ResNet Paper

**Is always on increasing depth in deep network results in better learning ?**

**A close-up of a text

AI-generated content may be incorrect.**

**NO,** because of

Vanishing/Exploding Gradients Problem:

* + Deeper networks often suffer from vanishing or exploding gradients, hindering effective training.
  + These gradient issues make it difficult for deep networks to converge from the beginning.
* Solutions :
  + Normalized Initialization:
    - Proper initialization helps stabilize gradients, mitigating vanishing/exploding problems.
  + Intermediate Normalization Layers (e.g., Batch Normalization):
    - Improves training stability and enables deep networks to converge.
* Benefits of this solution**:**
  + These techniques allow networks with tens of layers to converge using Stochastic GradientDescent (SGD) with backpropagation.

**Stochastic gradient descent (SGD)**

Stochastic Gradient Descent (SGD) speeds up computation by using a single random data point or a small batch to compute gradients at each step instead of the full dataset, which is slow and memory-intensive for large datasets.

**Another problem was found**

**A close-up of a text

AI-generated content may be incorrect.**

As deeper neural networks start converging,

a degradation problem emerges where accuracy first saturates and then rapidly declines.

This issue is not due to overfitting, but rather adding more layers increases training error.

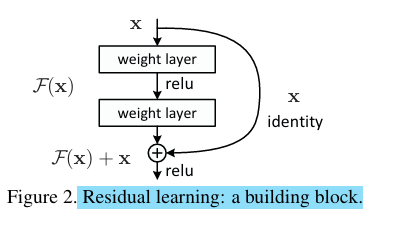
A graph of a test error

AI-generated content may be incorrect.

Residual Learning Framework:

To address the degradation problem, the authors propose a deep residual learning framework, where the network is structured to learn residual mappings instead of directly learning the desired function.

* Let H(x) be the underlying desired mapping. The network is designed to learn F(x)=H(x)−x, so the original mapping is expressed as F(x)+x.
* This residual formulation makes it easier for the network to optimize because it is simpler to push residuals to zero than to fit identity mappings through deep nonlinear layers.



**Identity Shortcut Connections:**

The framework uses shortcut connections that skip one or more layers and perform identity mapping . These shortcuts:

* Add no extra parameters or computational complexity.
* Enable efficient training with stochastic gradient descent (SGD) and backpropagation.

On evaluation of the method they show that:

* Residual network is easy to optimize
* Exhibits low training error and high accuracy gain with increase in depth

Also when we use identity short connections we can make sure that the important information is not lost as it passes through many layer and also it avoids overfitting.

**Highway network:**

Similar to the identity short connection highway network presents a gating function (function to choose input) , These gates are data dependent and adds parameters.

When the gate is closed (approaches zero) then that layer represents non-residual function.

It has not demonstrated any significant accuracy gain in extremely deep networks.

It’s output equation is as follows:

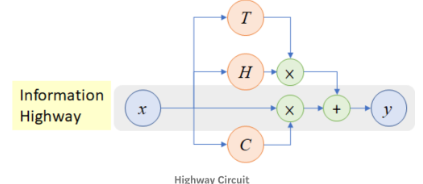
Y = H{x,Wh} . T{x,Wt} + x . C{x,Wc}

Here T : Non-linear transformation

H : Transform gate (decides how much of H(x) to pass)

C : Carry gate (decides how much of x to carry forward)

Generally , H + C = 1



**Importance:**

1. As training progresses, the network learns which layers are important.
2. The carry gate can skip many layers, preventing the network from getting stuck in bad local minima or suffering from vanishing gradients.

**Identity mapping by short connection :**

Let x and F{x,{Wi}} (represents the residual mapping to be learned)

If they are of same dimension then output vector is

A blue rectangular sign with black text

AI-generated content may be incorrect.

If not then



It Ws can also be a square matrix but identity mapping is sufficient and economical

**Network Architecture:**

**A text on a page

AI-generated content may be incorrect.**

**The plain network** : baseline is inspired by VGG nets, using 3x3 convolutional filters with two key design rules:

* maintaining the same number of filters for the same output feature map size.
* doubling the number of filters when halving the feature map size to preserve time complexity.

Downsampling(process of reducing the **spatial dimensions** (height and width) of feature maps in neural networks while preserving essential information) uses convolutional layers with a stride of 2, ending with global average pooling and a 1000-way fully connected softmax layer. With 34 weighted layers, the model has fewer filters and lower complexity than VGG nets, using only 18% of the FLOPs of VGG-19.

A text on a page

AI-generated content may be incorrect.

**Residual networks (ResNets)** : enhance plain networks by adding shortcut connections, turning them into residual versions. When dimensions increase, two methods are used:

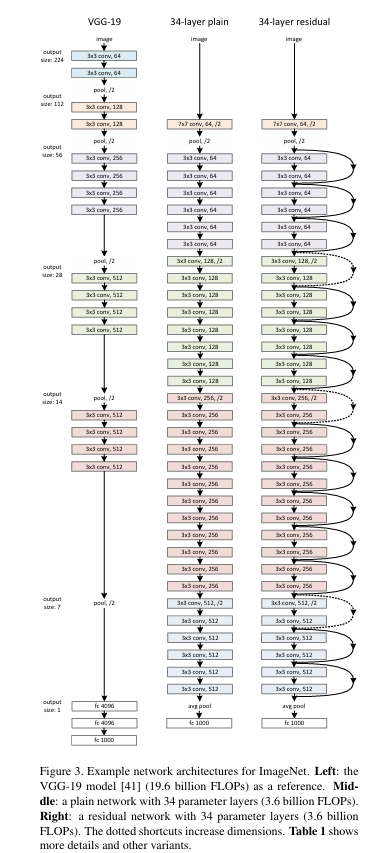
* identity mapping with zero padding for dimension expansion, requiring no extra parameters
* projection n shortcuts using 1x1 convolutions to match dimensions. For both methods, shortcuts with a stride of 2 handle feature maps of different sizes.

**Implementation:**

* Image Preprocessing:
  + - Shorter side randomly resized between [256, 480] for scale augmentation.
    - 224×224 random crop applied with horizontal flipping and per-pixel mean subtraction.
    - Standard color augmentation is used.
* Batch Normalization (BN):
  + - Applied right after each convolution and before activation.
* Training Configuration:
  + SGD optimizer with a mini-batch size of 256.
  + Learning rate starts at 0.1 and is divided by 10 when the error plateaus.
  + Training continues for 60,000 iterations.
  + Weight decay: 0.0001.
  + Momentum: 0.9.
  + No dropout is used.

**Why do we need image augmentation**

* Prevent Overfitting:
  + Deep networks having millions of parameters, making them prone to memorizing the training data.
  + Image augmentation artificially increases the size and diversity of the dataset, making it harder for the network to overfit.
* Simulate Real-World Variability:
  + In real-world scenarios, images vary due to changes in scale , rotation , lighting , position
* Augmentation helps the model learn to be invariant to these changes.
* Effective Regularization:
  + Augmentation is a form of implicit regularization because it forces the model to rely on more generalizable features rather than memorizing specific details.
* Without augmentation, deep networks like ResNet are likely to overfit due to their depth.



**Identity vs Projection shortcuts.**

* Three Shortcut Options for Increasing Dimensions:

1. Zero-padding shortcuts with parameter-free identity shortcuts.
2. Projection shortcuts for increasing dimensions, with other shortcuts as identity.
3. All shortcuts are projections.

* Performance comparison

1. B is slightly better than A due to zero-padded dimensions in A having no residual learning.
2. C is marginally better than B, attributed to the extra parameters introduced.

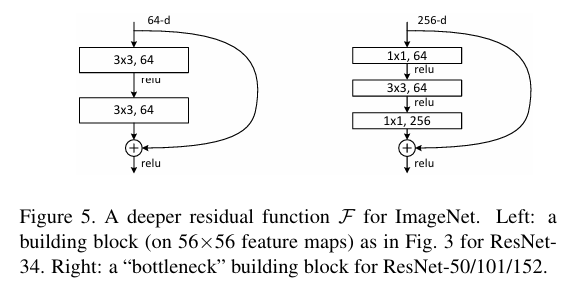
A close-up of a number

AI-generated content may be incorrect.

**Bottleneck Architecture :**

**Purpose of the Bottleneck Design:**

* + The goal is to build deeper networks while managing the training time within feasible limits.
  + To achieve this, the standard residual block is modified into a bottleneck design.
  + This approach allows for efficient computation without compromising model depth.
* **Structure of the Bottleneck Block:**
  + Instead of the typical two-layer residual block, the bottleneck block uses a three-layer stack .



* + The three layers consist of:
    1. 1x1 Convolution (Reduce Dimensions):
       - Compresses the input by reducing its dimensionality.
    2. 3x3 Convolution (Bottleneck):
       - Performs standard convolution on a smaller dimension.
    3. 1x1 Convolution (Restore Dimensions):
       - Expands the dimension back to its original size.
  + This design is efficient because the main convolutional operation (3x3) works on smaller input/output dimensions.
* **Advantages of the Bottleneck Design:**
  + Time Complexity:
    1. Despite having three layers, the time complexity remains comparable to traditional two-layer designs.
  + Efficient Use of Parameters:
    1. Compressing and restoring dimensions helps reduce computational cost.
* **Role of Identity Shortcuts:**
  + Identity shortcuts are particularly important in bottleneck architectures.
  + If these identity shortcuts are replaced with projection shortcuts, it significantly increases complexity.
    1. Time Complexity and Model Size Double:
       - This happens because projection shortcuts connect two high-dimensional ends.
  + Identity shortcuts provide a more efficient and lightweight solution.

**Layer response analysis :**

**A text on a page

AI-generated content may be incorrect.**

**What is a response and small response?**

In neural networks the response refers to the magnitude of the activation value of the each output layer.

More specifically , a small response means that each layer makes small adjustments in the input signal.

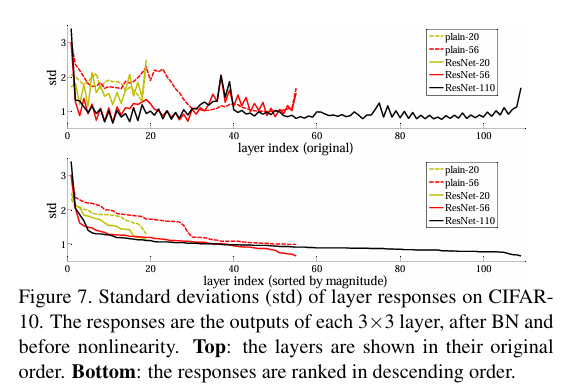
**Why residual network have small response?**

It is because the residual network is designed to learn the difference between input and output.

Y = F{x} + x

Since F{x} focus on learning small adjustment and is often closer to zero .

Thus , the total work is distributed among many layers and hence the output is closer to the identity mapping and also this is the reason because deeper network tend to show even smaller response than there shallow network.



**Exploring over 1000 layers: (110 layer network vs 1202 layer network)**

* Although the 1202-layer network has more layers and parameters, its testing performance is worse than the 110-layer network.
* Interestingly, both networks have similar training errors, meaning they perform equally well on the training set.
* The decline in testing performance suggests that the deeper network is likely overfitting, which means:
  + It learns training data too well, including noise and irrelevant details.
  + It fails to generalize effectively to new, unseen data.

**Why is it overfitting ?**

* Excessive Parameters for a Small Dataset.
* Lack of Strong Regularization
  + Regularization techniques such as:

1. **Maxout**: A method that selects the maximum activation across multiple neurons, reducing overfitting.
2. **Dropout**: Randomly turning off neurons during training to improve generalization.

These techniques were not applied in this paper, focusing instead on the design of deep, thin architectures to explore optimization difficulties.

A graph of error

AI-generated content may be incorrect.

**Comparison of different layered network of plain and residual**

