Assignment 2

Mingzhao Liu(ml116)

Task 1 Visualizing a CNN with CIFAR10

a. Dataset

i. 10 classes
ii. Image size 28 * 28
iii. Resolution 32 * 32

b. Train LetNet5 on CIFAR10

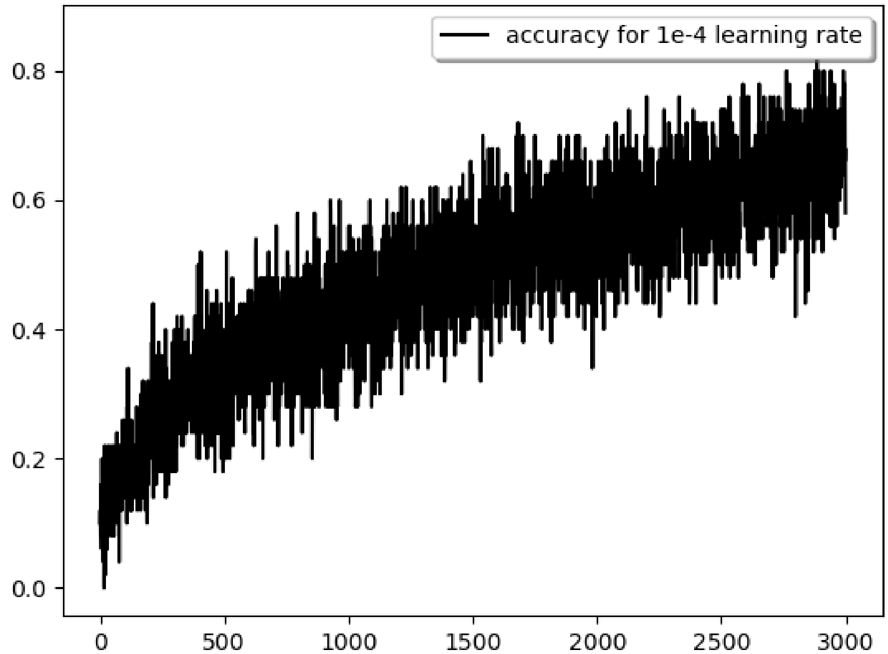
According to the results below, the learning rate with 1e-3 performs best among these three.

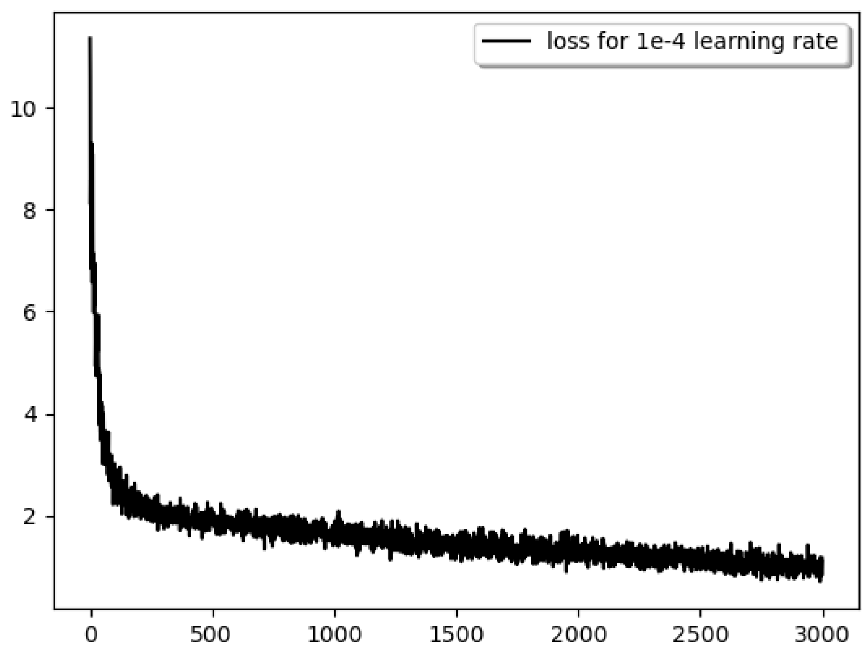
Thus, I decided to use this learning rate to compare the different optimizers.

And based on the results of training, the Adam optimizer performs better than the gradient descent optimizer

i. Different learning rate with 1e-4, 1e-3 and 1e-2

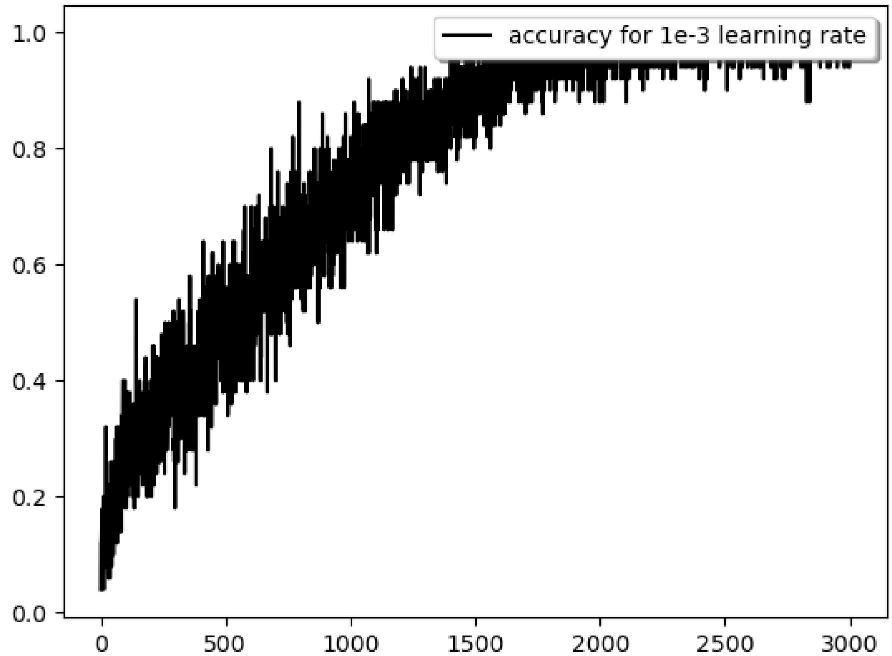
1. 1e-4

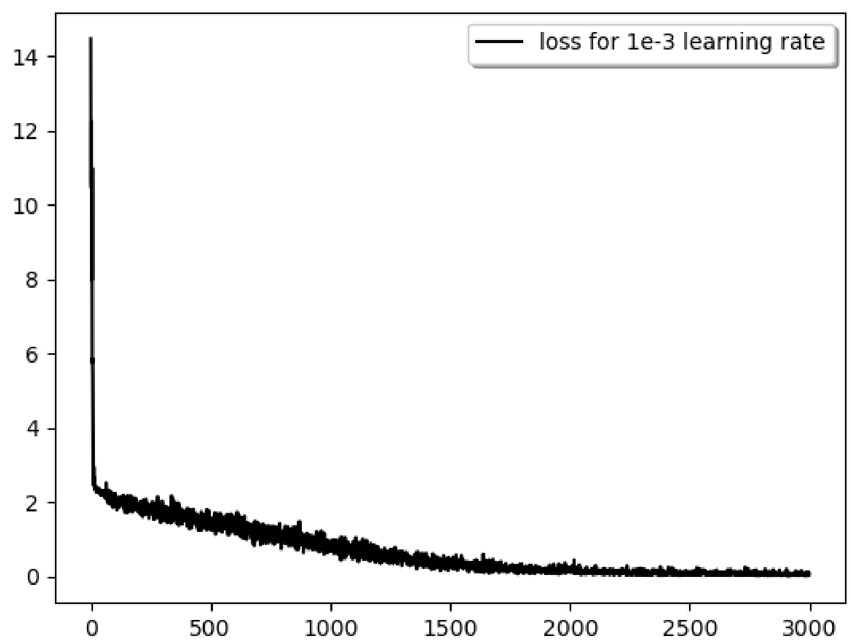




Test accuracy: 0.48

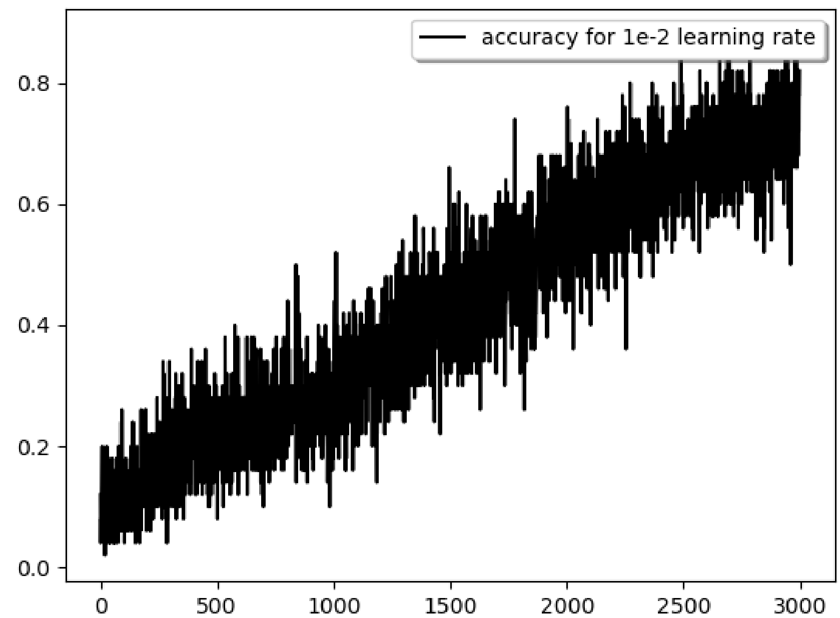
2. 1e-3

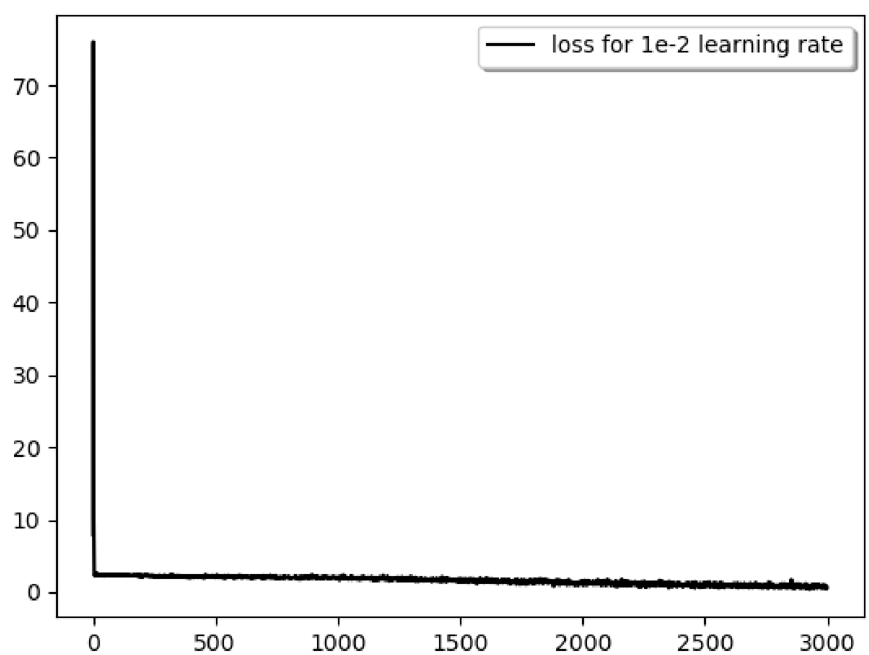




Test accuracy: 0.487

3. 1e-2

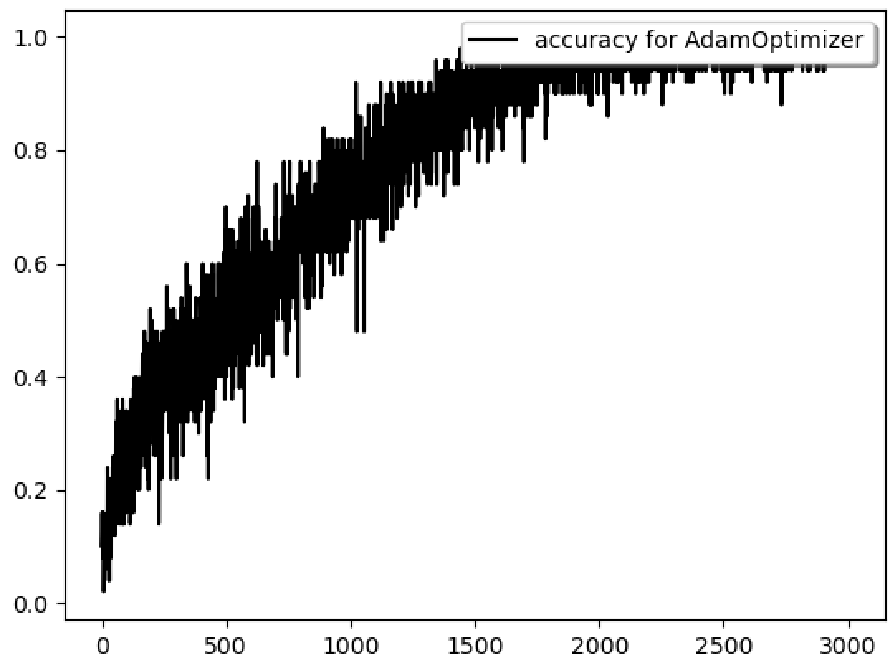


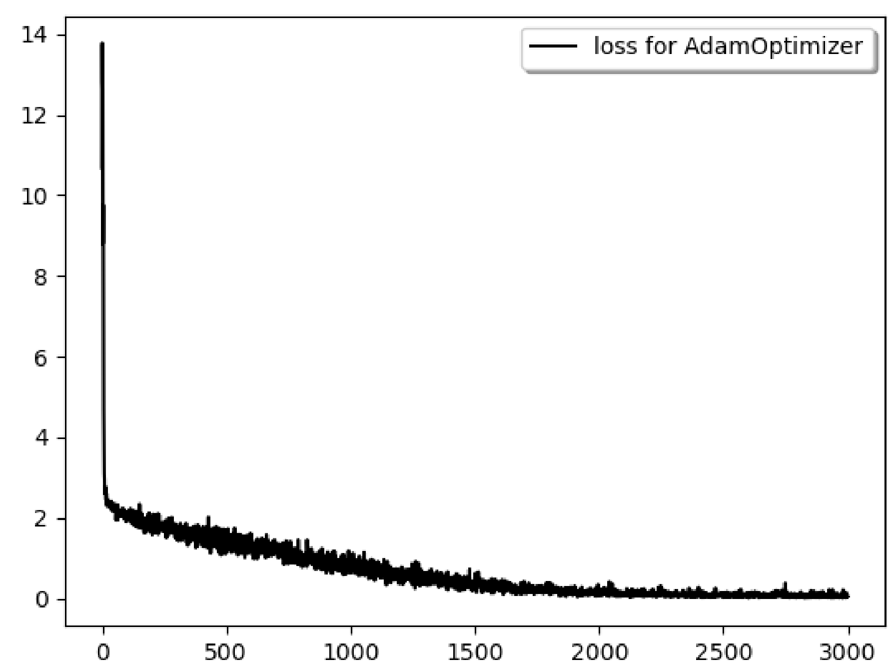


test accuracy 0.323

ii. Different optimizer:

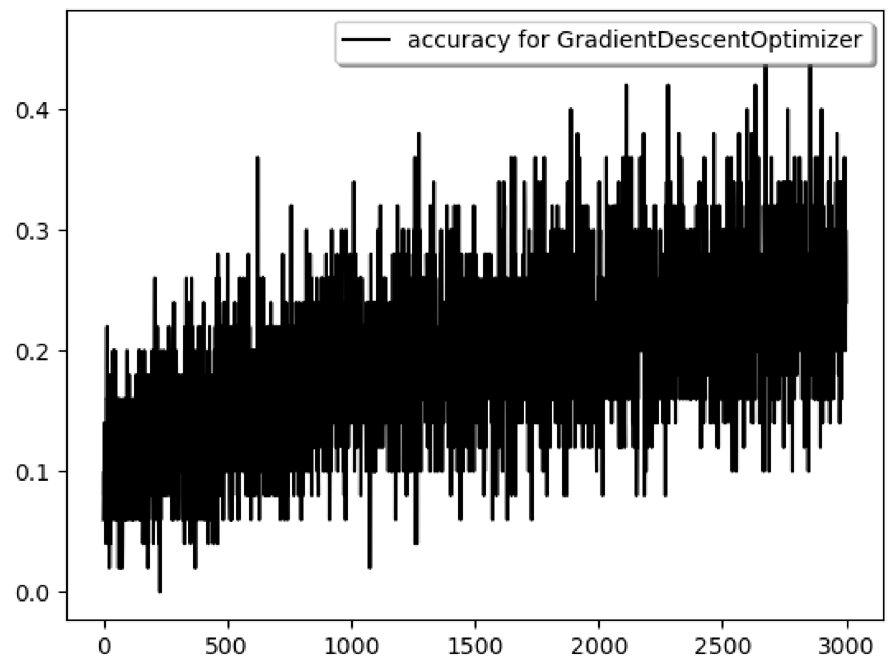
1. AdamOptimizer

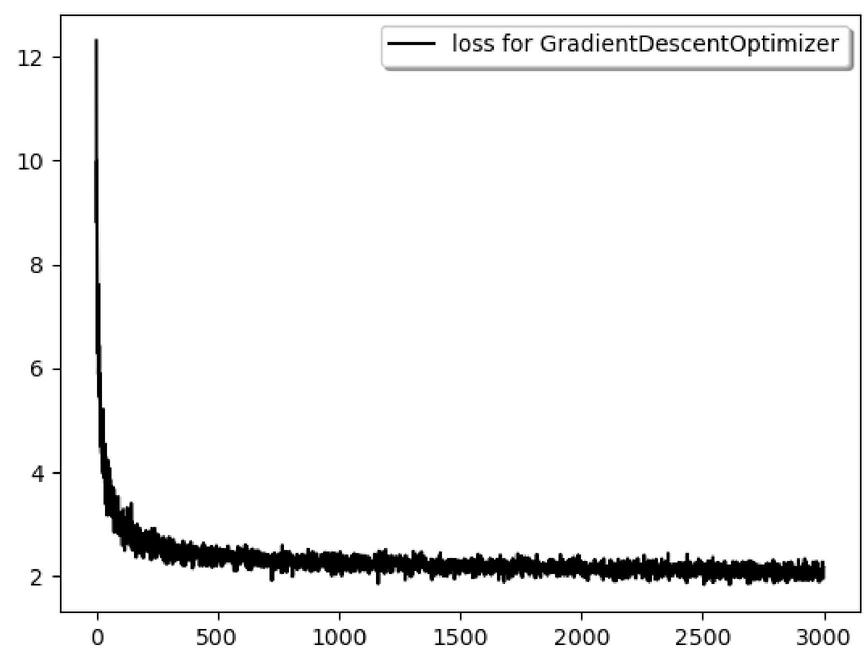




Test accuracy: 0.485

2. GradientDescentOptimizer

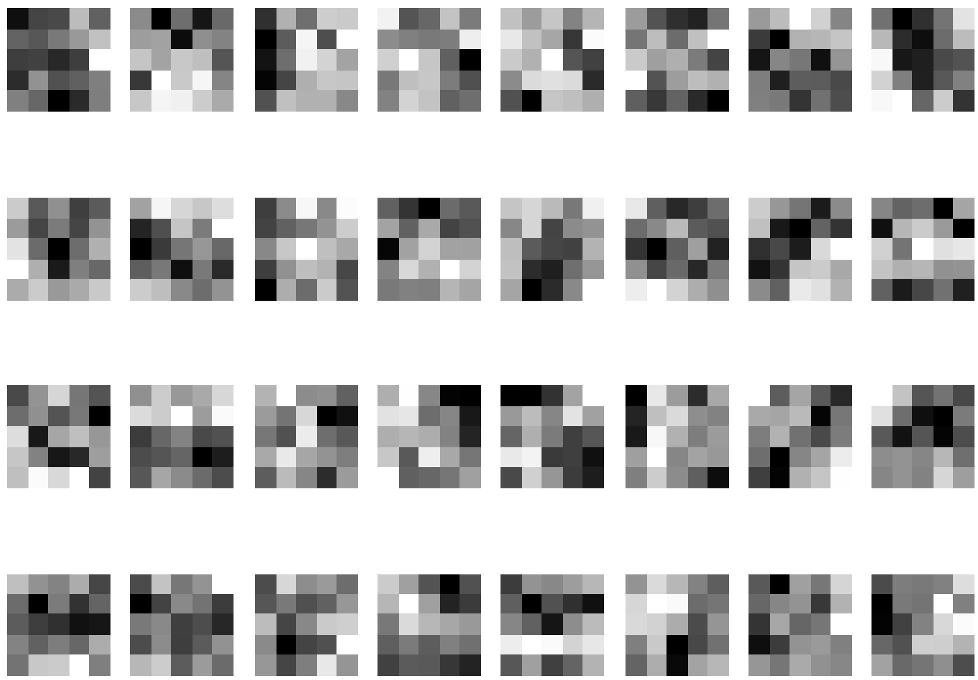




Test accuracy: 0.286

c. Visualize the trained network

i. First convolutional layer weights



Statistics of the activations in the convolutional layers on test images:

activation1: mean 0.07992579787969589, variance 0.013498915359377861
activation2: mean 0.016598623245954514, variance 0.009720266796648502

Task 2 Key Ideas for Visualizing and Understanding Convolutional Networks

This article aims to solve the problem of why large convolutional network models perform so well, or how to improve them with the visualization of CNN. The author introduces a novel visualization technique that can deeply understand the functions of the middle feature layer and the operation of the classifier. Their visualization technique uses a multi-layer deconvolution network (deconvnet) to project feature activations back into the input pixel space. The author uses standard fully-supervised convnet models throughout the paper, which map color 2D input images to probability vectors through a series of layers. The first few layers of the network are traditional fully connected networks, and the last layer is the softmax classifier.

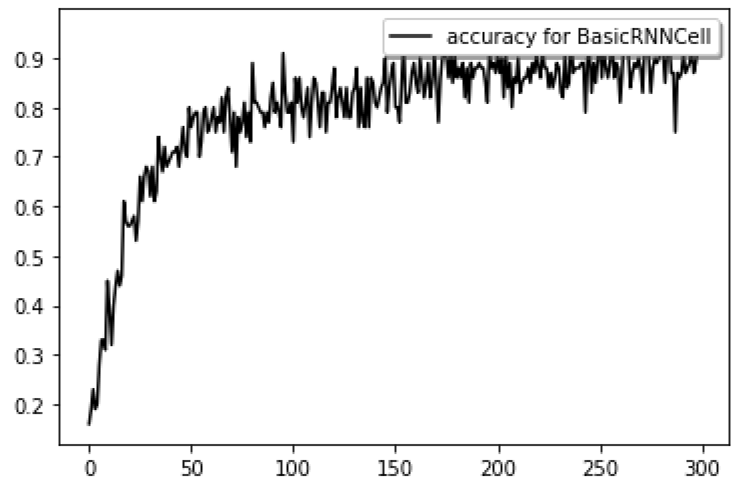
To visualize the features of each layer, the author proposes a new method of mapping these activities back to the input pixel space, showing the input patterns that initially led to a given activation in the feature map. To check the convnet, a deconvnet is attached to each of its layers. To reconstruct the activity of each layer, unpooling, rectification, and filters are repeatedly used until the input pixel space is reached.

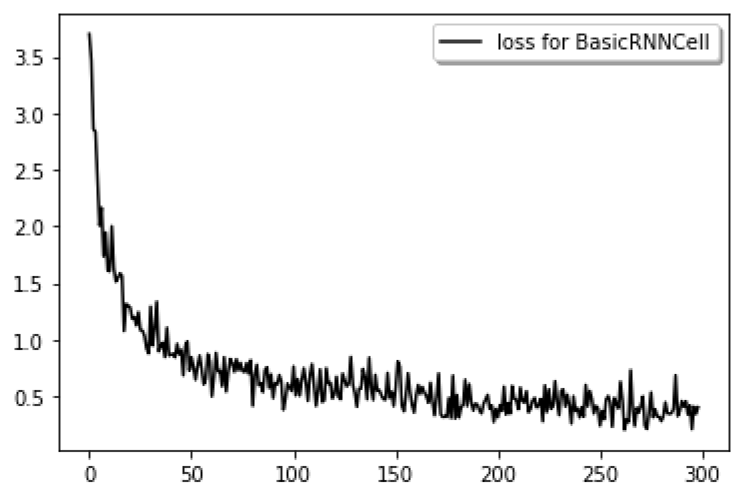
The model was trained on cropping and resizing on the ImageNet 2012 training set. Later, they also showed how the ImageNet training model can generalize well to other data sets. The author uses deconvnet to visualize the feature activation on the ImageNet validation set. This paper can help the work of feature visualization, feature evolution and feature invariance during training. In addition, it helps with architecture selection, occlusion sensitivity, and correspondence analysis.

Task 3 Build and train an RNN on MNIST

a. Setup the RNN

Number of nodes: 128
Learning rate: 1e-3
Number of iteration: 3000
Optimizer: Adam optimizer

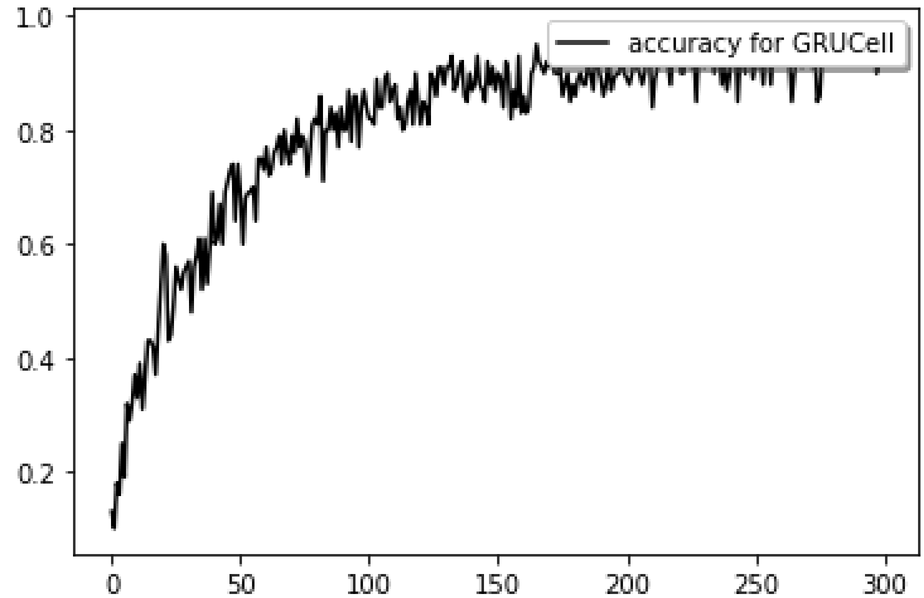


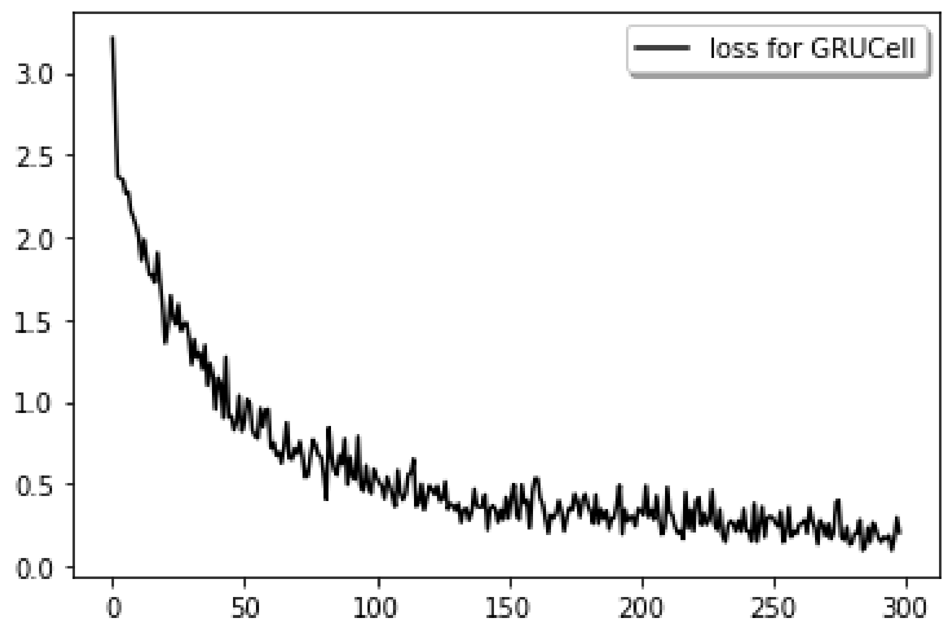


Testing Accuracy: 0.8853

b. Using LSTM or GRU instead of the RNN

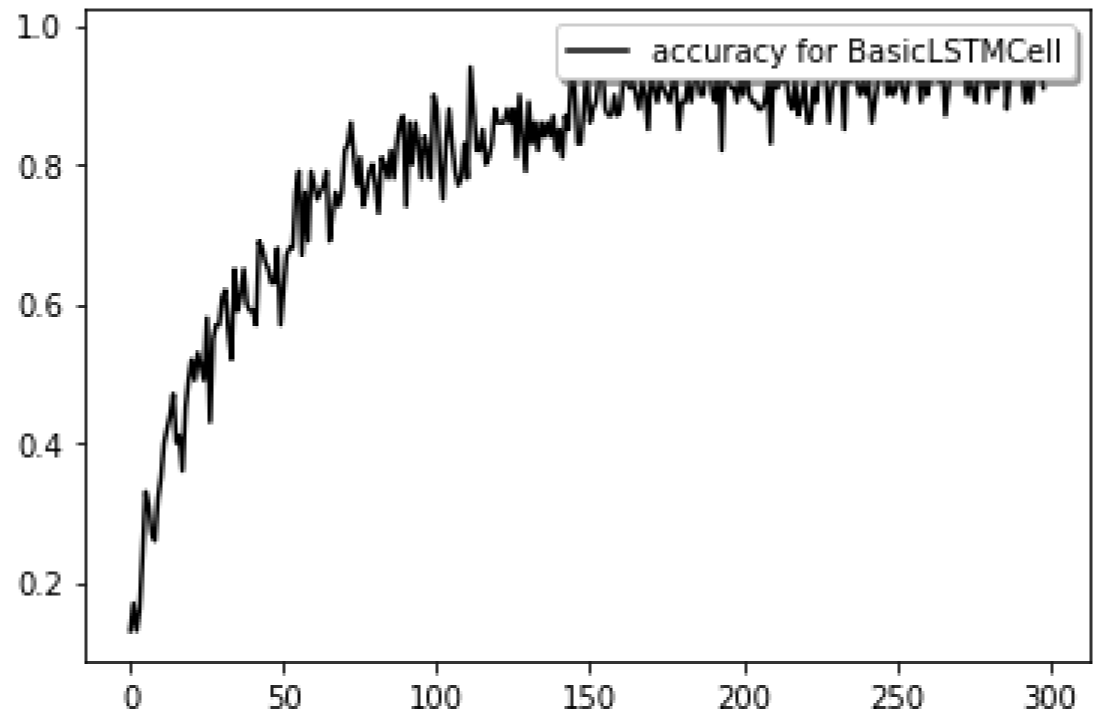
i. GRU

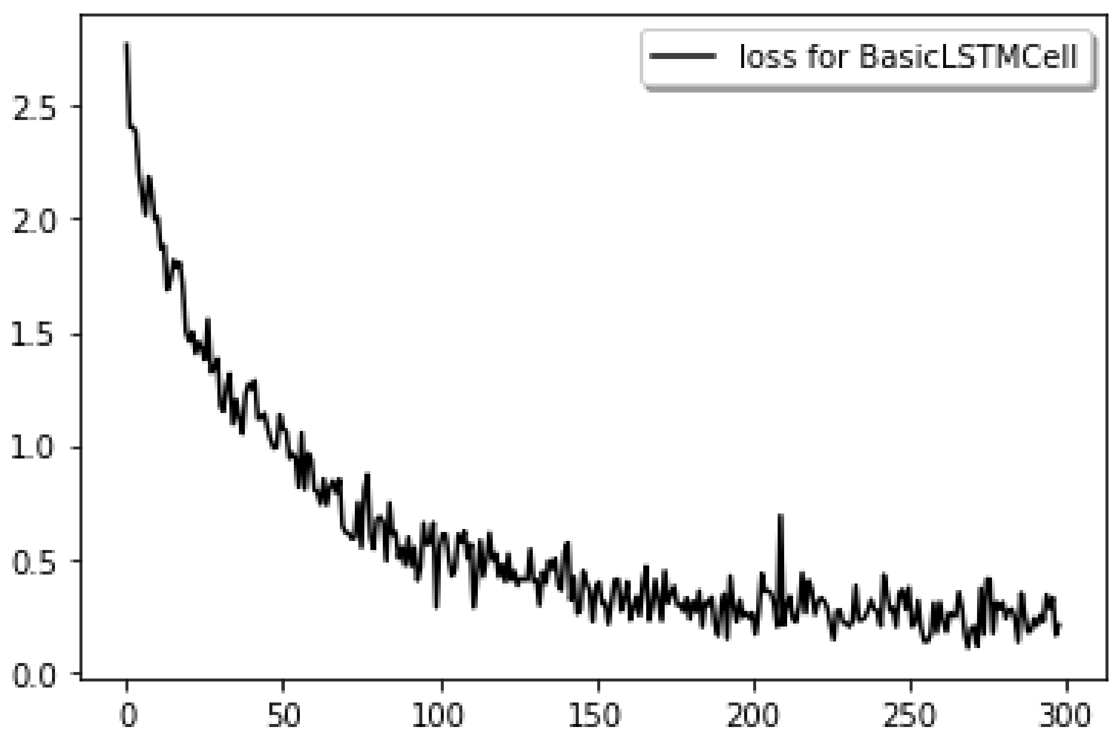




Test accuracy: 0.9293

ii. LSTM

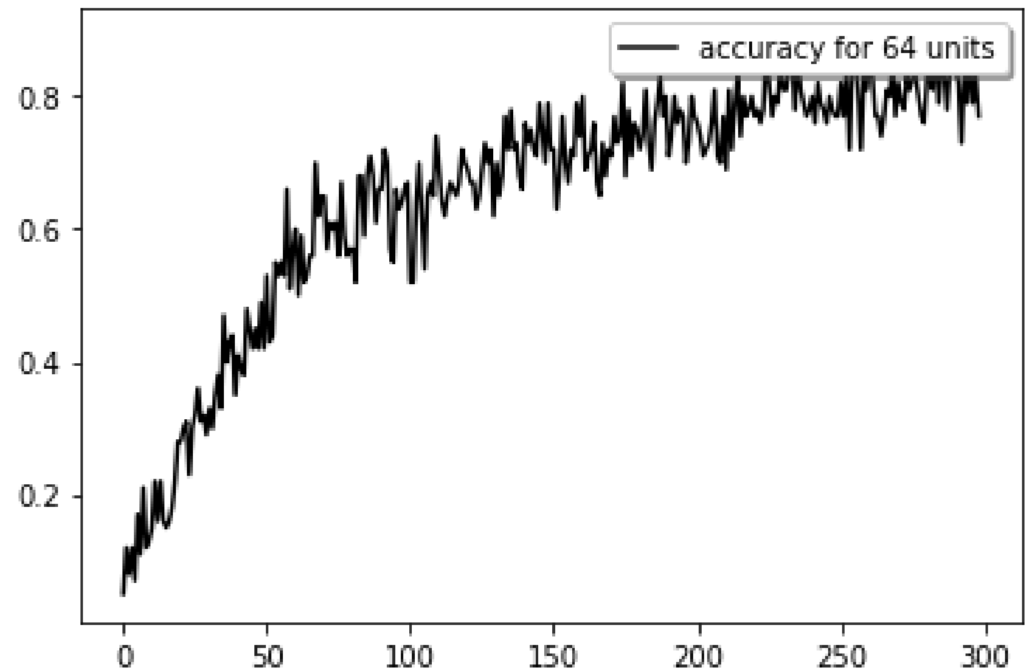


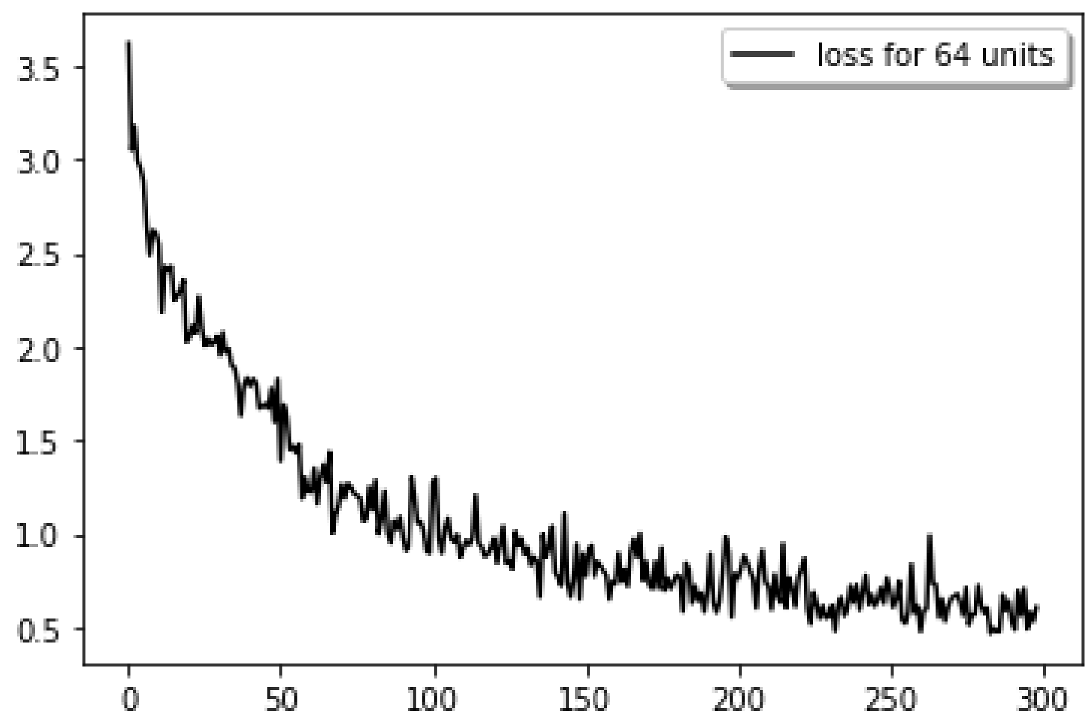


Test accuracy: 0.892

B. different number of hidden units

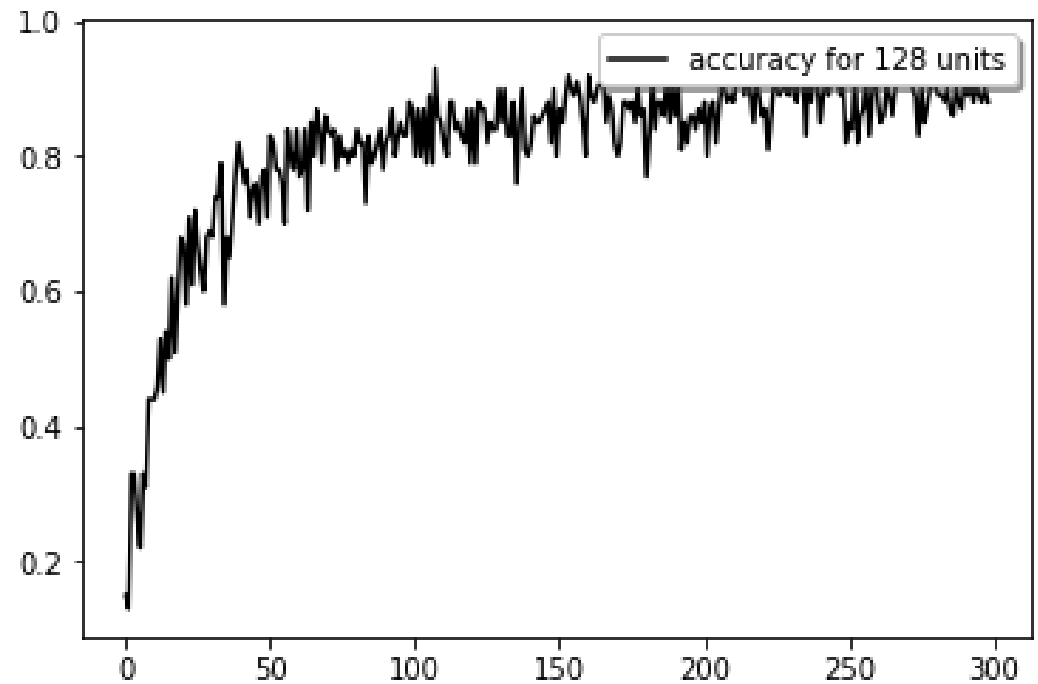
1. 64 units

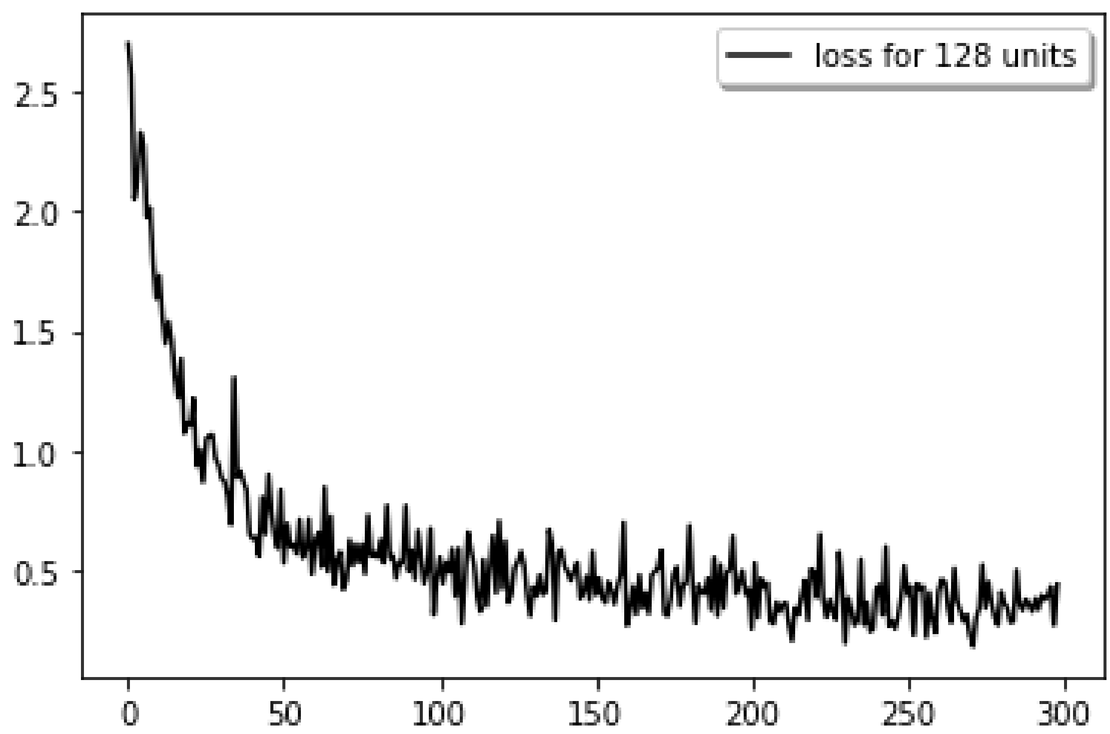




Test accuracy: 0.8115

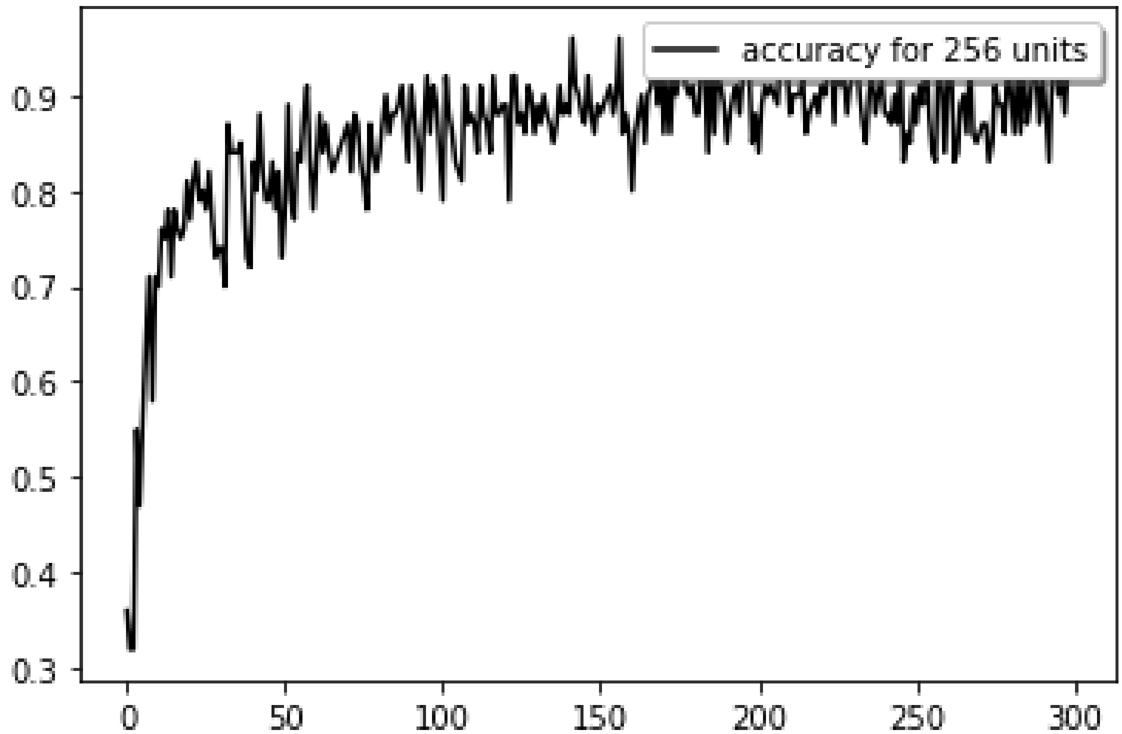
2. 128 units

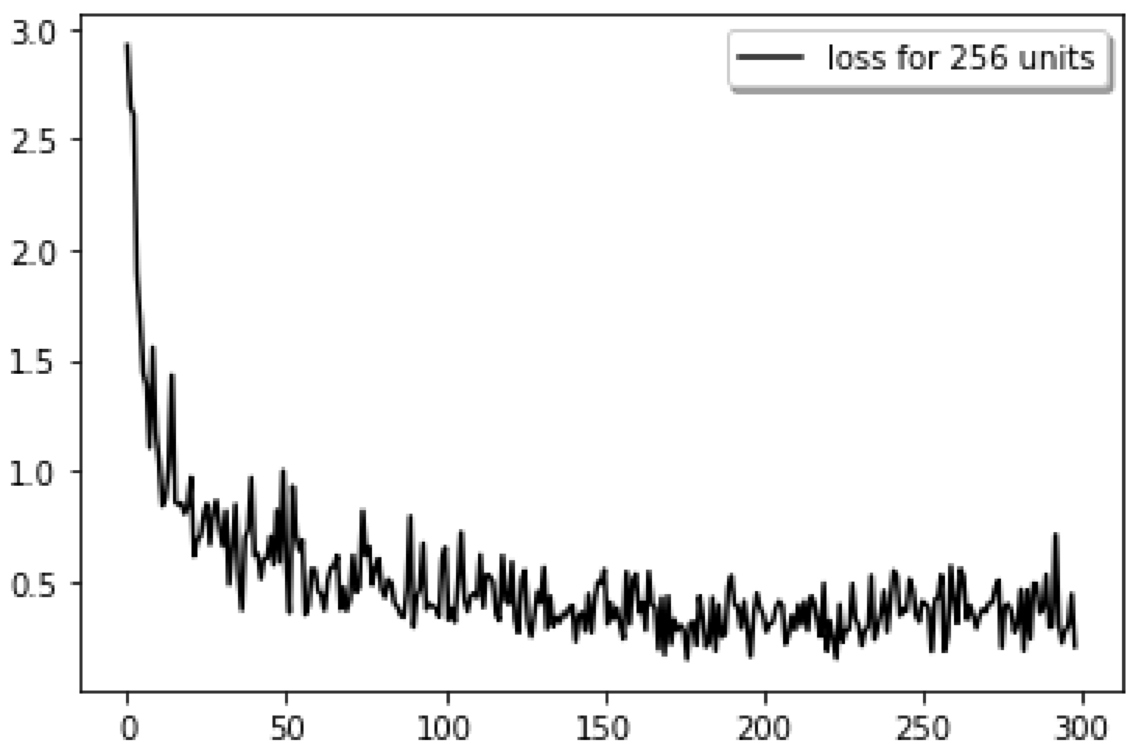




Test accuracy: 0.8764

3. 256 units





Test accuracy: 0.9047

According to the test accuracy results of previous training, we can easily know that the other two models perform better than the RNN, and the best number of neurons for RNN should be 256 since the test accuracy is the highest.

c. Compare with CNN in assignment 1

Similarities:

1. Both CNN and RNN are used on deep learning training task and the number of their layers can be very large.
2. Both of them can work on image recognition or some other tasks.
3. Some details of them are similar, like training methods, gradient methods, etc.

Differences:

RNN focuses on the dependency between lines of images, while CNN focuses on capturing the same patterns on all the different subfields of the image.
CNN allows to process and classify data in a more hierarchical way, and somewhat mimics the human visual system.
Thus, CNNs are ideal for images and videos processing.
Since RNN could handle the dependency, RNNs are ideal for text and speech analysis.