấn luyện thấp hơn đáng kể và có thể khái quát hóa cho dữ liệu xác nhận. Điều này cho thấy vấn đề suy thoái được giải quyết tốt trong bối cảnh này và chúng tôi quản lý để có được độ chính xác đạt được từ độ sâu tăng lên.

Thứ hai, so với phiên bản đơn giản của nó, 34 lớp

3Chúng tôi đã thử nghiệm với nhiều lần lặp huấn luyện hơn (3×) và vẫn quan sát được vấn đề xuống cấp, cho thấy rằng vấn đề này không thể giải quyết được được giải quyết một cách khả thi bằng cách sử dụng nhiều lần lặp hơn.

model	top-1 err.	top-5 err.
VGG-16 [40]	28.07	9.33
GoogLeNet [43]	-	9.15
PReLU-net [12]	24.27	7.38
plain-34	28.54	10.02
ResNet-34 A	25.03	7.76
ResNet-34 B	24.52	7.46
ResNet-34 C	24.19	7.40
ResNet-50	22.85	6.71
ResNet-101	21.75	6.05
ResNet-152	**21.43**	**5.71**

Table 3. Error rates (%, **10-crop** testing) on ImageNet validation. VGG-16 is based on our test. ResNet-50/101/152 are of option B that only uses projections for increasing dimensions.

method	top-1 err.	top-5 err.
VGG [40] (ILSVRC'14)	-	8.43†
GoogLeNet [43] (ILSVRC'14)	-	7.89
VGG [40] (v5)	24.4	7.1
PReLU-net [12]	21.59	5.71
BN-inception [16]	21.99	5.81
ResNet-34 B	21.84	5.71
ResNet-34 C	21.53	5.60
ResNet-50	20.74	5.25
ResNet-101	19.87	4.60
ResNet-152	**19.38**	**4.49**

Table 4. Error rates (%) of **single-model** results on the ImageNet validation set (except † reported on the test set).

method	top-5 err. (**test**)
VGG [40] (ILSVRC'14)	7.32
GoogLeNet [43] (ILSVRC'14)	6.66
VGG [40] (v5)	6.8
PReLU-net [12]	4.94
BN-inception [16]	4.82
ResNet (ILSVRC'15)	**3.57**

Table 5. Error rates (%) of **ensembles**. The top-5 error is on the test set of ImageNet and reported by the test server.

ResNet reduces the top-1 error by 3.5% (Table 2), resulting from the successfully reduced training error (Fig. 4 right *vs*. left). This comparison verifies the effectiveness of residual learning on extremely deep systems.

Last, we also note that the 18-layer plain/residual nets are comparably accurate (Table 2), but the 18-layer ResNet converges faster (Fig. 4 right *vs*. left). When the net is "not overly deep" (18 layers here), the current SGD solver is still able to find good solutions to the plain net. In this case, the ResNet eases the optimization by providing faster convergence at the early stage.

Identity *vs*. Projection Shortcuts. We have shown that

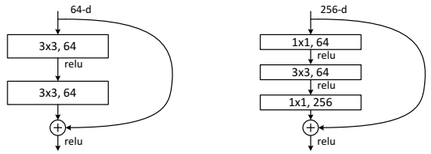


Figure 5. A deeper residual function $\mathcal{F}$ for ImageNet. Left: a building block (on 56×56 feature maps) as in Fig. 3 for ResNet-34. Right: a "bottleneck" building block for ResNet-50/101/152.

parameter-free, identity shortcuts help with training. Next we investigate projection shortcuts (Eqn.(2)). In Table 3 we compare three options: (A) zero-padding shortcuts are used for increasing dimensions, and all shortcuts are parameter-free (the same as Table 2 and Fig. 4 right); (B) projection shortcuts are used for increasing dimensions, and other shortcuts are identity; and (C) all shortcuts are projections.

Table 3 shows that all three options are considerably better than the plain counterpart. B is slightly better than A. We argue that this is because the zero-padded dimensions in A indeed have no residual learning. C is marginally better than B, and we attribute this to the extra parameters introduced by many (thirteen) projection shortcuts. But the small differences among A/B/C indicate that projection shortcuts are not essential for addressing the degradation problem. So we do not use option C in the rest of this paper, to reduce memory/time complexity and model sizes. Identity shortcuts are particularly important for not increasing the complexity of the bottleneck architectures that are introduced below.

Deeper Bottleneck Architectures. Next we describe our deeper nets for ImageNet. Because of concerns on the training time that we can afford, we modify the building block as a *bottleneck* design[4]. For each residual function $\mathcal{F}$, we use a stack of 3 layers instead of 2 (Fig. 5). The three layers are 1×1, 3×3, and 1×1 convolutions, where the 1×1 layers are responsible for reducing and then increasing (restoring) dimensions, leaving the 3×3 layer a bottleneck with smaller input/output dimensions. Fig. 5 shows an example, where both designs have similar time complexity.

The parameter-free identity shortcuts are particularly important for the bottleneck architectures. If the identity shortcut in Fig. 5 (right) is replaced with projection, one can show that the time complexity and model size are doubled, as the shortcut is connected to the two high-dimensional ends. So identity shortcuts lead to more efficient models for the bottleneck designs.

50-layer ResNet: We replace each 2-layer block in the

[4]Deeper *non*-bottleneck ResNets (*e.g*., Fig. 5 left) also gain accuracy from increased depth (as shown on CIFAR-10), but are not as economical as the bottleneck ResNets. So the usage of bottleneck designs is mainly due to practical considerations. We further note that the degradation problem of plain nets is also witnessed for the bottleneck designs.

người mẫu	lỗi top-1.	lỗi top 5.
VGG-16 [40]	28.07	9:33
GoogLeLeNet [43]	-	9 giờ 15
PReLU-net [12]	24,27	7,38
đồng bằng-34	28,54	10.02
ResNet-34 A	25.03	7,76
ResNet-34 B	24,52	7,46
ResNet-34 C	24.19	7 giờ 40
ResNet-50	22,85	6,71
ResNet-101	21,75	6.05
ResNet-152	21,43	5,71

Bảng 3. Tỷ lệ lỗi (%, thử nghiệm 10 vụ) khi xác thực ImageNet. VGG-16 dựa trên thử nghiệm của chúng tôi. ResNet-50/101/152 thuộc tùy chọn B chỉ sử dụng các phép chiếu để tăng kích thước.

phương pháp	lỗi top-1.	lỗi top 5.
VGG [40] (ILSVRC'14)	-	8,43†
GoogLeNet [43] (ILSVRC'14)	-	7,89
VGG [40] (v5)	24,4	7.1
PReLU-net [12]	21,59	5,71
BN-khởi đầu [16]	21,99	5,81
ResNet-34 B	21.84	5,71
ResNet-34 C	21.53	5,60
ResNet-50	20,74	5,25
ResNet-101	19,87	4,60
ResNet-152	19:38	4,49

Bảng 4. Tỷ lệ lỗi (%) của kết quả một mô hình trên ImageNet bộ xác thực (ngoại trừ † được báo cáo trên tập kiểm tra).

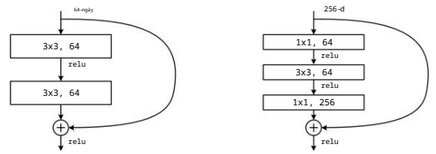
phương pháp	lỗi top 5. (Bài kiểm tra)
VGG [40] (ILSVRC'14)	7,32
GoogLeNet [43] (ILSVRC'14)	6,66
VGG [40] (v5)	6,8
PReLU-net [12]	4,94
BN-khởi đầu [16]	4,82
ResNet (ILSVRC'15)	3,57

Bảng 5. Tỷ lệ lỗi (%) của các nhóm. Lỗi top 5 nằm ở bộ thử nghiệm của ImageNet và được máy chủ thử nghiệm báo cáo.

ResNet giảm 3,5% lỗi top 1 (Bảng 2), dẫn đến từ việc giảm lỗi đào tạo thành công (Hình 4 bên phải so với bên phải). bên trái). Sự so sánh này xác minh tính hiệu quả của dư lượng học tập trên các hệ thống cực kỳ sâu sắc.

Cuối cùng, chúng tôi cũ ng lưu ý rằng lưới trơn/lưới dư 18 lớp tương đối chính xác (Bảng 2), nhưng ResNet 18 lớp hội tụ nhanh hơn (Hình 4 phải và trái). Khi mạng "không quá sâu" (18 lớp ở đây), bộ giải SGD hiện tại vẫn còn có thể tìm ra giải pháp tốt cho mạng đơn giản. Trong trường hợp này, ResNet giảm bớt việc tối ưu hóa bằng cách cung cấp khả năng hội tụ nhanh hơn ở giai đoạn đầu.

Các phím tắt nhận dạng và chiếu. Chúng tôi đã cho thấy điều đó



Hình 5. Hàm dư F sâu hơn cho ImageNet. Để lại một khối xây dựng (trên bản đồ đặc trưng 56×56) như trong Hình 3 cho ResNet-34. Phải: khối xây dựng "nút thắt cổ chai" cho ResNet-50/101/152.

không có tham số, các phím tắt nhận dạng giúp ích cho việc đào tạo. Kế tiếp chúng tôi điều tra các phím tắt chiếu (Eqn.(2)). Trong Bảng 3 chúng tôi so sánh ba tùy chọn: (A) các phím tắt không đệm được sử dụng để tăng kích thước và tất cả các phím tắt đều không có tham số (giống như Bảng 2 và Hình 4 bên phải); (B) các phím tắt chiếu được sử dụng để tăng kích thước và các phím tắt khác phím tắt là danh tính; và (C) tất cả các phím tắt đều là hình chiếu.

Bảng 3 cho thấy cả ba lựa chọn đều tốt hơn đáng kể so với lựa chọn thông thường. B tốt hơn A một chút. Chúng tôi lập luận rằng điều này là do kích thước không đệm trong A thực sự không có học tập dư thừa. C tốt hơn một chút so với B, và chúng tôi gán điều này cho các tham số bổ sung được giới thiệu bằng nhiều (mười ba) phím tắt trình chiếu. Nhưng những khác biệt nhỏ giữa A/B/C cho thấy rằng các phím tắt trình chiếu là không cần thiết để giải quyết vấn đề suy thoái. Vì vậy chúng tôi không sử dụng tùy chọn C trong phần còn lại của bài viết này để giảm độ phức tạp của bộ nhớ/thời gian và kích thước mô hình. Phím tắt nhận dạng là đặc biệt quan trọng để không làm tăng độ phức tạp của các kiến trúc thắt cổ chai được giới thiệu dưới đây.

Kiến trúc nút cổ chai sâu hơn. Tiếp theo chúng tôi mô tả mạng lưới sâu hơn cho ImageNet. Vì lo ngại về thời gian đào tạo mà chúng tôi có thể chi trả được, chúng tôi đã sửa đổi khối xây dựng như một thiết kế nút cổ chai4 . Đối với mỗi hàm dư F, chúng ta sử dụng chồng 3 lớp thay vì 2 lớp (Hình 5). Ba lớp là các phép chập 1×1, 3×3 và 1×1, trong đó các lớp 1×1 chịu trách nhiệm giảm và sau đó tăng (khôi phục) kích thước, khiến lớp 3 × 3 trở thành nút cổ chai với kích thước nhỏ hơn kích thước đầu vào/đầu ra. Hình 5 cho thấy một ví dụ, trong đó cả hai thiết kế đều có độ phức tạp về thời gian tương tự nhau.

Các lối tắt nhận dạng không có tham số đặc biệt quan trọng đối với các kiến trúc thắt cổ chai. Nếu lối tắt nhận dạng trong Hình 5 (phải) được thay thế bằng phép chiếu, người ta có thể cho thấy độ phức tạp về thời gian và kích thước mô hình tăng gấp đôi, vì phím tắt được kết nối với hai chiều cao kết thúc. Vì vậy, các lối tắt nhận dạng dẫn đến các mô hình hiệu quả hơn đối với các thiết kế nút cổ chai.

ResNet 50 lớp: Chúng tôi thay thế từng khối 2 lớp trong

4ResNets không bị tắc nghẽn sâu hơn (ví dụ: Hình 5 bên trái) cũ ng đạt được độ chính xác từ độ sâu tăng lên (như thể hiện trên CIFAR-10), nhưng không kinh tế bằng như nút cổ chai ResNets. Vì vậy việc sử dụng các thiết kế thắt cổ chai chủ yếu là do đến những cân nhắc thực tế. Chúng tôi lưu ý thêm rằng vấn đề suy thoái lưới trơn cũ ng được chứng minh cho các thiết kế cổ chai.

34-layer net with this 3-layer bottleneck block, resulting in a 50-layer ResNet (Table 1). We use option B for increasing dimensions. This model has 3.8 billion FLOPs.

101-layer and 152-layer ResNets: We construct 101-layer and 152-layer ResNets by using more 3-layer blocks (Table 1). Remarkably, although the depth is significantly increased, the 152-layer ResNet (11.3 billion FLOPs) still has *lower complexity* than VGG-16/19 nets (15.3/19.6 billion FLOPs).

The 50/101/152-layer ResNets are more accurate than the 34-layer ones by considerable margins (Table 3 and 4). We do not observe the degradation problem and thus enjoy significant accuracy gains from considerably increased depth. The benefits of depth are witnessed for all evaluation metrics (Table 3 and 4).

Comparisons with State-of-the-art Methods. In Table 4 we compare with the previous best single-model results. Our baseline 34-layer ResNets have achieved very competitive accuracy. Our 152-layer ResNet has a single-model top-5 validation error of 4.49%. This single-model result outperforms all previous ensemble results (Table 5). We combine six models of different depth to form an ensemble (only with two 152-layer ones at the time of submitting). This leads to **3.57%** top-5 error on the test set (Table 5). *This entry won the 1st place in ILSVRC 2015.*

4.2. CIFAR-10 and Analysis

We conducted more studies on the CIFAR-10 dataset [20], which consists of 50k training images and 10k testing images in 10 classes. We present experiments trained on the training set and evaluated on the test set. Our focus is on the behaviors of extremely deep networks, but not on pushing the state-of-the-art results, so we intentionally use simple architectures as follows.

The plain/residual architectures follow the form in Fig. 3 (middle/right). The network inputs are 32×32 images, with the per-pixel mean subtracted. The first layer is 3×3 convolutions. Then we use a stack of $6n$ layers with 3×3 convolutions on the feature maps of sizes $\{32, 16, 8\}$ respectively, with $2n$ layers for each feature map size. The numbers of filters are $\{16, 32, 64\}$ respectively. The subsampling is performed by convolutions with a stride of 2. The network ends with a global average pooling, a 10-way fully-connected layer, and softmax. There are totally $6n$+2 stacked weighted layers. The following table summarizes the architecture:

output map size	32×32	16×16	8×8
# layers	1+2n	2n	2n
# filters	16	32	64

When shortcut connections are used, they are connected to the pairs of 3×3 layers (totally $3n$ shortcuts). On this dataset we use identity shortcuts in all cases (*i.e.*, option A),

method			error (%)
Maxout [9]			9.38
NIN [25]			8.81
DSN [24]			8.22
	# layers	# params	
FitNet [34]	19	2.5M	8.39
Highway [41, 42]	19	2.3M	7.54 (7.72±0.16)
Highway [41, 42]	32	1.25M	8.80
ResNet	20	0.27M	8.75
ResNet	32	0.46M	7.51
ResNet	44	0.66M	7.17
ResNet	56	0.85M	6.97
ResNet	110	1.7M	**6.43** (6.61±0.16)
ResNet	1202	19.4M	7.93

Table 6. Classification error on the **CIFAR-10** test set. All methods are with data augmentation. For ResNet-110, we run it 5 times and show "best (mean±std)" as in [42].

so our residual models have exactly the same depth, width, and number of parameters as the plain counterparts.

We use a weight decay of 0.0001 and momentum of 0.9, and adopt the weight initialization in [12] and BN [16] but with no dropout. These models are trained with a mini-batch size of 128 on two GPUs. We start with a learning rate of 0.1, divide it by 10 at 32k and 48k iterations, and terminate training at 64k iterations, which is determined on a 45k/5k train/val split. We follow the simple data augmentation in [24] for training: 4 pixels are padded on each side, and a 32×32 crop is randomly sampled from the padded image or its horizontal flip. For testing, we only evaluate the single view of the original 32×32 image.

We compare $n = \{3, 5, 7, 9\}$, leading to 20, 32, 44, and 56-layer networks. Fig. 6 (left) shows the behaviors of the plain nets. The deep plain nets suffer from increased depth, and exhibit higher training error when going deeper. This phenomenon is similar to that on ImageNet (Fig. 4, left) and on MNIST (see [41]), suggesting that such an optimization difficulty is a fundamental problem.

Fig. 6 (middle) shows the behaviors of ResNets. Also similar to the ImageNet cases (Fig. 4, right), our ResNets manage to overcome the optimization difficulty and demonstrate accuracy gains when the depth increases.

We further explore $n = 18$ that leads to a 110-layer ResNet. In this case, we find that the initial learning rate of 0.1 is slightly too large to start converging[5]. So we use 0.01 to warm up the training until the training error is below 80% (about 400 iterations), and then go back to 0.1 and continue training. The rest of the learning schedule is as done previously. This 110-layer network converges well (Fig. 6, middle). It has *fewer* parameters than other deep and thin

[5]With an initial learning rate of 0.1, it starts converging (<90% error) after several epochs, but still reaches similar accuracy.

Lưới 34 lớp với khối thắt cổ chai 3 lớp này, dẫn đến
ResNet 50 lớp (Bảng 1). Chúng ta sử dụng phương án B để tăng
kích thước. Mô hình này có 3,8 tỷ FLOP.

ResNet 101 lớp và 152 lớp: Chúng tôi xây dựng 101-
lớp và ResNet 152 lớp bằng cách sử dụng nhiều khối 3 lớp hơn
(Bảng 1). Điều đáng chú ý là mặc dù độ sâu rất lớn
tăng lên, ResNet 152 lớp (11,3 tỷ FLOP) vẫn
có độ phức tạp thấp hơn lưới VGG-16/19 (15,3/19,6 tỷ FLOP).

ResNets lớp 50/101/152 chính xác hơn
những cái 34 lớp có biên độ đáng kể (Bảng 3 và 4).
Chúng tôi không quan sát thấy vấn đề xuống cấp và do đó đạt được mức
độ chính xác đáng kể từ việc tăng đáng kể
chiều sâu. Lợi ích của chiều sâu được chứng minh cho tất cả các đánh giá
số liệu (Bảng 3 và 4).

So sánh với các phương pháp hiện đại. Trong bảng 4
chúng tôi so sánh với các kết quả mô hình đơn tốt nhất trước đó.
ResNet 34 lớp cơ bản của chúng tôi đã đạt được độ chính xác rất cạnh
tranh. ResNet 152 lớp của chúng tôi có một mô hình duy nhất
lỗi xác thực top 5 là 4,49%. Kết quả mô hình đơn này
vượt trội hơn tất cả các kết quả tổng thể trước đó (Bảng 5). Chúng tôi
kết hợp sáu mô hình có chiều sâu khác nhau để tạo thành một quần thể
(chỉ với hai cái 152 lớp tại thời điểm gửi).
Điều này dẫn đến sai số top 5 là 3,57% trên tập kiểm tra (Bảng 5).
Bài dự thi này đã giành vị trí số 1 trong ILSVRC 2015.

4.2. CIFAR-10 và phân tích

Chúng tôi đã tiến hành nhiều nghiên cứu hơn về bộ dữ liệu CIFAR-10
[20], bao gồm 50k hình ảnh đào tạo và 10k hình ảnh thử nghiệm trong
10 lớp. Chúng tôi trình bày các thí nghiệm đã được huấn luyện
trên tập huấn luyện và được đánh giá trên tập kiểm tra. Trọng tâm của chúng tôi
là về hành vi của các mạng cực kỳ sâu, nhưng không phải về
thúc đẩy các kết quả tiên tiến nhất, vì vậy chúng tôi cố tình sử dụng
kiến trúc đơn giản như sau.

Các kiến trúc đơn giản/dư lượng theo mẫu trong Hình 3
(giữa/phải). Đầu vào mạng là hình ảnh 32×32, với
giá trị trung bình trên mỗi pixel bị trừ. Lớp đầu tiên là lớp chập
3×3. Sau đó, chúng tôi sử dụng một chồng gồm 6n lớp với các tổ hợp 3
× 3 trên bản đồ đặc trưng có kích thước tương ứng là {32, 16, 8},
với 2n lớp cho mỗi kích thước bản đồ đặc trưng. Những con số của
bộ lọc lần lượt là {16, 32, 64} . Việc lấy mẫu con được thực hiện
bằng các phép cuộn với bước tiến là 2. Mạng kết thúc
với tổng hợp trung bình toàn cầu, kết nối đầy đủ 10 chiều
lớp và softmax. Có tổng cộng 6n+2 trọng số xếp chồng lên nhau
các lớp. Bảng sau đây tóm tắt kiến trúc:

kích thước bản đồ đầu ra	32×32	16×16	8×8
# lớp 1+2n 2n	# bộ lọc		2n
	16	32	64

Khi kết nối phím tắt được sử dụng, chúng được kết nối
đến các cặp lớp 3×3 (tổng cộng là 3n phím tắt). Trên này
tập dữ liệu chúng tôi sử dụng các phím tắt nhận dạng trong mọi trường hợp (ví dụ: tùy chọn A),

phương pháp			lỗi (%)
Tối đa [9]			9,38
NIN [25]			8,81
DSN [24]			8,22
	# lớp # thông số		
FitNet [34]	19 2,5 triệu		8,39
Quốc lộ [41, 42]	19	2,3M	7,54 (7,72±0,16)
Quốc lộ [41, 42]	32	1,25M	8,80
ResNet	20	0,27M	8,75
ResNet	32	0,46M	7,51
ResNet	44	0,66M	7,17
ResNet	56	0,85M	6,97
ResNet	110	1,7 triệu	6,43 (6,61±0,16)
ResNet	1202	19,4M	7,93

Bảng 6. Lỗi phân loại trên bộ kiểm tra CIFAR-10. Tất cả các phương pháp đều có
tính năng tăng cường dữ liệu. Đối với ResNet-110, chúng tôi chạy nó 5 lần
và hiển thị "tốt nhất (trung bình±tiêu chuẩn)" như trong [42].

vì vậy các mô hình còn lại của chúng tôi có cùng chiều sâu, chiều rộng,
và số lượng tham số như các đối tác đơn giản.

Chúng tôi sử dụng độ giảm trọng lượng là 0,0001 và động lượng là 0,9,
và áp dụng khởi tạo trọng số trong [12] và BN [16] nhưng
không có tình trạng bỏ học. Các mô hình này được đào tạo với kích
thước lô nhỏ 128 trên hai GPU. Chúng ta bắt đầu với việc học
tỷ lệ 0,1, chia cho 10 ở lần lặp 32k và 48k và
kết thúc đào tạo ở 64k lần lặp, được xác định trên
chia 45k/5k tàu/val. Chúng tôi làm theo cách tăng cường dữ liệu đơn
giản trong [24] để huấn luyện: 4 pixel được đệm ở mỗi bên,
và ảnh cắt 32×32 được lấy mẫu ngẫu nhiên từ phần đệm
hình ảnh hoặc lật ngang của nó. Để thử nghiệm, chúng tôi chỉ đánh giá
chế độ xem duy nhất của hình ảnh 32 × 32 gốc.

Ta so sánh n = {3, 5, 7, 9} dẫn đến 20, 32, 44 và
Mạng 56 lớp. Hình 6 (trái) thể hiện hành vi của
lưới trơn. Các lưới ở vùng đồng bằng sâu phải chịu đựng độ sâu ngày càng tăng,
và thể hiện lỗi đào tạo cao hơn khi đi sâu hơn. Cái này
hiện tượng tương tự như trên ImageNet (Hình 4, bên trái) và
trên MNIST (xem [41]), gợi ý rằng việc tối ưu hóa như vậy
khó khăn là một vấn đề cơ bản.

Hình 6 (giữa) hiển thị hành vi của ResNets. Cũ ng
tương tự như các trường hợp ImageNet (Hình 4, bên phải), ResNets của chúng tôi
quản lý để khắc phục khó khăn tối ưu hóa và thể hiện mức độ chính xác
đạt được khi độ sâu tăng lên.

Chúng tôi khám phá thêm n = 18 dẫn đến 110 lớp
ResNet. Trong trường hợp này, chúng tôi thấy rằng tốc độ học tập ban đầu
0,1 là hơi quá lớn để bắt đầu hội tụ5 . Vì vậy chúng tôi sử dụng
0,01 để khởi động quá trình đào tạo cho đến khi sai số đào tạo ở mức dưới
80% (khoảng 400 lần lặp), sau đó quay lại 0,1 và tiếp tục đào tạo.
Phần còn lại của lịch học như đã hoàn thành
trước đó. Mạng 110 lớp này hội tụ tốt (Hình 6,
ở giữa). Nó có ít thông số hơn các loại sâu và mỏng khác

5Với tốc độ học ban đầu là 0,1, nó bắt đầu hội tụ (lỗi <90%)
sau vài kỷ nguyên nhưng vẫn đạt được độ chính xác tương tự.

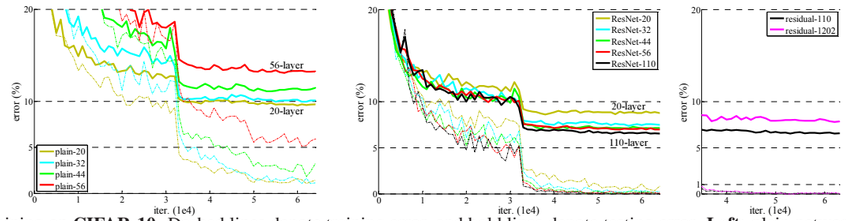


Figure 6. Training on **CIFAR-10**. Dashed lines denote training error, and bold lines denote testing error. **Left**: plain networks. The error of plain-110 is higher than 60% and not displayed. **Middle**: ResNets. **Right**: ResNets with 110 and 1202 layers.

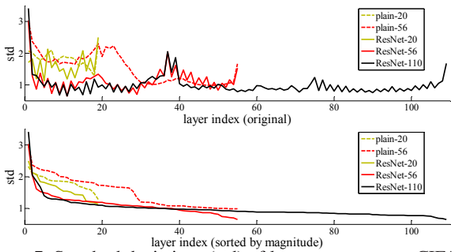


Figure 7. Standard deviations (std) of layer responses on CIFAR-10. The responses are the outputs of each 3×3 layer, after BN and before nonlinearity. **Top**: the layers are shown in their original order. **Bottom**: the responses are ranked in descending order.

networks such as FitNet [34] and Highway [41] (Table 6), yet is among the state-of-the-art results (6.43%, Table 6).

Analysis of Layer Responses. Fig. 7 shows the standard deviations (std) of the layer responses. The responses are the outputs of each 3×3 layer, after BN and before other nonlinearity (ReLU/addition). For ResNets, this analysis reveals the response strength of the residual functions. Fig. 7 shows that ResNets have generally smaller responses than their plain counterparts. These results support our basic motivation (Sec.3.1) that the residual functions might be generally closer to zero than the non-residual functions. We also notice that the deeper ResNet has smaller magnitudes of responses, as evidenced by the comparisons among ResNet-20, 56, and 110 in Fig. 7. When there are more layers, an individual layer of ResNets tends to modify the signal less.

Exploring Over 1000 layers. We explore an aggressively deep model of over 1000 layers. We set $n = 200$ that leads to a 1202-layer network, which is trained as described above. Our method shows *no optimization difficulty*, and this 10^3-layer network is able to achieve *training error* $<0.1\%$ (Fig. 6, right). Its test error is still fairly good (7.93%, Table 6).

But there are still open problems on such aggressively deep models. The testing result of this 1202-layer network is worse than that of our 110-layer network, although both

training data	07+12	07++12
test data	VOC 07 test	VOC 12 test
VGG-16	73.2	70.4
ResNet-101	**76.4**	**73.8**

Table 7. Object detection mAP (%) on the PASCAL VOC 2007/2012 test sets using **baseline** Faster R-CNN. See also appendix for better results.

metric	mAP@.5	mAP@[.5, .95]
VGG-16	41.5	21.2
ResNet-101	**48.4**	**27.2**

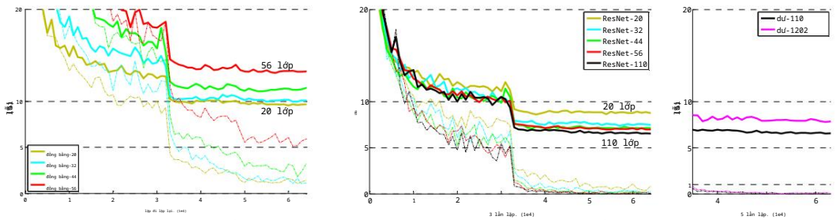
Table 8. Object detection mAP (%) on the COCO validation set using **baseline** Faster R-CNN. See also appendix for better results.

have similar training error. We argue that this is because of overfitting. The 1202-layer network may be unnecessarily large (19.4M) for this small dataset. Strong regularization such as maxout [9] or dropout [13] is applied to obtain the best results ([9, 25, 24, 34]) on this dataset. In this paper, we use no maxout/dropout and just simply impose regularization via deep and thin architectures by design, without distracting from the focus on the difficulties of optimization. But combining with stronger regularization may improve results, which we will study in the future.

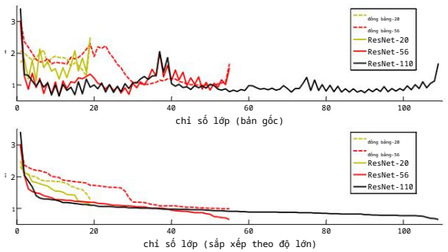
4.3. Object Detection on PASCAL and MS COCO

Our method has good generalization performance on other recognition tasks. Table 7 and 8 show the object detection baseline results on PASCAL VOC 2007 and 2012 [5] and COCO [26]. We adopt *Faster R-CNN* [32] as the detection method. Here we are interested in the improvements of replacing VGG-16 [40] with ResNet-101. The detection implementation (see appendix) of using both models is the same, so the gains can only be attributed to better networks. Most remarkably, on the challenging COCO dataset we obtain a 6.0% increase in COCO's standard metric (mAP@[.5, .95]), which is a 28% relative improvement. This gain is solely due to the learned representations.

Based on deep residual nets, we won the 1st places in several tracks in ILSVRC & COCO 2015 competitions: ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation. The details are in the appendix.



Hình 6. Đào tạo về CIFAR-10. Các đường đứt nét biểu thị lỗi huấn luyện và các đường đậm biểu thị lỗi kiểm tra. Bên trái: mạng đơn giản. Lỗi của plain-110 cao hơn 60% và không được hiển thị. Giữa: ResNets. Phải: ResNets với 110 và 1202 lớp.



Hình 7. Độ lệch chuẩn (std) của phản hồi lớp trên CIFAR-10. Các phản hồi là đầu ra của mỗi lớp 3×3, sau BN và trước sự phi tuyến. Trên cùng: các lớp được hiển thị trong bản gốc của chúng đặt hàng. Dưới cùng: các câu trả lời được xếp theo thứ tự giảm dần.

dữ liệu huấn luyện	07+12	07++12
dữ liệu kiểm tra	Kiểm tra VOC 07	kiểm tra VOC 12
VGG-16	73,2	70,4
ResNet-101	76,4	73,8

Bảng 7. MAP phát hiện đối tượng (%) trên PASCAL VOC Bộ thử nghiệm 2007/2012 sử dụng Faster R-CNN cơ bản. Xem thêm phụ lục để có kết quả tốt hơn.

Hệ mét	mAP@.5	mAP@[.5, .95]
VGG-16	41,5	21.2
ResNet-101	48,4	27,2

Bảng 8. MAP phát hiện đối tượng (%) trên bộ xác thực COCO sử dụng đường cơ sở Faster R-CNN. Xem thêm phụ lục để có kết quả tốt hơn.

các mạng như FitNet [34] và Highway [41] (Bảng 6), nhưng vẫn nằm trong số những kết quả tiên tiến nhất (6,43%, Bảng 6).

Phân tích phản hồi của lớp. Hình 7 thể hiện tiêu chuẩn độ lệch (std) của các phản hồi của lớp. Các câu trả lời là đầu ra của mỗi lớp 3 × 3, sau BN và trước lớp khác tính phi tuyến (ReLU/bổ sung). Đối với ResNets, phân tích này cho thấy cường độ đáp ứng của các hàm dư.
Hình 7 cho thấy ResNets nhìn chung có phản hồi nhỏ hơn hơn so với các đối tác đơn giản của họ. Những kết quả này hỗ trợ động lực cơ bản của chúng tôi (Phần 3.1) rằng các hàm dư có thể thường gần 0 hơn các hàm không dư.
Chúng tôi cũ ng nhận thấy rằng ResNet sâu hơn có mức độ phản hồi nhỏ hơn, bằng chứng là sự so sánh giữa ResNet-20, 56 và 110 trong Hình 7. Khi có nhiều hơn các lớp, một lớp ResNet riêng lẻ có xu hướng sửa đổi tín hiệu ít hơn.

Khám phá hơn 1000 lớp. Chúng tôi khám phá một cách tích cực mô hình sâu hơn 1000 lớp. Ta đặt n = 200 rằng dẫn đến mạng 1202 lớp, được huấn luyện như mô tả bên trên. Phương pháp của chúng tôi không gặp khó khăn gì trong việc tối ưu hóa và mạng 103 lớp này có thể gặp phải lỗi huấn luyện <0,1% (Hình 6, bên phải). Lỗi test của nó vẫn khá tốt (7,93%, Bảng 6).

Nhưng vẫn còn những vấn đề mở về sự tích cực như vậy những mô hình sâu sắc Kết quả thử nghiệm của mạng 1202 lớp này còn tệ hơn mạng 110 lớp của chúng tôi, mặc dù cả hai có lỗi đào tạo tương tự. Chúng tôi cho rằng điều này là do trang bị quá mức. Mạng 1202 lớp có thể không cần thiết lớn (19,4M) cho tập dữ liệu nhỏ này. Chính quy hóa mạnh mẽ chẳng hạn như maxout [9] hoặc dropout [13] được áp dụng để có được kết quả tốt nhất ([9, 25, 24, 34]) trên tập dữ liệu này. Trong bài báo này, chúng tôi không sử dụng maxout/dropout và chỉ đơn giản áp đặt chính quy hóa thông qua các kiến trúc sâu và mỏng theo thiết kế mà không làm xao lãng sự tập trung vào những khó khăn của việc tối ưu hóa. Nhưng kết hợp với việc chính quy hóa mạnh mẽ hơn có thể cải thiện kết quả mà chúng ta sẽ nghiên cứu trong tương lai.

4.3. Phát hiện đối tượng trên PASCAL và MS COCO

Phương pháp của chúng tôi có hiệu suất khái quát hóa tốt trên nhiệm vụ nhận dạng khác. Bảng 7 và 8 thể hiện kết quả cơ bản phát hiện vật thể trên PASCAL VOC 2007 và 2012 [5] và COCO [26]. Chúng tôi áp dụng R-CNN nhanh hơn [32] làm phương pháp khử nhiễu. Ở đây chúng tôi quan tâm đến những cải tiến thay thế VGG-16 [40] bằng ResNet-101. Việc phát hiện việc thực hiện (xem phụ lục) sử dụng cả hai mô hình là giống nhau, vì vậy lợi ích chỉ có thể được quy cho các mạng tốt hơn. Đáng chú ý nhất, trên tập dữ liệu COCO đầy thách thức, chúng tôi nhận được chỉ số tiêu chuẩn của COCO tăng 6,0% (mAP@[.5, .95]), mức cải thiện tương đối là 28%. Lợi ích này là chỉ nhờ vào các biểu diễn đã học.

Dựa trên lưới dư sâu, chúng tôi đã giành được vị trí số 1 trong một số bài hát trong cuộc thi ILSVRC & COCO 2015: Phát hiện ImageNet, Bản địa hóa ImageNet, Phát hiện COCO, và phân đoạn COCO. Chi tiết có trong phần phụ lục.

References

[1] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[2] C. M. Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995.

[3] W. L. Briggs, S. F. McCormick, et al. *A Multigrid Tutorial*. Siam, 2000.

[4] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *BMVC*, 2011.

[5] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *IJCV*, pages 303–338, 2010.

[6] R. Girshick. Fast R-CNN. In *ICCV*, 2015.

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[8] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010.

[9] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio. Maxout networks. *arXiv:1302.4389*, 2013.

[10] K. He and J. Sun. Convolutional neural networks at constrained time cost. In *CVPR*, 2015.

[11] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015.

[13] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv:1207.0580*, 2012.

[14] S. Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. *Diploma thesis, TU Munich*, 1991.

[15] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[16] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

[17] H. Jegou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *TPAMI*, 33, 2011.

[18] H. Jegou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, and C. Schmid. Aggregating local image descriptors into compact codes. *TPAMI*, 2012.

[19] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv:1408.5093*, 2014.

[20] A. Krizhevsky. Learning multiple layers of features from tiny images. *Tech Report*, 2009.

[21] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[22] 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*, 1989.

[23] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In *Neural Networks: Tricks of the Trade*, pages 9–50. Springer, 1998.

[24] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. *arXiv:1409.5185*, 2014.

[25] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv:1312.4400*, 2013.

[26] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *ECCV*. 2014.

[27] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[28] G. Montúfar, R. Pascanu, K. Cho, and Y. Bengio. On the number of linear regions of deep neural networks. In *NIPS*, 2014.

[29] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[30] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, 2007.

[31] T. Raiko, H. Valpola, and Y. LeCun. Deep learning made easier by linear transformations in perceptrons. In *AISTATS*, 2012.

[32] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[33] B. D. Ripley. *Pattern recognition and neural networks*. Cambridge university press, 1996.

[34] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.

[35] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *arXiv:1409.0575*, 2014.

[36] A. M. Saxe, J. L. McClelland, and S. Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv:1312.6120*, 2013.

[37] N. N. Schraudolph. Accelerated gradient descent by factor-centering decomposition. Technical report, 1998.

[38] N. N. Schraudolph. Centering neural network gradient factors. In *Neural Networks: Tricks of the Trade*, pages 207–226. Springer, 1998.

[39] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *ICLR*, 2014.

[40] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[41] R. K. Srivastava, K. Greff, and J. Schmidhuber. Highway networks. *arXiv:1505.00387*, 2015.

[42] R. K. Srivastava, K. Greff, and J. Schmidhuber. Training very deep networks. *1507.06228*, 2015.

[43] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[44] R. Szeliski. Fast surface interpolation using hierarchical basis functions. *TPAMI*, 1990.

[45] R. Szeliski. Locally adapted hierarchical basis preconditioning. In *SIGGRAPH*, 2006.

[46] T. Vatanen, T. Raiko, H. Valpola, and Y. LeCun. Pushing stochastic gradient towards second-order methods–backpropagation learning with transformations in nonlinearities. In *Neural Information Processing*, 2013.

[47] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms, 2008.

[48] W. Venables and B. Ripley. Modern applied statistics with s-plus. 1999.

[49] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional neural networks. In *ECCV*, 2014.

Người giới thiệu

[1] Y. Bengio, P. Simard và P. Frasconi. Việc học các phần phụ thuộc dài hạn có độ dốc giảm dần là điều khó khăn. Giao dịch của IEEE trên Mạng thần kinh, 5(2):157–166, 1994.

[2] Giám mục CM. Các mạng thần kinh để nhận dạng mẫu. Oxford Nhà xuất bản Đại học, 1995.

[3] WL Briggs, SF McCormick, và cộng sự. Hướng dẫn về Multigrid. Xiêm, 2000.

[4] K. Chatfield, V. Lempitsky, A. Vedaldi và A. Zisserman. Điều khó khăn nằm ở chi tiết: đánh giá các phương pháp mã hóa tính năng gần đây. Trong BMVC, 2011.

[5] M. Everingham, L. Van Gool, CK Williams, J. Winn và A. Zis-serman. Thử thách các lớp đối tượng trực quan Pascal (VOC). IJCV, trang 303-338, 2010.

[6] R. Girshick. R-CNN nhanh. Trong ICCV, 2015.

[7] R. Girshick, J. Donahue, T. Darrell và J. Malik. Hệ thống phân cấp tính năng phong phú giúp phát hiện đối tượng chính xác và phân đoạn theo ngữ nghĩa. Trong CVPR, 2014.

[8] X. Glorot và Y. Bengio. Hiểu được khó khăn trong việc đào tạo mạng lưới thần kinh chuyển tiếp sâu. Trong AISTATS, 2010.

[9] IJ Goodfellow, D. Warde-Farley, M. Mirza, A. Courville và Y. Bengio. Mạng tối đa. arXiv:1302.4389, 2013.

[10] K. He và J. Sun. Mạng lưới thần kinh tích chập tại thời điểm hạn chế trị giá. Trong CVPR, 2015.

[11] K. He, X. Zhang, S. Ren và J. Sun. Nhóm kim tự tháp không gian trong các mạng tích chập sâu để nhận dạng trực quan. Trong ECCV, 2014.

[12] K. He, X. Zhang, S. Ren và J. Sun. Đi sâu vào bộ chỉnh lưu: Vượt qua hiệu suất ở cấp độ con người trong phân loại imagenet. Trong ICCV, 2015.

[13] GE Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever và RR Salakhutdinov. Cải thiện mạng lưới thần kinh bằng cách ngăn chặn sự đồng thích ứng của các bộ phát hiện tính năng. arXiv:1207.0580, 2012.

[14] S. Hochreiter. Untersuchungen zu dynamischen nơ-ron thần kinh trên mạng. Luận văn tốt nghiệp, TU Munich, 1991.

[15] S. Hochreiter và J. Schmidhuber. Trí nhớ ngắn hạn dài. thần kinh tính toán, 9(8):1735–1780, 1997.

[16] S. Ioffe và C. Szegedy. Chuẩn hóa hàng loạt: Tăng tốc đào tạo mạng sâu bằng cách giảm sự dịch chuyển hiệp phương sai nội bộ. Trong ICML, 2015.

[17] H. Jegou, M. Douze và C. Schmid. Lượng tử hóa sản phẩm gần nhất tìm kiếm hàng xóm. TPAMI, 33, 2011.

[18] H. Jegou, F. Perronnin, M. Douze, J. Sanchez, P. Perez và C. Schmid. Tổng hợp các bộ mô tả hình ảnh cục bộ thành các mã nhỏ gọn. TPAMI, 2012.

[19] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama và T. Darrell. Caffe: Kiến trúc tích chập để nhúng tính năng nhanh. arXiv:1408.5093, 2014.

[20] A. Krizhevsky. Tìm hiểu nhiều lớp tính năng từ các hình ảnh nhỏ. Báo cáo Công nghệ, 2009.

[21] A. Krizhevsky, I. Sutskever và G. Hinton. Phân loại Imagenet với mạng lưới thần kinh tích chập sâu. Trong NIPS, 2012.

[22] Y. LeCun, B. Boser, JS Denker, D. Henderson, RE Howard, W. Hubbard và LD Jackel. Lan truyền ngược được áp dụng cho nhận dạng mã zip viết tay. Tính toán thần kinh, 1989.

[23] Y. LeCun, L. Bottou, GB Orr và K.-R. Muller. Chống đỡ hiệu quả. Trong Mạng thần kinh: Thủ thuật giao dịch, trang 9-50. Springer, 1998.

[24] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang và Z. Tu. Lưới được giám sát sâu. arXiv:1409.5185, 2014.

[25] M. Lin, Q. Chen và S. Yan. Mạng trong mạng. arXiv:1312.4400, 2013.

[26] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar và CL Zitnick. Microsoft COCO: Các đối tượng phổ biến trong ngữ cảnh. Trong ECCV. 2014.

[27] J. Long, E. Shelhamer và T. Darrell. Mạng tích chập hoàn toàn để phân đoạn ngữ nghĩa. Trong CVPR, 2015.

[28] G. Montufar, R. Pascanu, K. Cho và Y. Bengio. Về số vùng tuyến tính của mạng lưới thần kinh sâu. Trong NIPS, 2014.

[29] V. Nair và GE Hinton. Đơn vị tuyến tính được chỉnh sửa cải thiện bị hạn chế máy Boltzmann. Trong ICML, 2010.

[30] F. Perronnin và C. Dance. Hạt Fisher về từ vựng trực quan cho phân loại hình ảnh. Trong CVPR, 2007.

[31] T. Raiko, H. Valpola và Y. LeCun. Học sâu trở nên dễ dàng hơn nhờ các phép biến đổi tuyến tính trong perceptron. Trong AISTATS, 2012.

[32] S. Ren, K. He, R. Girshick và J. Sun. R-CNN nhanh hơn: Hướng tới phát hiện đối tượng theo thời gian thực với các mạng đề xuất khu vực. Trong NIPS, 2015.

[33] BD Ripley. Nhận dạng mẫu và mạng lưới thần kinh. Cambridge Nhà xuất bản Đại học, 1996.

[34] A. Romero, N. Ballas, SE Kahou, A. Chassang, C. Gatta và Y. Bengio. Fitnets: Gợi ý cho lưới mỏng sâu. Trong ICLR, 2015.

[35] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, và những người khác. Thử thách nhận dạng hình ảnh quy mô lớn của Imagenet. arXiv:1409.0575, 2014.

[36] AM Saxe, JL McClelland và S. Ganguli. Các giải pháp chính xác cho động lực phi tuyến của việc học trong mạng lưới thần kinh tuyến tính sâu. arXiv:1312.6120, 2013.

[37] NN Schraudolph. Giảm độ dốc được tăng tốc bằng cách tập trung vào yếu tố sự phân hủy. Báo cáo kỹ thuật, 1998.

[38] NN Schraudolph. Tập trung các yếu tố độ dốc mạng lưới thần kinh. Trong Mạng thần kinh: Thủ thuật giao dịch, trang 207-226. Springer, 1998.

[39] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, và Y. Le-Cun. Overfeat: Tích hợp nhận dạng, bản địa hóa và phát hiện bằng cách sử dụng mạng tích chập. Trong ICLR, 2014.

[40] K. Simonyan và A. Zisserman. Mạng tích chập rất sâu để nhận dạng hình ảnh quy mô lớn. Trong ICLR, 2015.

[41] RK Srivastava, K. Greff và J. Schmidhuber. Mạng lưới đường cao tốc. arXiv:1505.00387, 2015.

[42] RK Srivastava, K. Greff và J. Schmidhuber. Đào tạo rất sâu mạng. 1507.06228, 2015.

[43] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Er-han, V. Vanhoucke và A. Rabinovich. Đi sâu hơn với sự phức tạp. Trong CVPR, 2015.

[44] R. Szeliski. Nội suy bề mặt nhanh bằng cách sử dụng hàm cơ sở phân cấp ý kiến. TPAMI, 1990.

[45] R. Szeliski. Điều kiện tiên quyết về cơ sở phân cấp được điều chỉnh theo địa phương. Trong SIGGRAPH, 2006.

[46] T. Vatanen, T. Raiko, H. Valpola, và Y. LeCun. Đẩy gradient ngẫu nhiên sang các phương pháp bậc hai – học lan truyền ngược với các phép biến đổi trong phi tuyến. Trong Xử lý thông tin thần kinh, 2013.

[47] A. Vedaldi và B. Fulkerson. VLFeat: Thư viện mở và di động của thuật toán thị giác máy tính, 2008.

[48] W. Venables và B. Ripley. Thống kê ứng dụng hiện đại với s-plus. 1999.

[49] MD Zeiler và R. Fergus. Hình dung và hiểu sự phức tạp mạng lưới thần kinh tional. Trong ECCV, 2014.