ETC3555 2018 - Lab 6

Neural networks and backpropagation

Cameron Roach and Souhaib Ben Taieb

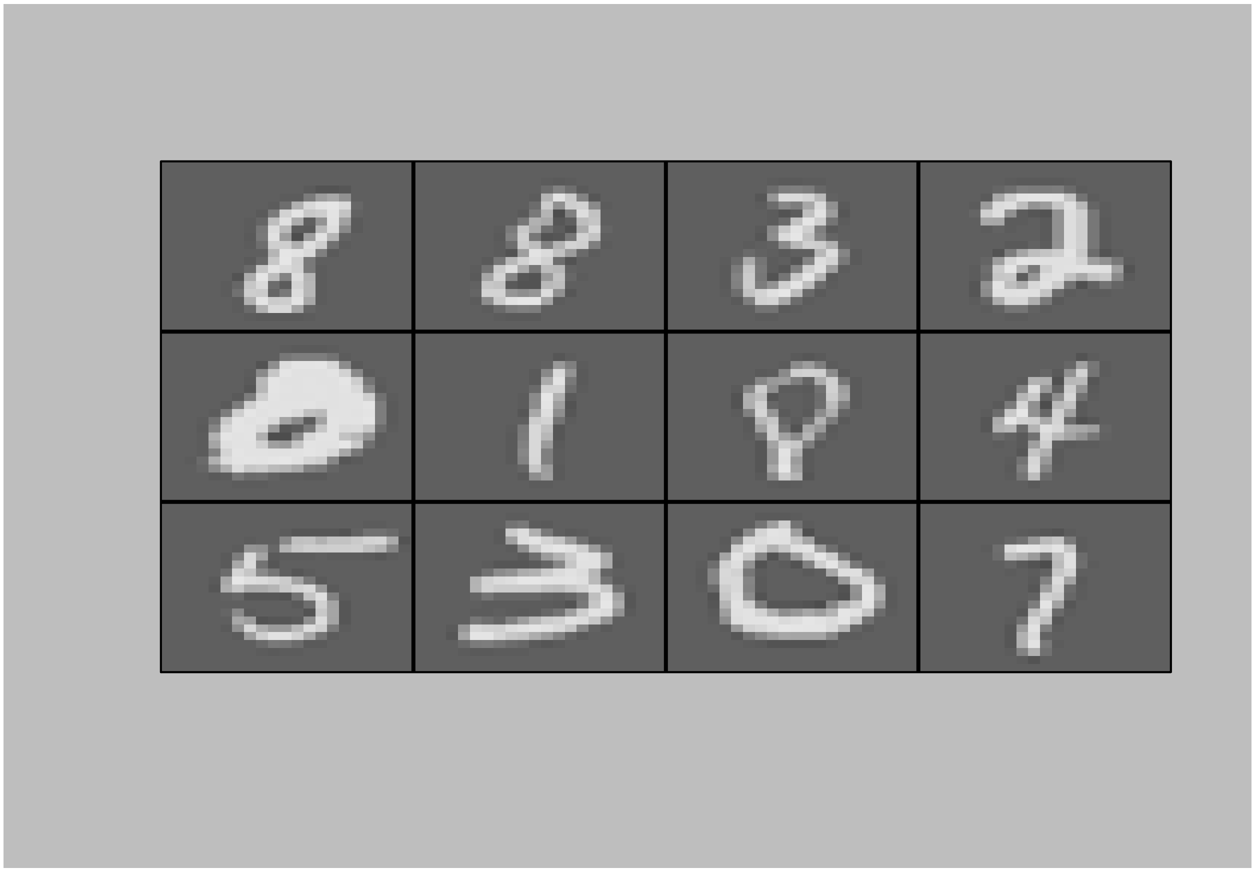
29 August 2018

Exercise 1

We will consider a dataset which contains 5000 training examples of handwritten digits (from 0 to 9), where each training example is a 20 pixels by 20 pixels grayscale image of the digit. Each pixel indicate the grayscale intensity at that location in the image. We have vectorized the 20 by 20 grid of pixels into a 400-dimensional vector. This gives us a 5000 by 400 input matrix X where each row is a training example for a handwrittend digit image. The output vector is a 5000 dimensional vector y where the digits "1" to "9" are labeled as "1" to "9" while a "0" is labeled "10".

The following code visualizes some training examples

```r
source("plotDigits.R")
load("digits.Rdata") # load X and y
X <- cbind(1, X)
n <- nrow(X)
plotDigits(X[sample(n, 12), -1])
```



Consider a neural network with an input layger, one hidden layer and a output layer with 10 units (corresponding to the 10 digit classes). Do not forget the extra bias units which always outputs +1. A logistic activation function will be used for all units. This neural network can be used for multi-class classification. The prediction for x will be the label that has the largest output. In other words, we assign class C to x where $C = \text{argmax}_{k \in \{1,2,\ldots,10\}} h_k(x)$ and $h_k(x)$ is the value of the kth output unit.

Complete the function *feedforward_predict* which apply forward propagation to compute the prediction for x. This function will take the following arguments:

1. `x`: a vector of dimension $(d^{(0)} + 1) \times 1$.
2. `W1`: a weight matrix of dimension $(d^{(0)} + 1) \times d^{(1)}$.
3. `W2`: a weight matrix of dimension $(d^{(1)} + 1) \times d^{(2)}$.

(Note: We will use the matrix notations presented in Lecture 12)

```
sigmoid <- function(z) {
  g <- 1 / (1 + exp(-1 * z))
  g
}

feedforward_predict <-  function(x, W1, W2){
  x0 <- x
  s1 <- t(W1) %*% x0
  x1 <- rbind(1, sigmoid(s1))
  s2 <- t(W2) %*% x1
  x2 <- sigmoid(s2)
  h <- which.max(x2)
  h
}
```

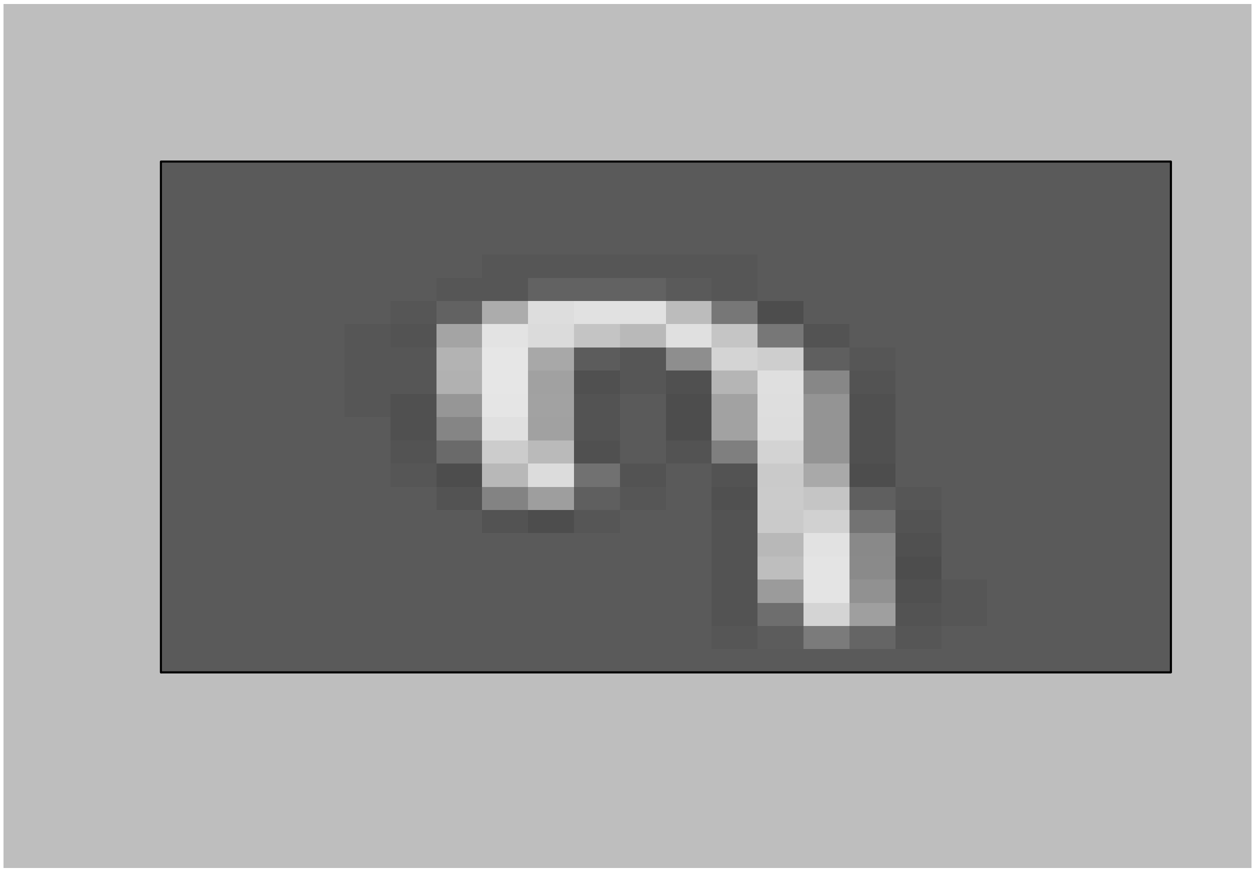
The file *weights.Rdata* provides the network parameters W_1 and W_2 already trained by us. You should see that the classification accuracy is about 97.5% for the training data.

```
load("weights.Rdata")
W1_given <- t(W1)
W2_given <- t(W2)

pred <- sapply(seq(nrow(X)), function(i){
   feedforward_predict( matrix(X[i, ], ncol = 1)  , W1_given, W2_given)
})

print(paste("Training Set Accuracy: ", mean(pred == y) * 100))
```

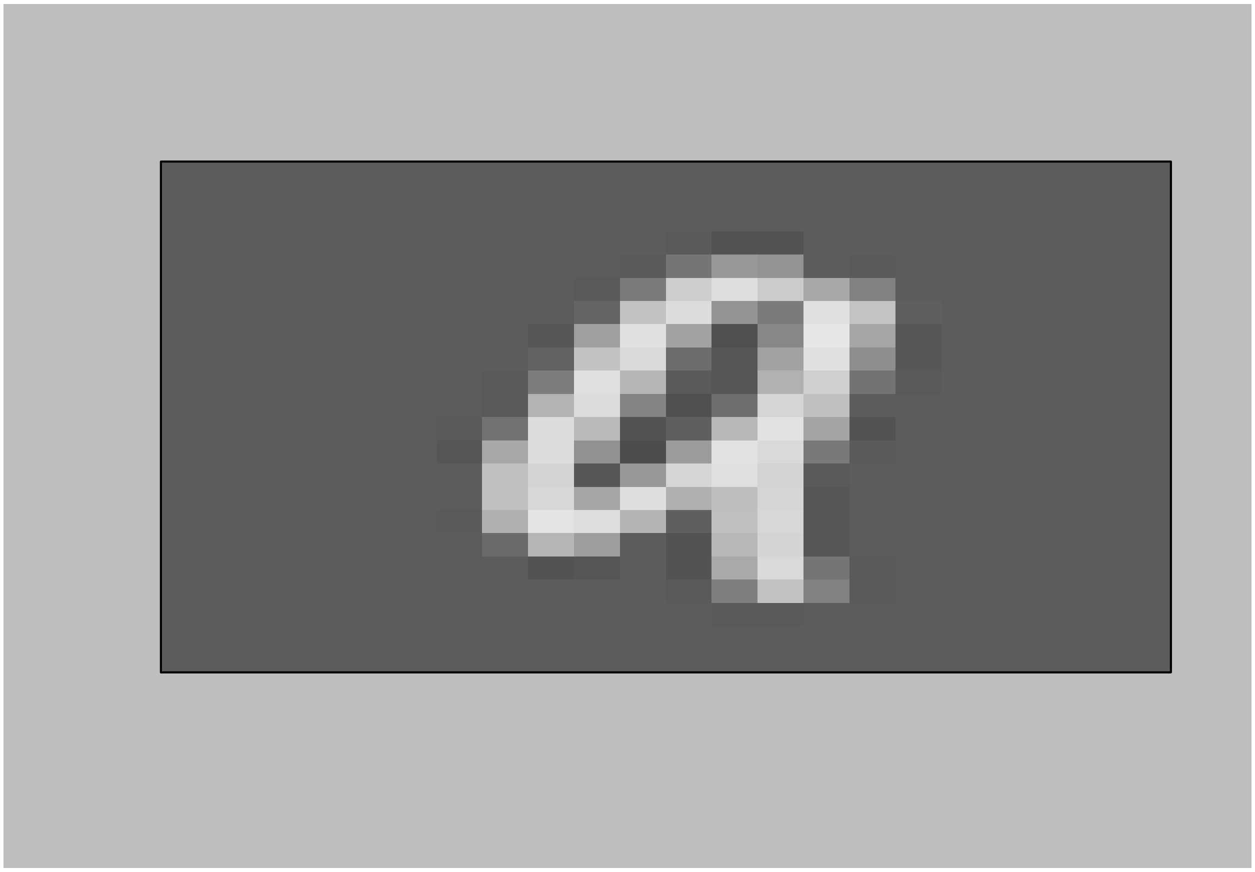
```
## [1] "Training Set Accuracy:  97.52"
```

The following code can be used to visualize some predictions of the neural networks.

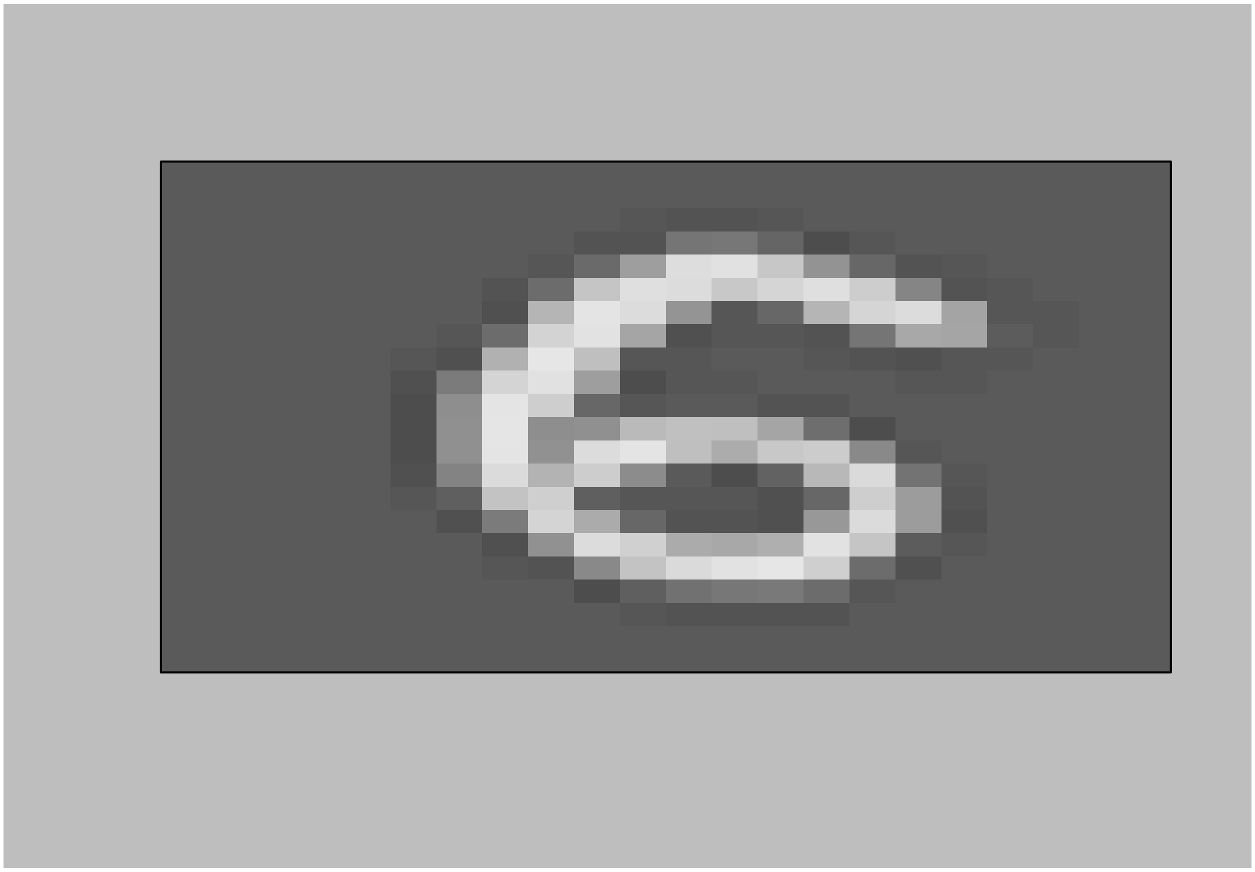
```
id_missclass <- which(y != pred)
id_class <- which(y == pred)

ids <- c(sample(id_missclass, 3), sample(id_class, 3))

for (i in  ids){
  plotDigits(X[i, -1])
  print(paste("Neural network prediction : ", pred[i] , " (digit ", pred[i]%%10, ")", " - ", y[i], sep
}
```



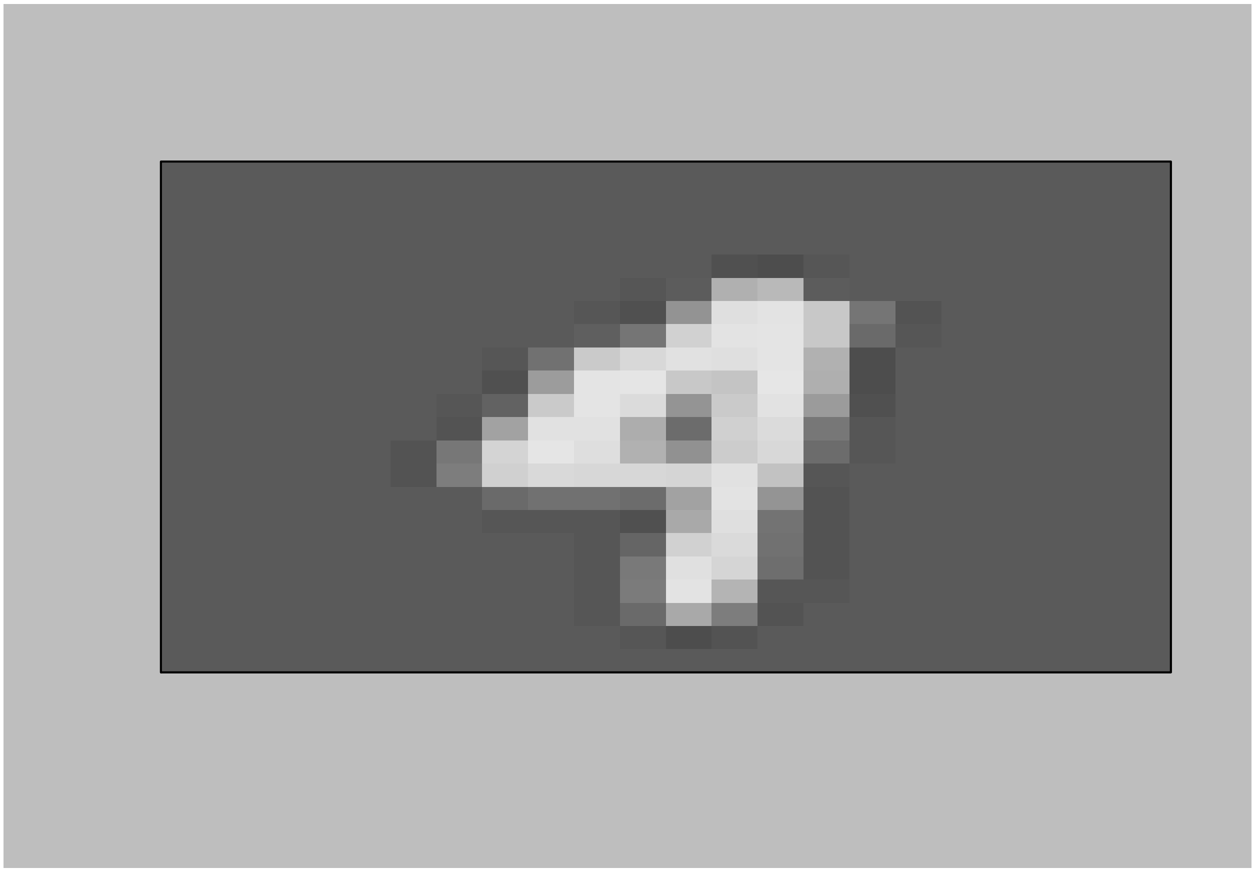
```
## [1] "Neural network prediction : 9 (digit 9) - 7"
```



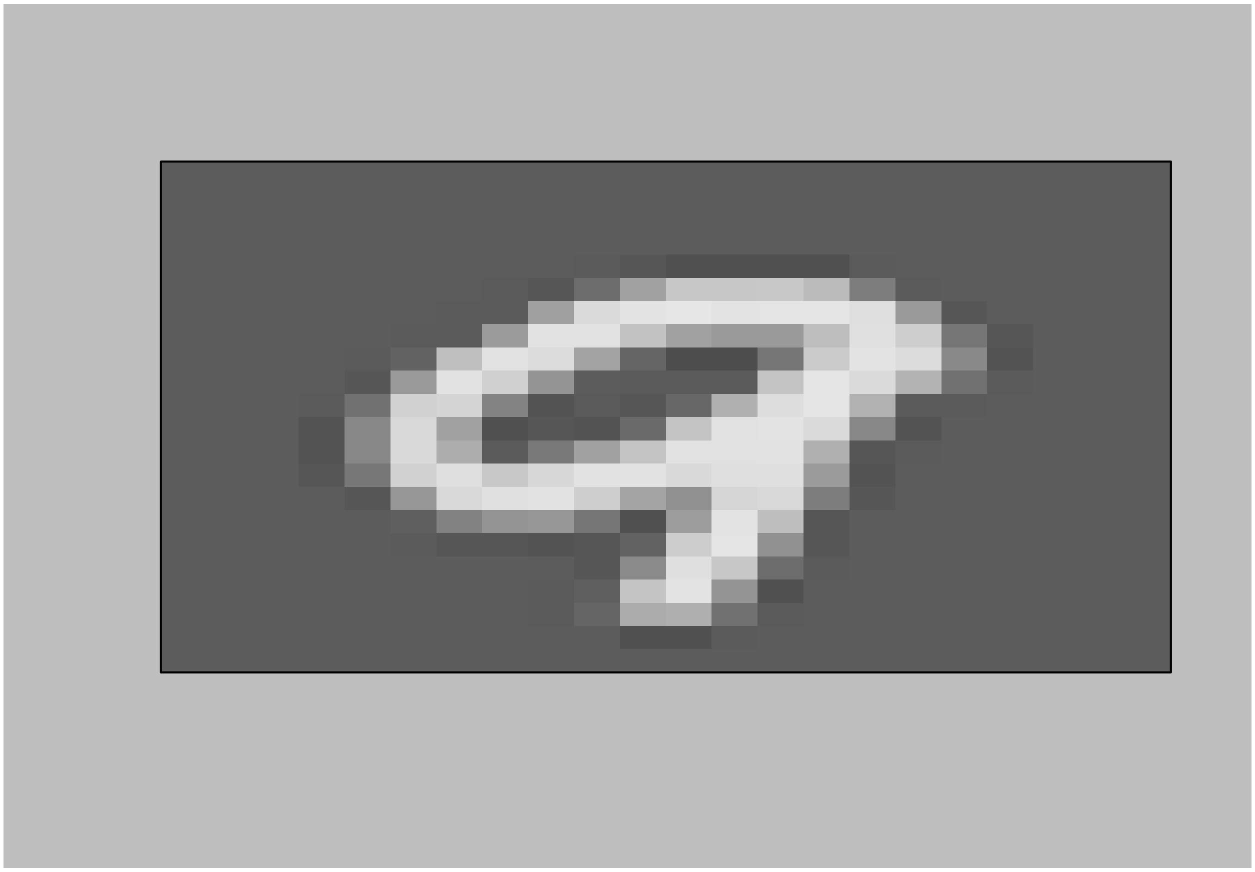
```
## [1] "Neural network prediction : 8 (digit 8) - 9"
```



```
## [1] "Neural network prediction : 5 (digit 5) - 6"
```



```
## [1] "Neural network prediction : 4 (digit 4) - 4"
```



```
## [1] "Neural network prediction : 9 (digit 9) - 9"
```

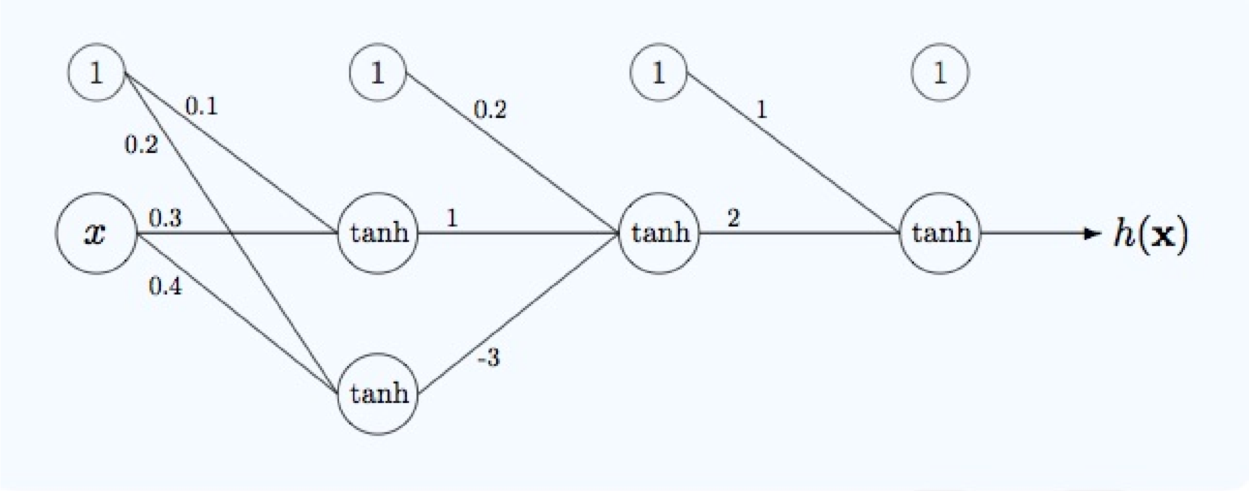
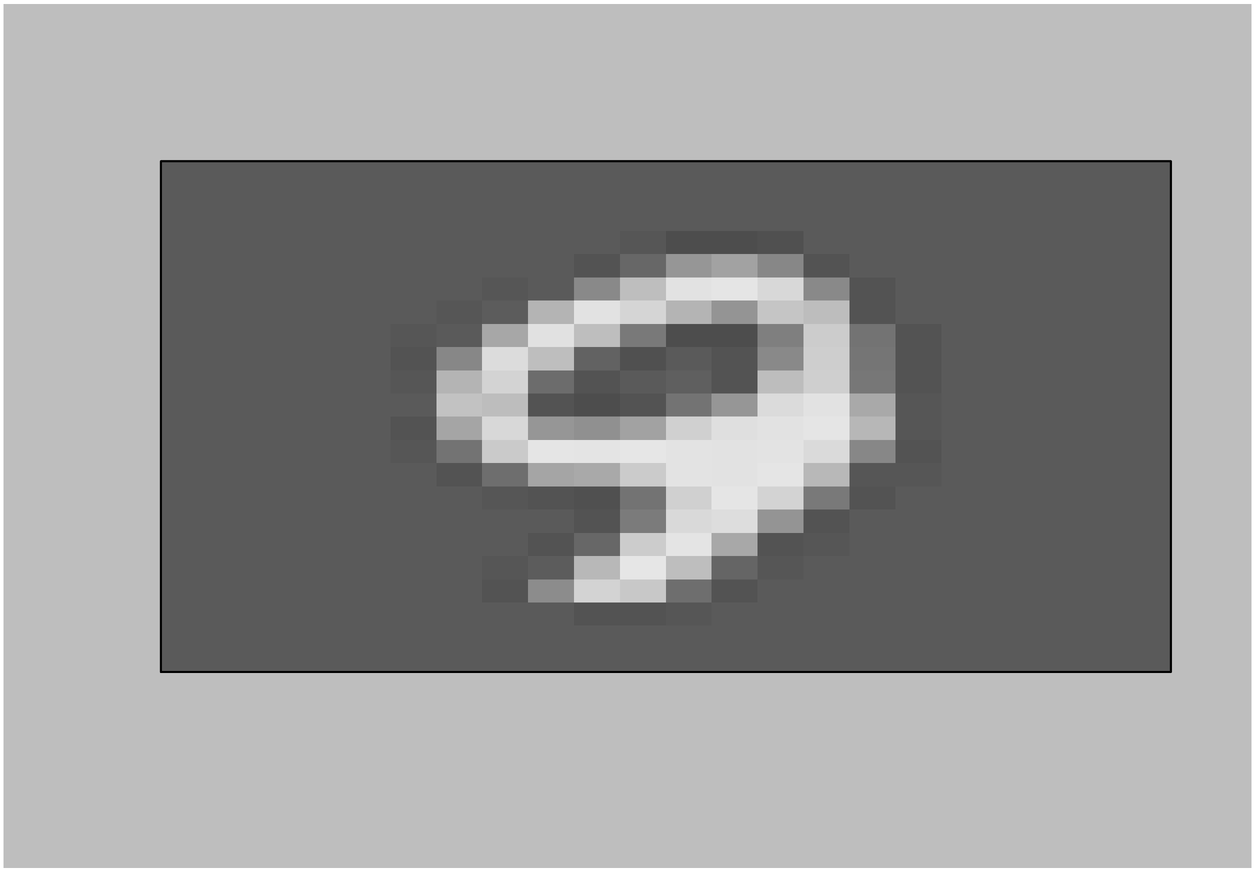


Figure 1: Source: Abu-Mostafa et al. Learning from data. AMLbook.



```
## [1] "Neural network prediction : 9 (digit 9) - 9"
```

Exercise 2

Consider the following networks, and suppose we want to minimize squared errors losses, i.e. $e = (y - h(\mathbf{x}))^2$. (Note: We will use the matrix notations presented in Lecture 12)

1. What are the values of $W^{(1)}$, $W^{(2)}$ and $W^{(3)}$?

```r
W1 <- cbind(c(0.1, 0.3), c(0.2, 0.4))
W2 <- matrix(c(0.2, 1, -3), ncol = 1)
W3 <- matrix(c(1, 2), ncol = 1)
```

2. Run forward propagation for the data point x = -2. What are the values of $\mathbf{x}^{(0)}$, $\mathbf{s}^{(1)}$, $\mathbf{x}^{(1)}$, $\mathbf{s}^{(2)}$, $\mathbf{x}^{(2)}$, $\mathbf{s}^{(3)}$, $\mathbf{x}^{(3)}$?

```r
x <- -2
x0 <- matrix(c(1, x), ncol = 1)
s1 <- t(W1) %*% x0
x1 <- rbind(1, tanh(s1))
s2 <- t(W2) %*% x1
x2 <- rbind(1, tanh(s2))
s3 <- t(W3) %*% x2
x3 <- tanh(s3)

print(x0)
```

```
##      [,1]
## [1,]    1
## [2,]   -2
```

```r
print(s1)
```

```
##      [,1]
## [1,] -0.5
## [2,] -0.6
```

```r
print(x1)
```

```
##            [,1]
## [1,]  1.0000000
## [2,] -0.4621172
## [3,] -0.5370496
```

```r
print(s2)
```

```
##          [,1]
## [1,] 1.349032
```

```r
print(x2)
```

```
##           [,1]
## [1,] 1.0000000
## [2,] 0.8738245
```

```r
print(s3)
```

```
##          [,1]
## [1,] 2.747649
```

```r
print(x3)
```

```
##           [,1]
## [1,] 0.9918215
```

3. Run backward propagation for the data point x = -2, y = 2. What are the values of $\delta^{(3)}$, $\delta^{(2)}$ and $\delta^{(1)}$?

```
y <- 2
delta3 <- 2 * (x3 - y) * (1 - x3^2)
theta_prime_s2 <- 1 - tail(x2 * x2, - 1)
delta2 <- theta_prime_s2 *  tail(W3 %*%  delta3, - 1)
theta_prime_s1 <- 1 - tail(x1 * x1, - 1)
delta1 <- theta_prime_s1 *  tail(W2 %*%  delta2, - 1)

print(delta3)
```

```
##              [,1]
## [1,] -0.03284662
```

```
print(delta2)
```

```
##             [,1]
## [2,] -0.0155319
```

```
print(delta1)
```

```
##              [,1]
## [2,] -0.01221503
## [3,]  0.03315646
```

4. What are the values of $\frac{\partial e}{\partial W^{(1)}}$, $\frac{\partial e}{\partial W^{(2)}}$ and $\frac{\partial e}{\partial W^{(3)}}$?

```
gW1 <- x0 %*% t(delta1)
gW2 <- x1 %*% t(delta2)
gW3 <- x2 %*% t(delta3)

print(gW1)
```

```
##              [2,]        [3,]
## [1,] -0.01221503  0.03315646
## [2,]  0.02443005 -0.06631292
```

```
print(gW2)
```

```
##               [2,]
## [1,] -0.015531899
## [2,]  0.007177557
## [3,]  0.008341400
```

```
print(gW3)
```

```
##              [,1]
## [1,] -0.03284662
## [2,] -0.02870218
```

5. Repeat the computations for the case when the output transformation is the identity.

Replace the following lines in the previous code

```
x3 <- s3
delta3 <- 2 * (x3 - y)
```