Homework 5

I’m going to give instructions for completing this homework in R. If you want to use python, that’s fine but you’ll have to set everything up!

For starters, you’ll need to install keras and tensorflow. **Note: We are going to use keras not keras3!** You can find basic instructions on installing keras and tensofrlow at [this site website](https://tensorflow.rstudio.com/install/).

Now I was unable to get this to install on my desktop or laptop, likely due to my machines having weird Windows profiles required by our department.

As an alternative, we can use a docker container to run keras and RStudio. If you can’t the installation to Work, I’m posting a video on starting up a docker container where you can fit the models. Check that out on the Moodle site. It isn’t actually very painful :)

# Implementation Problems

We’ll fit two basic models from the notes in this section! This is essentially a proof of concept homework. Just see that you can get the code below to run and save the graphs and things that come out (see below for details).

Let’s fit models to the digits data (dataset\_mnist()).

1. Load in the library and prep the data. Run the following code:

library(keras)  
mnist <- dataset\_mnist()  
  
set.seed(1001)  
# Prep training set  
train\_images <- mnist$train$x %>%  
 array\_reshape(c(60000, 28 \* 28))  
train\_images <- train\_images / 255  
train\_labels <- mnist$train$y %>%  
 to\_categorical(10)  
# Prep test set  
test\_images <- mnist$test$x %>%  
 array\_reshape(c(10000, 28 \* 28))  
test\_images <- test\_images / 255  
test\_labels <- mnist$test$y %>%  
 to\_categorical(10)

## First Model

1. Great, now let’s set up a basic two layer model as done in the notes. Run the following code:

network <- keras\_model\_sequential() %>%  
 layer\_dense(units = 512, activation = "relu",   
 input\_shape = c(28\*28)) %>%  
 layer\_dense(units = 10, activation = "softmax")  
  
network

Either copy the output from network or have it print out (if you are using R Markdown or quarto).

1. Let’s compile and train the model. Run the following code!

network %>% compile(  
 optimizer = optimizer\_rmsprop(),  
 loss = "categorical\_crossentropy",  
 metrics = c("accuracy")  
)  
## training  
history <- network %>%   
 fit(train\_images, train\_labels,   
 epochs = 15, batch\_size = 128,  
 validation\_split = 0.2)  
  
plot(history)

Include these plots in your final document. If you are using python, I’m not sure how to create the plots so you’ll need to figure that out.

1. Lastly, let’s get a confusion matrix and overall accuracy. Run the following code:

pred <- predict\_classes(network, test\_images)  
conf\_mat <- table(as.factor(pred), as.factor(mnist$test$y))  
conf\_mat  
sum(diag(conf\_mat))/sum(conf\_mat)

The output should be included in your final document.

## Second Model

1. Now let’s fit the multinomial logistic regression model from the notes. Run the following code:

set.seed(1001)  
## Multinomial logit regression  
mlogit <- keras\_model\_sequential() %>%  
 layer\_dense(input\_shape = 28\*28,   
 units = 10, activation = "softmax")  
  
## compile network  
mlogit %>% compile(  
 optimizer = optimizer\_rmsprop(),  
 loss = "categorical\_crossentropy",  
 metrics = c("accuracy")  
)  
## training  
history <- mlogit %>%   
 fit(train\_images, train\_labels,   
 epochs = 15, batch\_size = 128,  
 validation\_split = 0.2)  
  
plot(history)

Save the plots into your final document.

1. Now let’s produce our confusion matrix and overall accuracy on the test set.

## prediction  
pred <- predict\_classes(mlogit, test\_images)  
conf\_mat\_mlogit <- table(as.factor(pred), as.factor(mnist$test$y))  
conf\_mat\_mlogit  
sum(diag(conf\_mat\_mlogit))/sum(conf\_mat\_mlogit)

Show this output in your final document as well.

That’s it! Good luck and let me know if you run into issues.