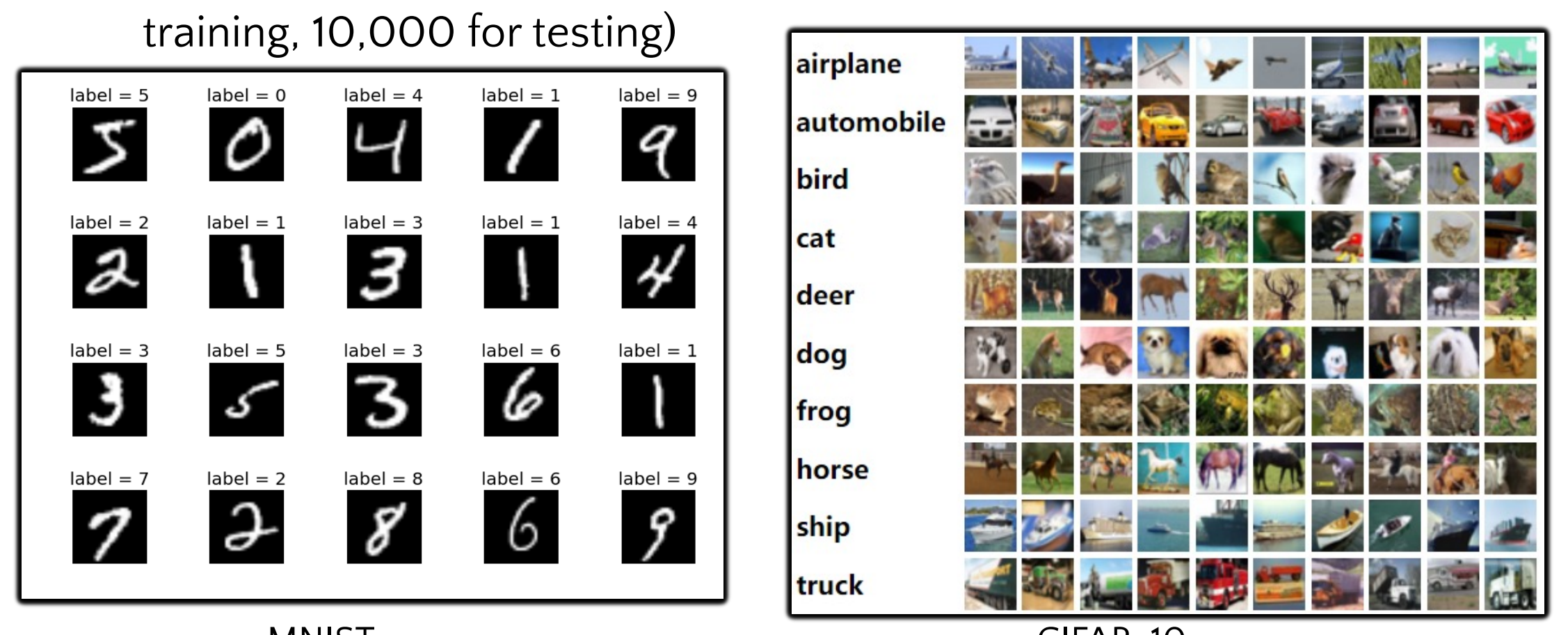
Understanding Neural Networks: Insights into Fully Connected, CNN, and ResNet Models

Sulabh Patel[1], Nikhil Thomas[1], Gagandeep Singh[2]
Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL[1]
School of Computing and Data Science, University of Illinois at Urbana-Champaign, Urbana, IL[2]

Introduction

- Neural Networks are machine learning models for data processing and decision-making
- In Feedforward Neural Networks (FNNs), information moves only from inputs through hidden layers to outputs
- Examples of FNNs include Fully Connected Networks (FCN), Convolutional Neural Networks (CNN), and Residual Networks (ResNet)
- Two datasets were used in our study:
  - MNIST: Contains 70,000 hand-drawn digit images (60,000 for training, 10,000 for testing)
  - CIFAR-10: Contains 60,000 images of animals and vehicles (50,000 for training, 10,000 for testing)



MNIST CIFAR-10

Figure 1

The Perceptron/Neuron

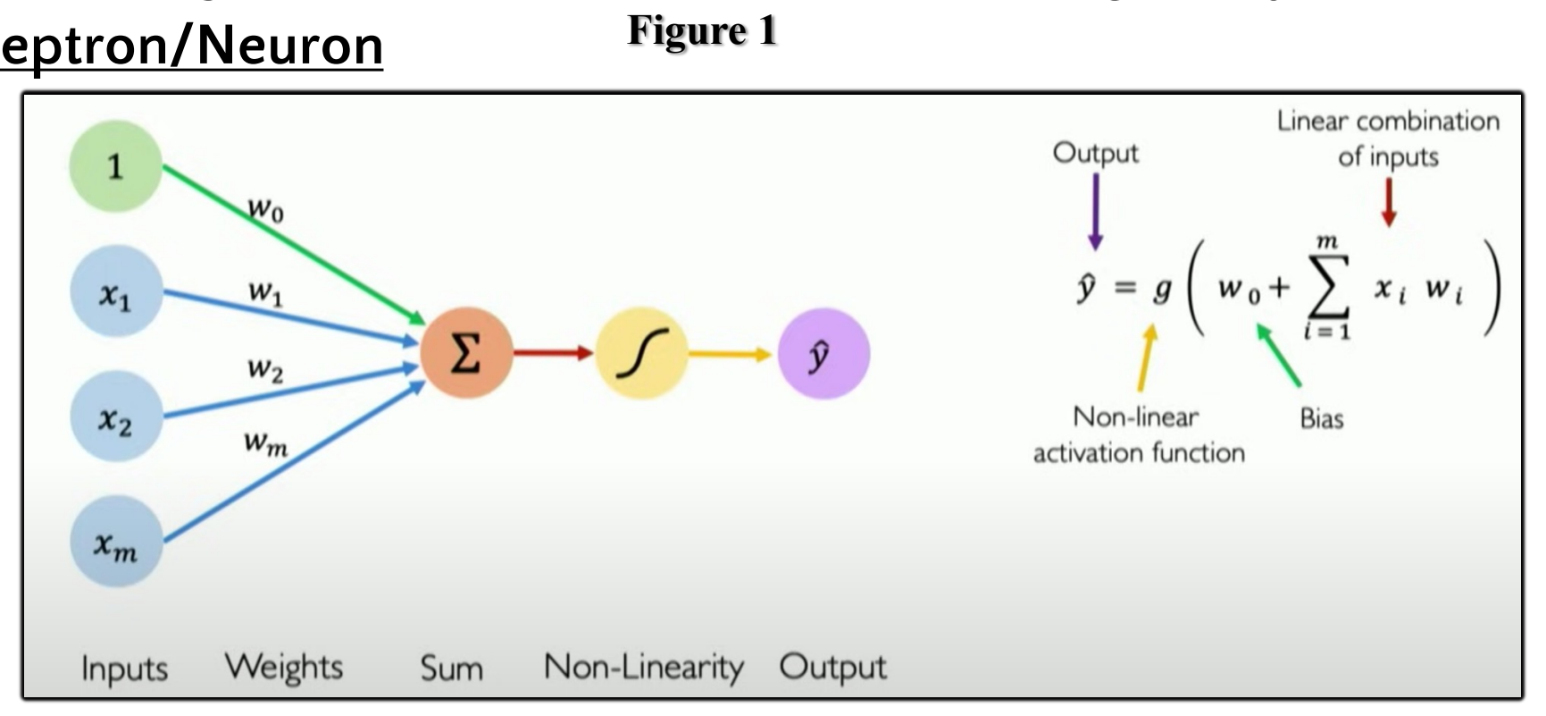

1
x_1
x_2
x_m
w_0
w_1
w_2
w_m
Σ
$\hat{y}$
Output
Linear combination of inputs
$\hat{y} = g\left(w_0 + \sum_{i=1}^{m} x_i w_i\right)$
Non-linear activation function
Bias
Inputs
Weights
Sum
Non-Linearity
Output

Figure 2

In a neuron, inputs are the outputs from neurons in the previous layer, each multiplied by a weight. These weighted inputs are summed and then a bias—a constant specific to each neuron—is added. The resulting sum is typically passed through an **activation function** to determine the neuron's final output.

Backpropagation

Backpropagation enables a neural network to learn by adjusting its parameters through two main steps. During the forward pass, inputs are processed through multiple layers to generate an output and computes the loss using a **loss function**. For our models, it is $-\log(p(y))$ where $p(y)$ is the predicted probability for the true class. Next, the **gradient** is found by calculating the derivative of the loss function with respective to each weight and bias. **Gradients** are calculated backwards from the output to the input layer using the chain rule. To minimize the loss, weights and biases are updated in the direction opposite to the gradient, using the formula $\mathrm{w} := w - n * \frac{\partial L}{\partial w}$, where n is the learning rate that determines the step size for adjustments. Due to the high computational cost of processing each training image individually, mini-batch gradient descent is used. This splits the training set into smaller batches, where the average loss of the whole batch is calculated and then used to execute backpropagation. This allows for more efficient and accelerated training process.

Common Training Issues

- **Overfitting:** A model learns the training data too well, including the noise and outliers. This leads to poor generalization and accuracy on unseen data.
- **Vanishing Gradients:** The gradients of neurons becomes so small that the model can no longer take big enough steps towards convergence, typically seen in the sigmoid and tanh activation functions. This hinders learning.
- **Dying ReLU Problem:** Many neurons become zero due to having negative values. This effectively kills the gradient of any neuron passing through said neuron. These inactive neurons cease to learn and limit the model learning.
- **Exploding Gradients:** The gradients of neurons become so large that the steps taken towards convergence are too large for a model to find convergence. This results in big oscillation about the point for the best model, resulting in inaccurate outputs.
- **Insufficient Data:** When the dataset is too small, the model may not have enough examples to learn the underlying patterns. This can cause poor generalization and stunts the model from accurately predicting new data points.

Essential Techniques

- **Normalization**: Scaling the features of the dataset to a standard range of mean of 0 and standard deviation of 1 to improve performance and convergence.
- **Activation Functions**: Functions applied to neuron outputs to introduce non-linearity into the model. This helps with more complex data Examples include ReLU, Sigmoid, and tanh.
- **Batch Normalization**: A technique that normalizes the inputs of each layer in a batch to have a mean of zero and a deviation of one, helping to stabilize and accelerate training.
- **Dropout**: A regularization method that randomly "drops out" a fraction of neurons during training to prevent overfitting by ensuring the model does not rely too heavily on any single neuron.

Fully Connected Neural Network

Model of a Fully Connected Linear Network

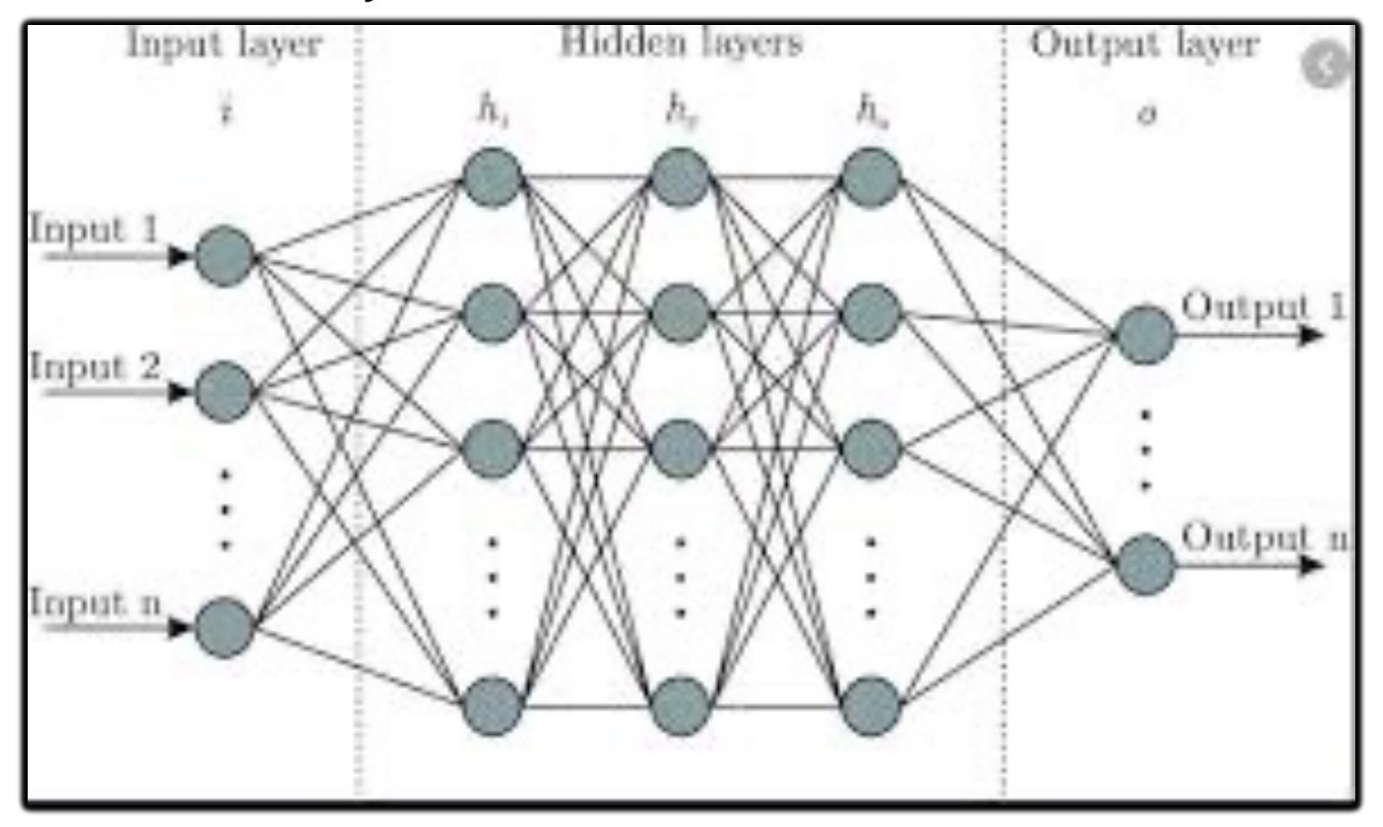


Figure 3

In a Fully Connected Network (FCN), every neuron in a layer is connected to every neuron in the preceding and following layers. The number of neurons in the input layer corresponds to the shape of the data. This is determined by the image dimensions for our datasets. The MNIST has images in grayscale with shape 28x28x1 and the CIFAR-10 has color images with shape 32x32x3. MNIST images are single-channel using only the pixel intensity (0-255), whereas CIFAR-10 images have three channels for red, green, and blue intensities. Each model has a single output layer with 10 neurons, corresponding to the 10 classes in each dataset. The number of hidden layers and neurons within them can be adjusted as needed.

Convolutional Neural Network

Model of a Convolutional Neural Network

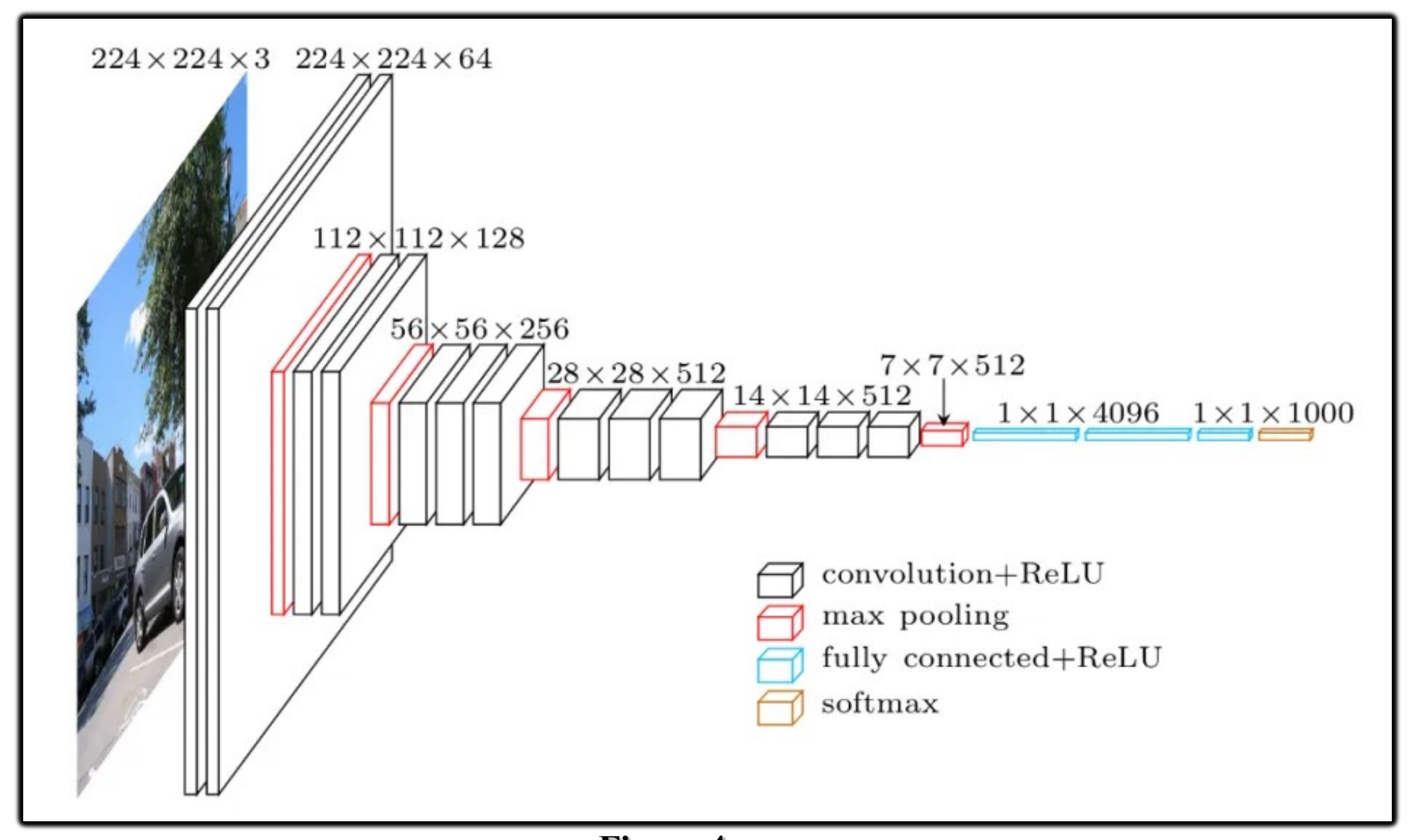


Figure 4

Convolutional neural networks (CNN) function by picking out key features in an image, which help to identify what the input is meant to represent.
It accomplishes this by passing the image through layers that are made up of many filters. The number of filters increase as you go through more layers to capture finer details. The output of each layer also downsizes the dimensions of the image by half, promoting generalization and speeding up calculations. The first layer is meant to pick out obvious, more noticeable features. Then, the outputs from that layer are passed through the next. The model can distinguish some subtle features now. This process continues for a total of 6 layers. At the end, the output of the last layer will be flattened into a single layer. This layer will be fully connected to the last output layer, where the image classification decision is made.

Residual Neural Network

Model of a Residual Neural Network

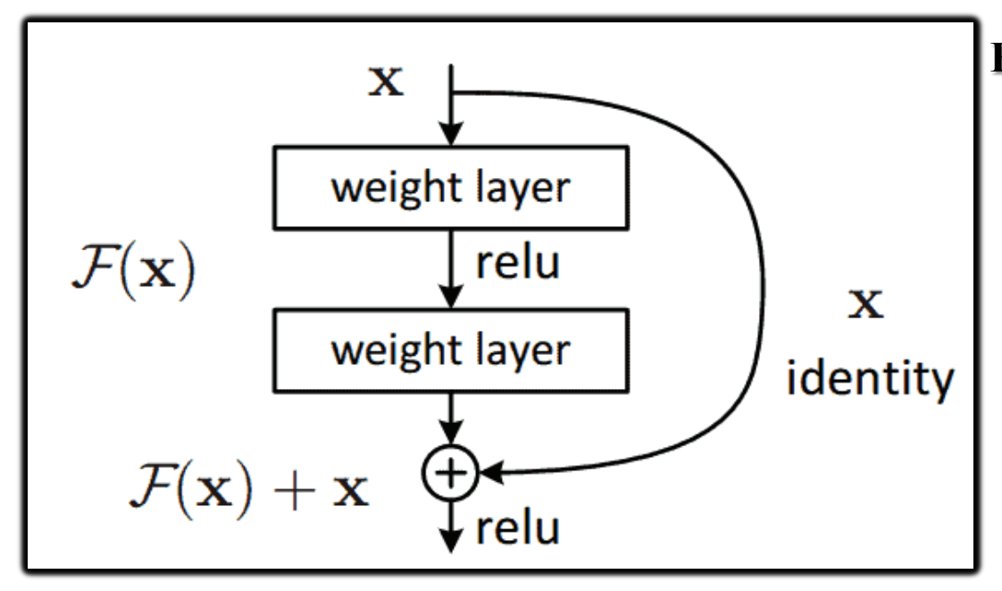

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu
x
identity

Figure 5

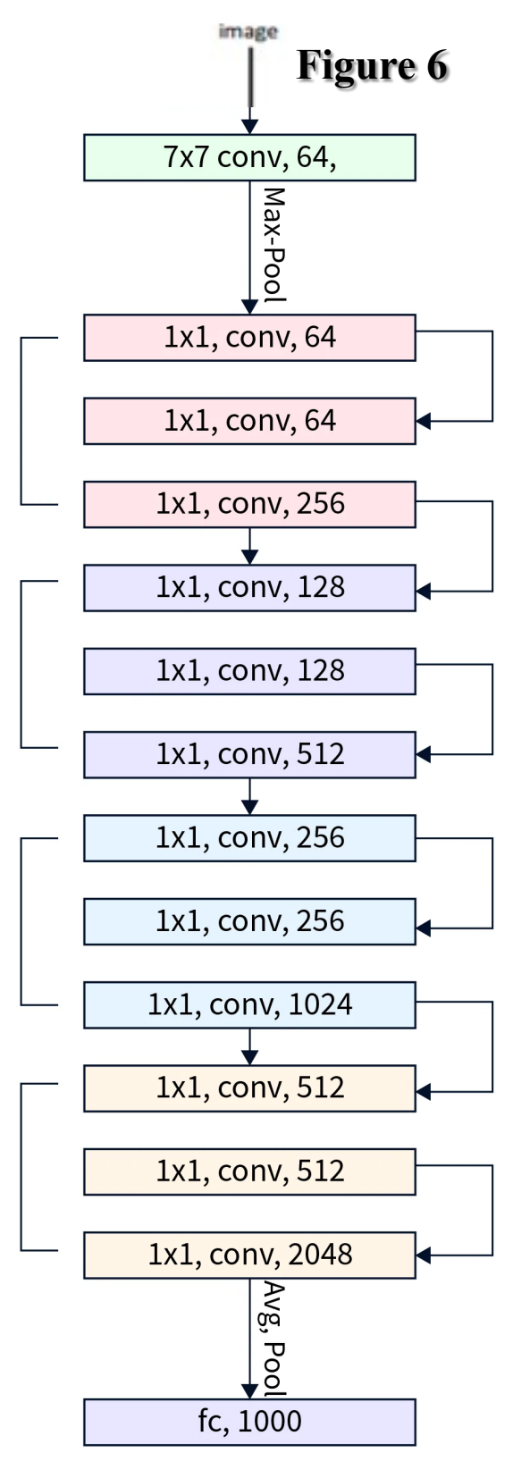


Figure 6

Residual Networks (ResNets) are an extension of CNNs. The difference is that a ResNet reintroduces outputs from previous layers as inputs. As illustrated in Figure 5, the output x from an earlier block is added to the output of a later block.

Why should we add less filtered information?

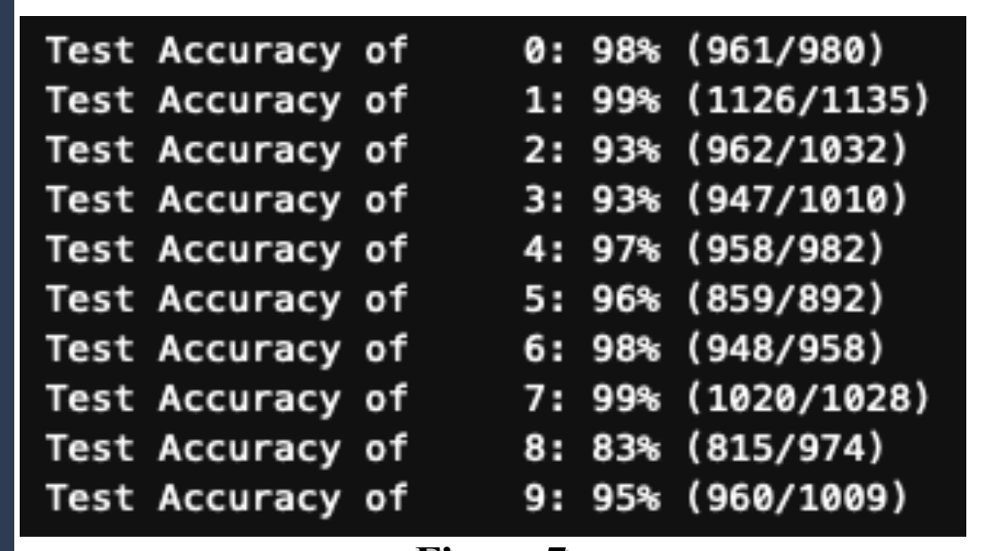
This addition helps the model retain relevant signals from the original image, even after passing through a deep neural network with many layers and filters. When passing an image through a CNN, information can become abstract and be lost when downsizing a convolution. By adding previous, unaltered outputs to newly filtered outputs, the model **retains a closer tie** to the original input that also incorporates additional features to better identify features. Adding the identity also helps resist against **vanishing gradients**. If a neuron's gradient were to be vanishing, the addition would bring back the neuron because the addition ends up adding to the gradient by a constant.

Comparing Neural Networks

Using the FCN model, we were able to get an accuracy of 98% for the MNIST dataset and 55% for CIFAR-10. The CNN model achieved 99% for MNIST and 82% for CIFAR-10.

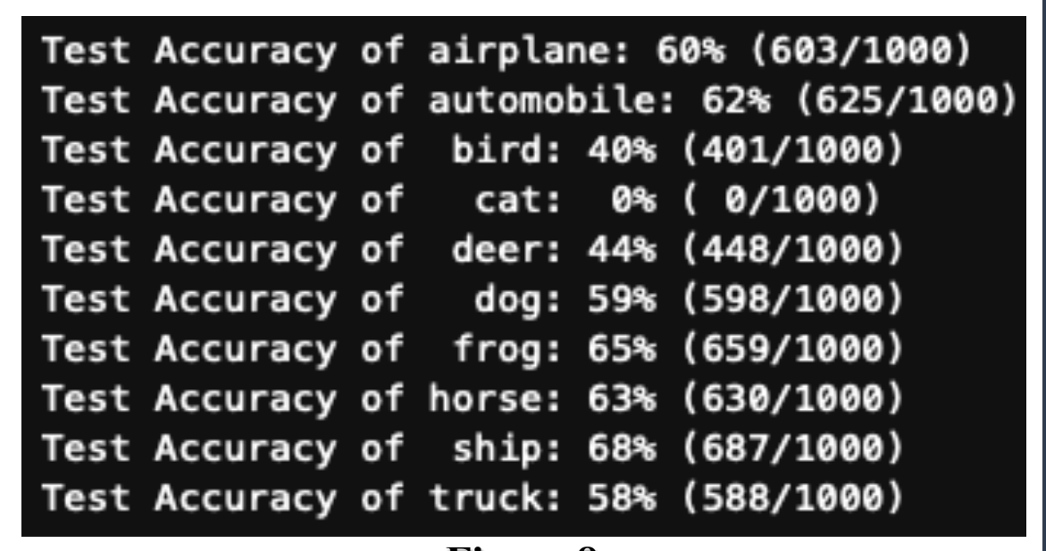
MNIST CIFAR-10

FCN



```
Test Accuracy of     0: 98% (961/980)
Test Accuracy of     1: 99% (1126/1135)
Test Accuracy of     2: 93% (962/1032)
Test Accuracy of     3: 93% (947/1010)
Test Accuracy of     4: 97% (958/982)
Test Accuracy of     5: 96% (859/892)
Test Accuracy of     6: 98% (948/958)
Test Accuracy of     7: 99% (1020/1028)
Test Accuracy of     8: 83% (815/974)
Test Accuracy of     9: 95% (960/1009)
```

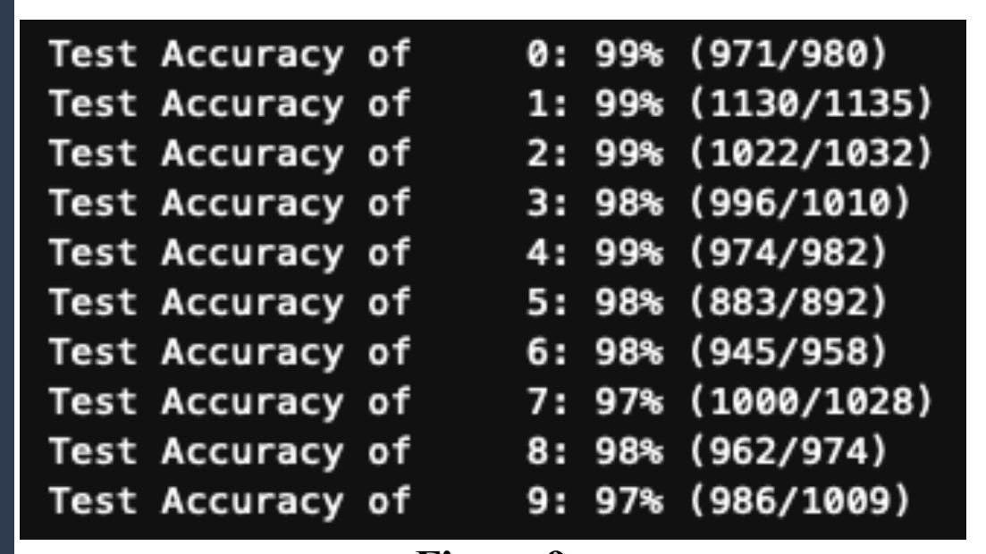
Figure 7



```
Test Accuracy of airplane: 60% (603/1000)
Test Accuracy of automobile: 62% (625/1000)
Test Accuracy of  bird: 40% (401/1000)
Test Accuracy of   cat:  0% ( 0/1000)
Test Accuracy of  deer: 44% (448/1000)
Test Accuracy of   dog: 59% (598/1000)
Test Accuracy of  frog: 65% (659/1000)
Test Accuracy of horse: 63% (630/1000)
Test Accuracy of  ship: 68% (687/1000)
Test Accuracy of truck: 58% (588/1000)
```

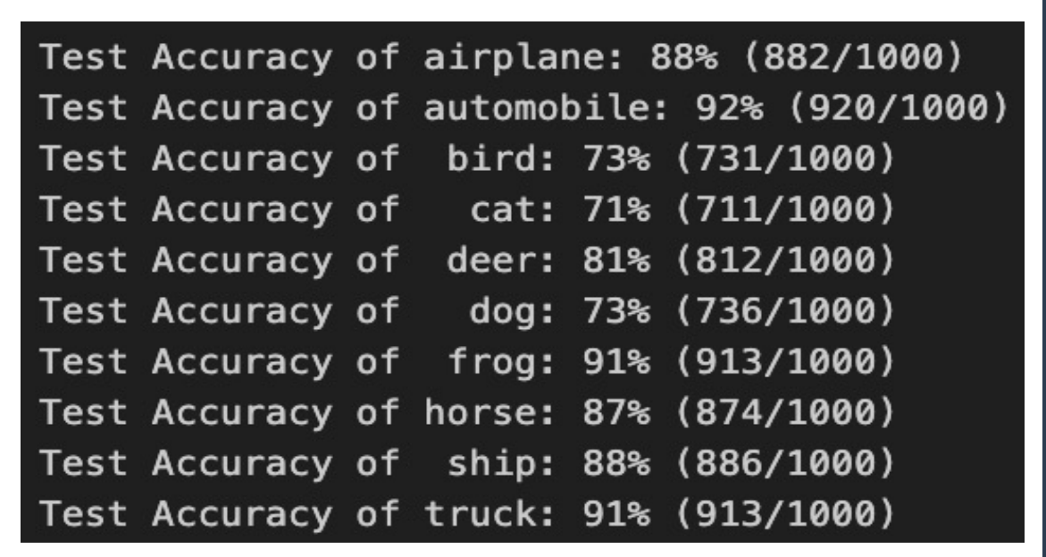
Figure 8

CNN



```
Test Accuracy of     0: 99% (971/980)
Test Accuracy of     1: 99% (1130/1135)
Test Accuracy of     2: 99% (1022/1032)
Test Accuracy of     3: 98% (996/1010)
Test Accuracy of     4: 99% (974/982)
Test Accuracy of     5: 98% (883/892)
Test Accuracy of     6: 98% (945/958)
Test Accuracy of     7: 97% (1000/1028)
Test Accuracy of     8: 98% (962/974)
Test Accuracy of     9: 97% (986/1009)
```

Figure 9

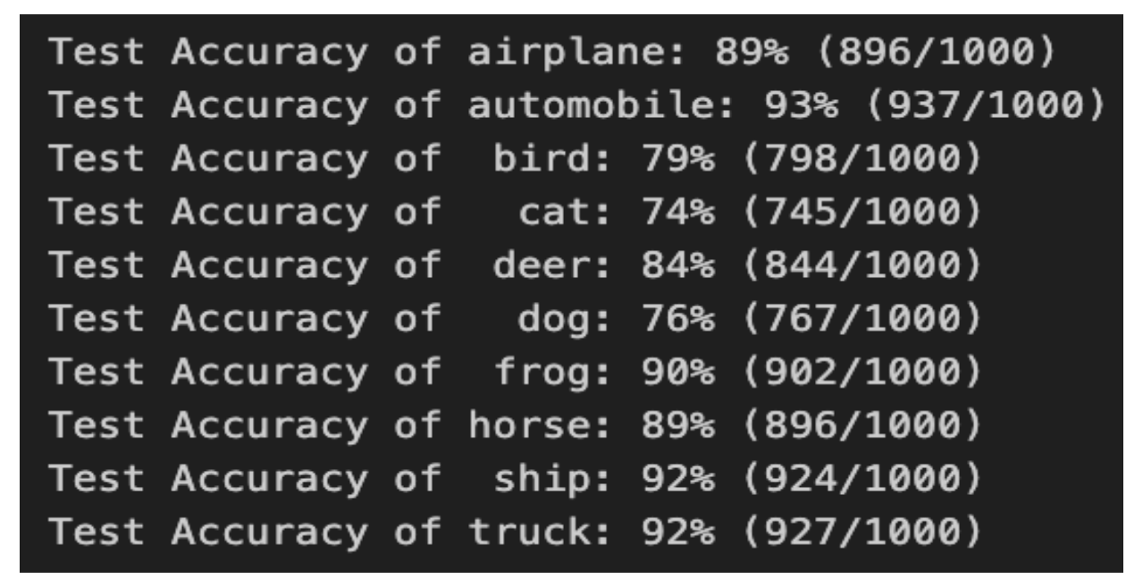


```
Test Accuracy of airplane: 88% (882/1000)
Test Accuracy of automobile: 92% (920/1000)
Test Accuracy of  bird: 73% (731/1000)
Test Accuracy of   cat: 71% (711/1000)
Test Accuracy of  deer: 81% (812/1000)
Test Accuracy of   dog: 73% (736/1000)
Test Accuracy of  frog: 91% (913/1000)
Test Accuracy of horse: 87% (874/1000)
Test Accuracy of  ship: 88% (886/1000)
Test Accuracy of truck: 91% (913/1000)
```

Figure 10

Convolutional Neural Networks (CNNs) excel at identifying complex features compared to Fully Connected Networks (FCNs). As illustrated in Figures 7 and 8, the FCN struggled with certain classifications: achieving only 83% accuracy on digit 8 in the MNIST dataset and failing to correctly identify cats in the CIFAR-10 dataset, with generally low accuracy. In contrast, the CNN demonstrated superior performance, achieving a minimum of 97% accuracy on MNIST and 7% on CIFAR-10, with many classifications near 90%.
While adding regularizations, adjusting parameters, and modifying layers could further enhance accuracy, there are inherent limitations. The FCN performs well with simpler datasets like MNIST, where images have fewer features and less color complexity. However, it falters with more intricate datasets like CIFAR-10, which involve more detailed and varied features.

Comparing Neural Networks



```
Test Accuracy of airplane: 89% (896/1000)
Test Accuracy of automobile: 93% (937/1000)
Test Accuracy of  bird: 79% (798/1000)
Test Accuracy of   cat: 74% (745/1000)
Test Accuracy of  deer: 84% (844/1000)
Test Accuracy of   dog: 76% (767/1000)
Test Accuracy of  frog: 90% (902/1000)
Test Accuracy of horse: 89% (896/1000)
Test Accuracy of  ship: 92% (924/1000)
Test Accuracy of truck: 92% (927/1000)
```

CIFAR-10
ResNet
Figure 10

Our ResNet model was able to achieve an accuracy of 87% for the CIFAR-10 dataset. It outperformed the CNN on every classification task. The reason a ResNet can achieve better accuracy that a CNN, even though they are quite similar, is because ResNet of its architecture. The model allows data that might have been lost, to be used again for learning through its residual layers. This allows the gradient to flow more effectively and make it more resistant to the vanishing gradient problem, allowing faster convergences and the ability to train deeper networks.

References

Bibliography
[1]
I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, Massachusetts: The Mit Press, 2016.
[2]
AIML.com, "What is a Perceptron? What is the role of bias in a perceptron (or neuron)?," *AIML.com*, Jun. 27, 2022. https://aiml.com/what-is-a-perceptron/
[3]
dshahid380, "Convolutional Neural Network," *Medium*, Feb. 24, 2019. https://towardsdatascience.com/covolutional-neural-network-cb0883dd6529
[4]
M. Khatri, "Resnet Block Explanation with a Terminology Deep Dive," *Medium*, Oct. 23, 2018. https://medium.com/@MaheshNKhatri/resnet-block-explanation-with-a-terminology-deep-dive-989e15e3d691 (accessed Jul. 24, 2024).
[5]
A. Sarkar, "Creating Deeper Bottleneck ResNet from Scratch using Tensorflow," *Medium*, Sep. 02, 2020. https://towardsdatascience.com/creating-deeper-bottleneck-resnet-from-scratch-using-tensorflow-93e11ff7eb02

Acknowledgements

GEAR UP
I ILLINOIS