ResNet for Doodle Recognition: Understanding Deep Residual Networks

Doodle Recognition Project

November 30, 2025

Contents

1 Introduction

This document explains how Machine Learning models, specifically Residual Networks (ResNet), work and how they are applied to recognize doodles from the QuickDraw dataset. The implementation uses ResNet18 with transfer learning, achieving high accuracy in classifying hand-drawn sketches across 340 different categories.

2 Fundamentals of Doodle Recognition

Before diving into ResNet, it is essential to understand how a machine "sees" and recognizes a doodle.

2.1 Images as Numbers

To a computer, a doodle is not a sketch but a grid of numbers (pixels). A 160×160 pixel grayscale image is represented as a matrix of values ranging from 0 (black) to 255 (white).

- **Input**: 160×160 matrix of integers.
- **Goal**: Map this matrix to one of 340 categories (e.g., "cat", "airplane").

2.2 Feature Extraction

A machine learning model does not look at the image holistically like a human. Instead, it extracts **features**:

1. **Low-level features**: Detecting horizontal lines, vertical lines, and curves.
2. **Mid-level features**: Combining lines to form shapes (circles, squares).
3. **High-level features**: Assembling shapes into objects (ears, wheels).

2.3 Classification

The model assigns a probability to each category based on the extracted features. For example, if the model detects "pointed ears" and "whiskers", it increases the probability for "cat".

3 What is ResNet?

ResNet (Residual Network) is a deep convolutional neural network architecture introduced by He et al. in 2015. The key innovation of ResNet is the introduction of **residual connections** (also called skip connections), which allow the network to learn identity mappings and solve the problem of vanishing gradients in very deep networks.

3.1 The Problem with Deep Networks

Traditional deep neural networks face a fundamental problem: as networks get deeper, training becomes more difficult. This is due to:

- **Vanishing Gradients**: Gradients become exponentially small as they propagate backward through many layers
- **Degradation Problem**: Deeper networks often perform worse than shallower ones, even when the deeper network can represent the same functions

3.2 The ResNet Solution: Residual Connections

ResNet solves these problems by introducing residual blocks. Instead of learning the desired mapping $H(x)$ directly, ResNet learns the **residual** (difference) $F(x) = H(x) - x$. The output is then computed as:

$$H(x) = F(x) + x \tag{1}$$

This simple addition allows the network to:

- Learn identity mappings easily (by setting $F(x) = 0$)
- Preserve gradients during backpropagation
- Train very deep networks (ResNet can have 50, 101, or even 152 layers)

4 ResNet Architecture

4.1 Residual Block Structure

A residual block consists of two main paths:

1. **Main Path**: Convolutional layers that learn the residual $F(x)$
2. **Skip Connection**: Direct connection that adds the input x to the output

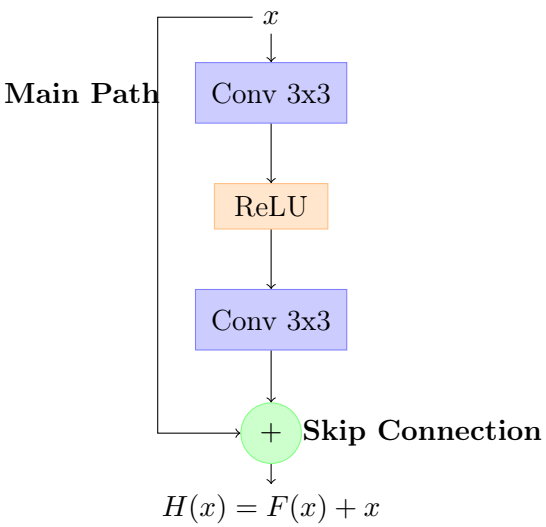

Figure 1: Basic Residual Block Structure

4.2 ResNet18 Architecture

ResNet18 is a variant with 18 layers. The architecture used in our doodle recognition model consists of:

- **Initial Convolution**: 7x7 conv, stride 2 (reduces spatial dimensions)
- **Max Pooling**: 3x3, stride 2
- **4 Residual Blocks**: Each containing 2 convolutional layers

  - Block 1: 64 channels, 2 layers
  - Block 2: 128 channels, 2 layers
  - Block 3: 256 channels, 2 layers
  - Block 4: 512 channels, 2 layers
- **Global Average Pooling**: Reduces spatial dimensions to 1x1
- **Fully Connected Layer**: Maps to number of classes

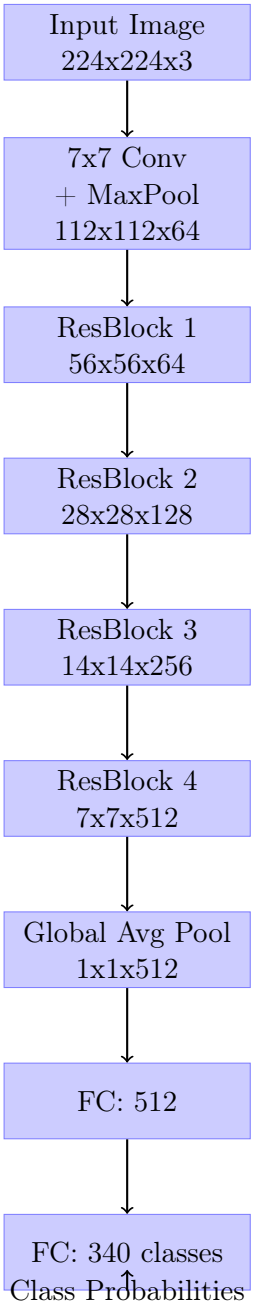

Figure 2: ResNet18 Architecture Overview

5 How ResNet Recognizes Doodles

5.1 Feature Hierarchy

ResNet learns a hierarchical representation of features:

1. **Early Layers**: Detect low-level features (edges, curves, basic shapes)
2. **Middle Layers**: Combine features into more complex patterns (shapes, contours)
3. **Late Layers**: Recognize high-level concepts (objects, categories)

For doodle recognition, this means:

- Layer 1-2: Detects strokes, lines, curves
- Layer 3-4: Recognizes shapes (circles, rectangles, triangles)
- Layer 5-6: Identifies object parts (wheels, wings, faces)
- Layer 7-8: Classifies complete objects (cat, airplane, car)

5.2 Transfer Learning Approach

Our implementation uses **transfer learning** with ImageNet pre-trained weights:

1. **Pre-trained Backbone**: ResNet18 trained on ImageNet (1.2M images, 1000 classes)
2. **Custom Classifier**: Replace final layer for 340 doodle categories
3. **Two-Stage Training**:
   - Stage 1: Freeze backbone, train only classifier (10 epochs)
   - Stage 2: Unfreeze backbone, fine-tune everything (30 epochs)

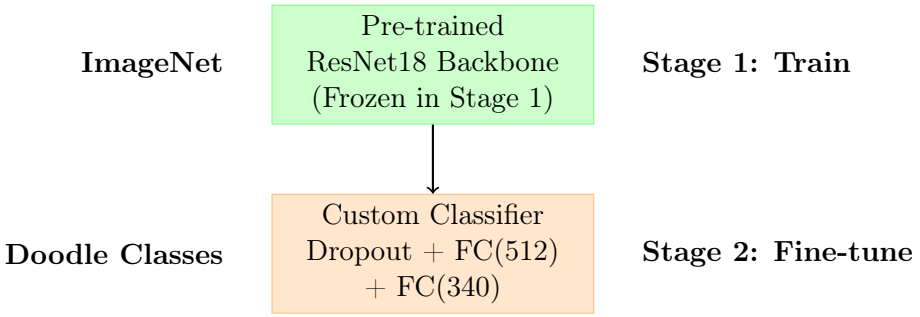

Figure 3: Transfer Learning Architecture

5.3 Why Transfer Learning Works

Transfer learning is effective for doodle recognition because:

- **Feature Reusability**: Low-level features (edges, textures) are similar across image types
- **Data Efficiency**: Requires less training data than training from scratch
- **Faster Training**: Pre-trained weights provide good initialization
- **Better Generalization**: Leverages knowledge from diverse ImageNet dataset

6 Visual Example: Predicting a "Cat" Doodle

Let's trace the prediction process for a real example from the dataset: `cat/4503626191994880.png`.

6.1 Step 1: Input Image

Below is the actual doodle image from the QuickDraw dataset that we will analyze:

Figure 4: Input: A cat doodle from the QuickDraw dataset (`cat/4503626191994880.png`)

To the model, this image is a 28×28 grid of pixel values (0-255), which is resized to 160×160 before being fed into ResNet18.

6.2 Step 2: Convolution - How Kernels Detect Features

The first layers of ResNet apply **convolutional filters** (kernels) to detect basic features like edges. Here's how a vertical edge detector works:

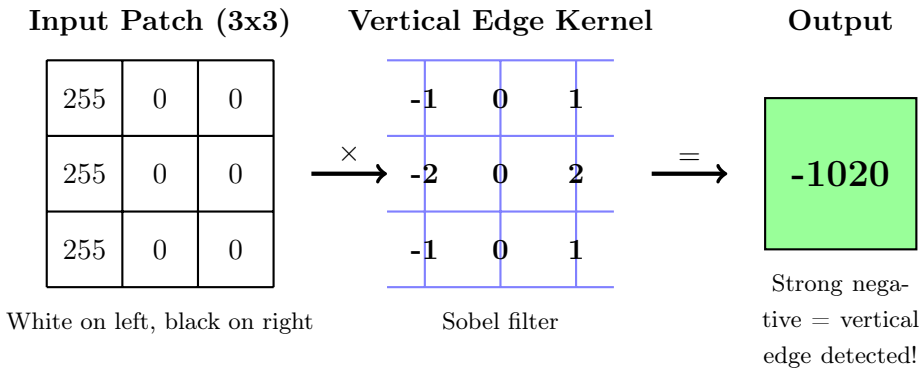

Figure 5: Convolution operation: The kernel slides over the image and produces high activations where edges are detected.

Calculation: $(255 \times -1) + (255 \times -2) + (255 \times -1) + (0 \times 1) + (0 \times 2) + (0 \times 1) = -1020$
A large magnitude indicates a strong edge was detected at this location.

6.3 Step 3: Feature Hierarchy Through Layers

As the image passes through ResNet's layers, simple features combine into complex ones:

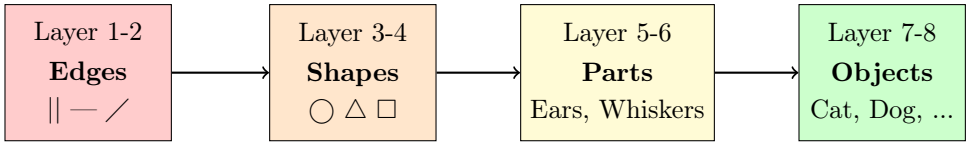

Figure 6: Feature hierarchy: Early layers detect edges, later layers recognize complete objects.

For our cat doodle:

- **Layer 1-2**: Detects vertical lines (ears), curved lines (head outline), horizontal lines (whiskers).
- **Layer 3-4**: Recognizes triangular shapes (pointed ears), circular shape (head).
- **Layer 5-6**: Identifies cat-specific features: "pointed ears" + "whiskers" + "round face".
- **Layer 7-8**: Combines all features and outputs class probabilities.

6.4 Step 4: Final Prediction

The model outputs a probability distribution over all 340 classes:

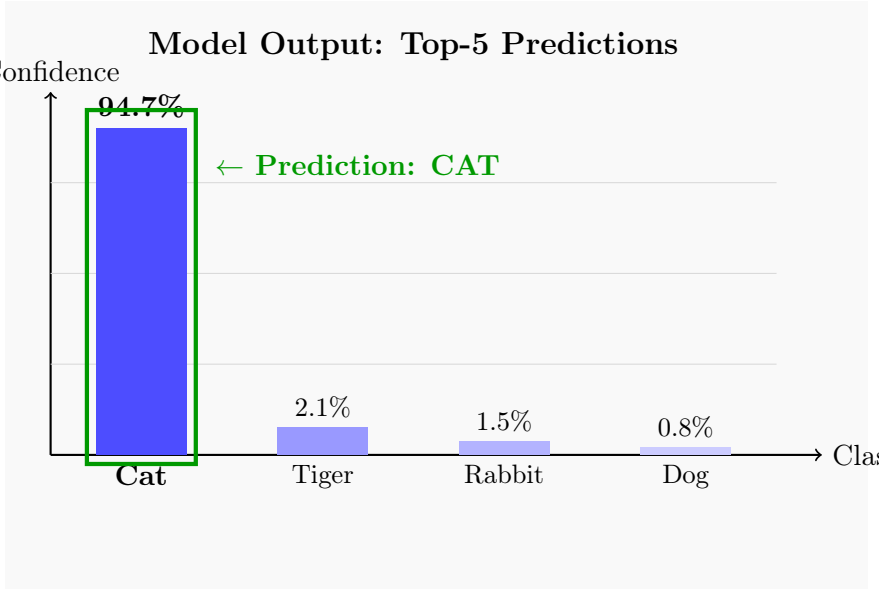

Figure 7: Final prediction probabilities for the cat doodle. The model correctly classifies it as "Cat" with 94.7% confidence.

6.5 Another Example: Airplane

Figure 8: Input: An airplane doodle from the QuickDraw dataset

For the airplane, ResNet detects:

- **Edges**: Long horizontal lines (wings), vertical line (fuselage)
- **Shapes**: Elongated body, triangular tail/nose
- **Parts**: Wing configuration, tail fin, cockpit area
- **Result**: Classified as "Airplane" with high confidence

7 Implementation Details

7.1 Model Configuration

The ResNet18 model used in the notebook has the following configuration:

- **Input Size**: 160x160 pixels (resized from original doodles)
- **Pre-trained Weights**: ImageNet1K_V1
- **Custom Classifier**:
  - Dropout (0.5)
  - Linear: 512 features
  - BatchNorm + ReLU
  - Dropout (0.3)
  - Linear: 340 classes (number of doodle categories)
- **Total Parameters**: 11.6M (11,614,612)
- **Trainable Parameters**: 438K (438,100) in Stage 1

7.2 Training Strategy

The two-stage training approach:

1. **Stage 1** (Epochs 1-10):
   - Freeze ResNet18 backbone
   - Train only custom classifier

- Learning rate: 1e-3
- Purpose: Stabilize classifier with pre-trained features

2. **Stage 2** (Epochs 11-40):
   - Unfreeze all parameters
   - Fine-tune entire network
   - Learning rates: Backbone 1e-4, Classifier 5e-4
   - Purpose: Adapt features to doodle-specific patterns

7.3 Data Augmentation

To improve generalization, the following augmentations are applied:

- Random crop (160x160)
- Random horizontal flip (50% probability)
- Random rotation (±15 degrees)
- Random affine transformation (translation, shear, scale)
- Color jitter (brightness, contrast, saturation, hue)
- ImageNet normalization

8 Results

The ResNet18 model achieves:

- **Test Accuracy**: 75.03%
- **Top-3 Accuracy**: 90.70%
- **Training Samples**: 238,000
- **Validation Samples**: 51,000
- **Test Samples**: 51,000
- **Number of Classes**: 340

9 Why Residual Connections Matter

Residual connections enable ResNet to:

1. **Learn Identity Mappings**: If optimal mapping is identity, $F(x) = 0$ is easier to learn than $H(x) = x$
2. **Preserve Gradients**: Skip connections provide direct gradient paths
3. **Enable Deep Networks**: Can train networks with 50+ layers effectively
4. **Improve Accuracy**: Deeper networks can learn more complex features

Mathematically, the gradient flows through both paths:

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial H} \cdot \left(1 + \frac{\partial F}{\partial x}\right) \tag{2}$$

Even if $\frac{\partial F}{\partial x}$ is small, the gradient can still flow through the identity connection.

10 Conclusion

ResNet's residual connections solve the vanishing gradient problem, enabling training of very deep networks. When combined with transfer learning, ResNet18 provides an effective solution for doodle recognition, achieving high accuracy on the QuickDraw dataset with relatively few trainable parameters. The hierarchical feature learning and skip connections make ResNet particularly well-suited for recognizing the diverse, hand-drawn sketches in the dataset.

11 References

- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. CVPR.
- QuickDraw Dataset: `https://github.com/googlecreativelab/quickdraw-dataset`
- PyTorch ResNet Implementation: `https://pytorch.org/vision/stable/models.html`