

---

# Fully connected Autoencoder

## Table of Contents

Initialization .....	1
Load and convert data to the desired format .....	1
Autoencoder 1 .....	1
Autoencoder 2 .....	2
Testing parameters .....	2
Test network 1 .....	2
Test network 2 .....	3
Divide networks .....	3
Network 1 montage .....	3
Network 1 scatter plot .....	4
Network 2 montage .....	4
Network 2 scatter .....	5
Check rule .....	6

## Initialization

```
clear; clc;
loadNetworks = true;
```

## Load and convert data to the desired format

```
[xTrain, tTrain, xValid, tValid, xTest, tTest] = LoadMNIST(1);
```

## Autoencoder 1

```
layers1 = [
    sequenceInputLayer(784)

    fullyConnectedLayer(50, 'WeightsInitializer', 'glorot')
    reluLayer

    fullyConnectedLayer(2, 'WeightsInitializer', 'glorot')
    reluLayer

    fullyConnectedLayer(784, 'WeightsInitializer', 'glorot')
    reluLayer

    regressionLayer];

options = trainingOptions('adam', ...
    'MiniBatchSize', 8192, ...
    'InitialLearnRate', 0.001, ...
    'Shuffle', 'every-epoch', ...
    'MaxEpochs', 10000, ...
```

```
    'Plots','training-progress');

if ~loadNetworks
    [net1, trainingInfo1] = trainNetwork(xTrain, xTrain, layers1,
    options);
end
```

## Autoencoder 2

```
layers2 = [
    sequenceInputLayer(784)

    fullyConnectedLayer(50, 'WeightsInitializer', 'glorot')
    reluLayer

    fullyConnectedLayer(4, 'WeightsInitializer', 'glorot')
    reluLayer

    fullyConnectedLayer(784, 'WeightsInitializer', 'glorot')
    reluLayer

    regressionLayer];

if ~loadNetworks
    [net2, trainingInfo2] = trainNetwork(xTrain, xTrain, layers2,
    options);
end

if ~loadNetworks

    save('Homework3_2_networks.mat', 'net1', 'trainingInfo1', 'net2', 'trainingInfo2')
end

if loadNetworks
    load('Homework3_2_networks.mat');
end
```

## Testing parameters

```
pauseTime = 1;
numberOfImages = 5;
```

## Test network 1

```
figure(1)
index = randi([1 length(xTrain) - numberOfImages]);
for iImage = index:index+numberOfImages
    imageData = xTrain(:,iImage);

    subplot(1,2,1)
    image = reshape(imageData, 28, 28, 1);
    imshow(image);
```

```
subplot(1,2,2)
predictedImageData = predict(net1, imageData);
predictedImage = reshape(predictedImageData, 28, 28, 1);
imshow(predictedImage);

pause(pauseTime);
end
```

## Test network 2

```
figure(2)
for iImage = index:index+numberOfImages
    imageData = xTrain(:,iImage);

    subplot(1,2,1)
    image = reshape(imageData, 28, 28, 1);
    imshow(image);

    subplot(1,2,2)
    predictedImageData = predict(net2, imageData);
    predictedImage = reshape(predictedImageData, 28, 28, 1);
    imshow(predictedImage);
    a = iImage;
    pause(pauseTime);
end
```

## Divide networks

```
net1_layers_encode(1:5) = [net1.Layers(1:4); regressionLayer];
net1_layers_decode(1:5) = [sequenceInputLayer(2); net1.Layers(5:8)];
net1_encode = assembleNetwork(net1_layers_encode);
net1_decode = assembleNetwork(net1_layers_decode);

net2_layers_encode(1:5) = [net2.Layers(1:4); regressionLayer];
net2_layers_decode(1:5) = [sequenceInputLayer(4); net2.Layers(5:8)];
net2_encode = assembleNetwork(net2_layers_encode);
net2_decode = assembleNetwork(net2_layers_decode);
```

## Network 1 montage

```
currentDigit = 0;
iImage = 20;
figure(3)
clf
while currentDigit < 10
    if find(tTrain(:,iImage))-1 == currentDigit
        imageData = xTrain(:,iImage);

        subplot(5,4,currentDigit*2+1)
        image = reshape(imageData, 28, 28, 1);
        imshow(image);
```

```
        subplot(5,4,currentDigit*2+2)
        predictedImageData = predict(net1, imageData);
        predictedImage = reshape(predictedImageData, 28, 28, 1);
        imshow(predictedImage);
        currentDigit = currentDigit+1;
    end
    iImage = iImage+1;
end
```

## Network 1 scatter plot

```
successfulDigits = [1];
colors = {'red'};
b = [];

figure(4)
clf
plotDigit = 1;

for iImage = 1:1000
    digit = find(tTrain(:,iImage))-1;
    a = predict(net1_encode, xTrain(:,iImage));
    if ismember(digit, successfulDigits)
        hold on
        if plotDigit < length(successfulDigits)+1
            if digit == successfulDigits(plotDigit)
                b(plotDigit) = plot(a(1), a(2), '.', 'color',
char(colors(successfulDigits == digit)));
                plotDigit = plotDigit + 1;
                continue
            end
        end
        plot(a(1), a(2), '.', 'color', char(colors(successfulDigits ==
digit)));
    else
        plot(a(1), a(2), '.blue')
    end
end
b(plotDigit) = plot(a(1), a(2), '.', 'color', 'blue');
legend(b, '1', 'Other digits','Location', 'north')
xlabel('First network output')
ylabel('Second network output')
```

## Network 2 montage

```
currentDigit = 0;
iImage = 300;
figure(5)
clf
while currentDigit < 10
    if find(tTrain(:,iImage))-1 == currentDigit
        imageData = xTrain(:,iImage);
```

```
subplot(5,4,currentDigit*2+1)
image = reshape(imageData, 28, 28, 1);
imshow(image);

subplot(5,4,currentDigit*2+2)
predictedImageData = predict(net2, imageData);
predictedImage = reshape(predictedImageData, 28, 28, 1);
imshow(predictedImage);
currentDigit = currentDigit+1;
end
iImage = iImage+1;
end
```

## Network 2 scatter

```
successfulDigits = [1,2,7];
colors = {'red', 'green', 'cyan'};
b = [];

figure(6)
clf
plotDigit = 1;

for iImage = 1:1000
    digit = find(tTrain(:,iImage))-1;
    a = predict(net2_encode, xTrain(:,iImage));
    if ismember(digit, successfulDigits)
        hold on
        if plotDigit < length(successfulDigits)+1
            if digit == successfulDigits(plotDigit)
                b(plotDigit) = plot(a(1), a(2), '.', 'color',
char(colors(successfulDigits == digit)));
                plotDigit = plotDigit + 1;
                continue
            end
        end
        plot(a(1), a(2), '.', 'color', char(colors(successfulDigits ==
digit)));
    else
        plot(a(1), a(2), '.blue')
    end
end
b(plotDigit) = plot(a(1), a(2), '.', 'color', 'blue');
legend(b, '1','2','7', 'Other digits')
xlabel('First network output')
ylabel('Second network output')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
b = [];

figure(7)
clf
```

```
plotDigit = 1;

for iImage = 1:1000
    digit = find(tTrain(:,iImage))-1;
    a = predict(net2_encode, xTrain(:,iImage));
    if ismember(digit, successfulDigits)
        hold on
        if plotDigit < length(successfulDigits)+1
            if digit == successfulDigits(plotDigit)
                b(plotDigit) = plot(a(3), a(4), '.', 'color',
char(colors(successfulDigits == digit)));
                plotDigit = plotDigit + 1;
                continue
            end
        end
        plot(a(3), a(4), '.', 'color', char(colors(successfulDigits ==
digit)));
    else
        plot(a(3), a(4), '.blue')
    end
end
b(plotDigit) = plot(a(3), a(4), '.', 'color', 'blue');
legend(b, '1','2','7', 'Other digits')
xlabel('Third network output')
ylabel('Fourth network output')
```

## Check rule

```
figure(8)
a1 = [25; 3; 50; 175];
imageData = predict(net2_decode, a1);
image = reshape(imageData, 28, 28, 1);
imshow(image);
```

```
figure(9)
a2 = [9; 9; 2; 5];
imageData = predict(net2_decode, a2);
image = reshape(imageData, 28, 28, 1);
imshow(image);
```

```
figure(10)
a3 = [25; 25; 110; 35];
imageData = predict(net2_decode, a3);
image = reshape(imageData, 28, 28, 1);
imshow(image);
```

*Published with MATLAB® R2020b*