Part 1 Option B - Handwritten Digit Recognition with Neural Networks

UnderGrad Team

```
knitr::opts_chunk$set(echo = TRUE)

library(ggthemes)
library(keras)
library(R.matlab)
library(tidyverse)

set.seed(42)
```

Part 7

See https://tensorflow.rstudio.com/guide/keras/ for documentation.

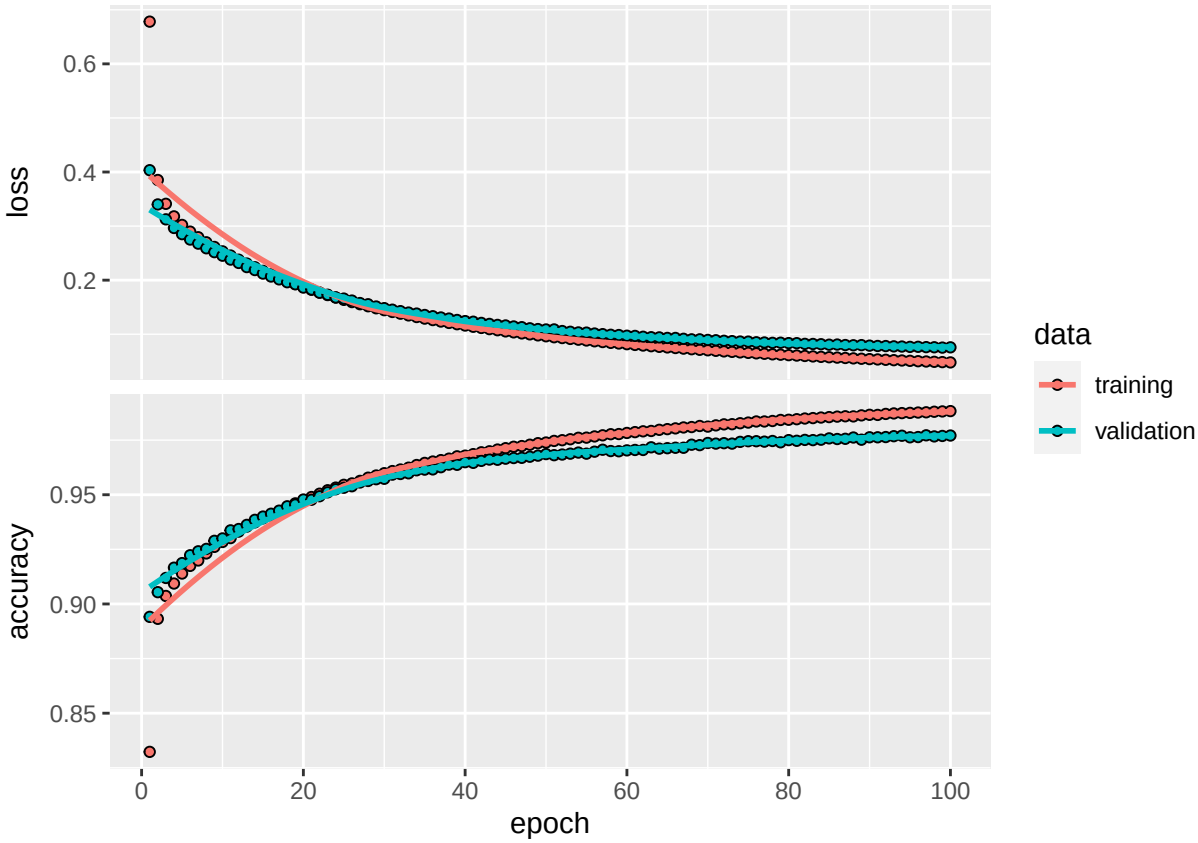
```
# Load data
mnist <- readMat('mnist_all.mat')
data_train <- data.frame()
data_test <- data.frame()

for (i in 0:9) {
  train_digit <- mnist[paste0('train', i)][[1]] %>% data.frame
  train_digit['Y'] <- i
  data_train <- rbind(train_digit, data_train)

  test_digit <- mnist[paste0('test', i)][[1]] %>% data.frame
  test_digit['Y'] <- i
  data_test <- rbind(test_digit, data_test)
}

# Shuffle training dataset
data_train <- data_train[sample(nrow(data_train)), ]

# Split into X and Y
X_train <- data_train %>% select(-Y) %>% as.matrix()
Y_train <- data_train$Y
X_test <- data_test %>% select(-Y) %>% as.matrix()
Y_test <- data_test$Y

# Scale by 255
X_train <- X_train / 255.0
X_test <- X_test / 255.0
```

```r
# Convert Y to categorical
Y_train <- to_categorical(Y_train)
Y_test <- to_categorical(Y_test)
```

```r
# Create model
model <- keras_model_sequential()
model %>%
  layer_dense(units = 300, activation = 'tanh', input_shape = c(ncol(X_train))) %>%
  layer_dense(units = 10, activation = 'softmax')

model %>% compile(
  loss = 'categorical_crossentropy',
  optimizer = optimizer_sgd(lr = 0.01),
  metrics = c('accuracy')
)

summary(model)
```

```
## Model: "sequential"
## _________________________________________________________________________
## Layer (type)                 Output Shape              Param #
## =========================================================================
## dense_1 (Dense)              (None, 300)               235500
## _________________________________________________________________________
## dense (Dense)                (None, 10)                3010
## =========================================================================
## Total params: 238,510
## Trainable params: 238,510
## Non-trainable params: 0
## _________________________________________________________________________
```

```r
# Train model
history <- model %>% fit(
  X_train, Y_train,
  epochs = 100, batch_size = 50,
  validation_data = list(X_test, Y_test))

plot(history)
```


loss
0.6
0.4
0.2
accuracy
0.95
0.90
0.85
0
20
40
60
80
100
epoch
data
training
validation