Final Report

Sabir Meah, Michael Miller, Liyang Yuan

2022-12-16

Introduction

Alzheimer's disease is

We decided to use an Alzheimer's MRI imaging dataset from Kaggle. The data includes a total of 6400 MRI images from individuals classified into four stages: no dementia, very mild dementia, mild dementia, and moderate dementia. Each MRI result is a 128x128 .jpg image. The images have been preprocessed so that they are of a standard form and orientation, but we still will need to do the step of processing the data and gray scale standardization from .jpg images into a tabular format suitable for standard R functions.

Data Read-In

We read in the 128x128 images from .jpg files to three dimensional arrays, with the first element coresponding to the index

```
no_dementia <- array(NA, dim = c(3200, 128, 128))
for (i in 1:3200){
  no_dementia[i,,] <- readJPEG(paste0(getwd(),
                                      "/Data/Non_Demented/non_", i, ".jpg"))
}
```

```
verymild_dementia <- array(NA, dim = c(2240, 128, 128))
for (i in 1:2240){
  verymild_dementia[i,,] <- readJPEG(paste0(getwd(),
                                            "/Data/Very_Mild_Demented/verymild_", i, ".jpg"))
}
```

```
mild_dementia <- array(NA, dim = c(896, 128, 128))
for (i in 1:896){
  mild_dementia[i,,] <- readJPEG(paste0(getwd(),
                                        "/Data/Mild_Demented/mild_", i, ".jpg"))
}
```

```
moderate_dementia <- array(NA, dim = c(64, 128, 128))
for (i in 1:64){
  moderate_dementia[i,,] <- readJPEG(paste0(getwd(),
                                            "/Data/Moderate_Demented/moderate_", i, ".jpg"))
}
```

```
lenNoDem <- 3200
lenVeryMildDem <- 2240
lenMildDem <- 896
lenModDem <- 64
totalLen <- lenNoDem + lenVeryMildDem + lenMildDem + lenModDem
```

```
class <- c(rep("No Dementia", lenNoDem),
           rep("Very Mild Dementia", lenVeryMildDem),
           rep("Mild Dementia", lenMildDem),
           rep("Moderate Dementia", lenModDem))
```

```
#dementia_data <- list(no_dementia, verymild_dementia, mild_dementia, moderate_dementia)
dementia_data <- array(NA, dim = c(6400, 128, 128))
dementia_data[1:lenNoDem,,] <- no_dementia
dementia_data[3201:(lenNoDem + lenVeryMildDem),,] <- verymild_dementia
dementia_data[5441:(lenNoDem + lenVeryMildDem + lenMildDem),,] <- mild_dementia
dementia_data[6337:totalLen,,] <- moderate_dementia
```

Model 1: 3 classes with one hot encoding

```
lenNoDem <- 3200
lenVeryMildDem <- 2240
lenMildDem <- 896
lenModDem <- 64

img_rows <- 128
img_cols <- 128

totalLen <- lenNoDem+lenVeryMildDem+lenMildDem+lenModDem

noDemClass <- cbind(rep(1,lenNoDem), rep(0,lenNoDem),
                    rep(0,lenNoDem))
veryMildDemClass <- cbind(rep(0,lenVeryMildDem), rep(1,lenVeryMildDem),
                          rep(0,lenVeryMildDem))
mildModDemClass <- cbind(rep(0,lenMildDem+lenModDem), rep(0,lenMildDem+lenModDem),
                         rep(1,lenMildDem+lenModDem))
classMat <- rbind(noDemClass,
                  veryMildDemClass,
                  mildModDemClass)

noDemTestIdx <- sample(lenNoDem, round(lenNoDem/4.0), replace = FALSE)
noDemTrainIdx <- setdiff(1:lenNoDem, noDemTestIdx)

veryMildDemTestIdx <- sample((lenNoDem+1):(lenNoDem+lenVeryMildDem),
                             round(lenVeryMildDem/4.0), replace = FALSE)
veryMildDemTrainIdx <- setdiff((lenNoDem+1):(lenNoDem+lenVeryMildDem),
                             veryMildDemTestIdx)

mildModDemTestIdx <- sample((lenNoDem+lenVeryMildDem+1):
                          (lenNoDem+lenVeryMildDem+lenMildDem+lenModDem),
                          round((lenMildDem+lenModDem)/4.0), replace = FALSE)
mildModDemTrainIdx <- setdiff((lenNoDem+lenVeryMildDem+1):
                          (lenNoDem+lenVeryMildDem+lenMildDem+lenModDem),
                          mildModDemTestIdx)

x_train <- dementia_data[c(noDemTrainIdx,veryMildDemTrainIdx,
                           mildModDemTrainIdx),,]
y_train <- classMat[c(noDemTrainIdx,veryMildDemTrainIdx,
```

```
                        mildModDemTrainIdx),]
x_test <- dementia_data[c(noDemTestIdx,veryMildDemTestIdx,
                          mildModDemTestIdx),,]
y_test <- classMat[c(noDemTestIdx,veryMildDemTestIdx,
                     mildModDemTestIdx),]

shuffleIdx <- sample(1:round(3.0*totalLen/4.0))
x_train <- x_train[shuffleIdx,,]
y_train <- y_train[shuffleIdx,]

x_train <- array_reshape(x_train, c(nrow(x_train), img_rows, img_cols, 1))
x_test <- array_reshape(x_test, c(nrow(x_test), img_rows, img_cols, 1))
input_shape <- c(img_rows, img_cols, 1)

batch_size <- 32
num_classes <- 3
epochs <- 100
```

```
mod1 <- keras_model_sequential() %>%
  layer_conv_2d(filters = 64, kernel_size = c(3,3),
                activation = 'relu', input_shape = input_shape) %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_dropout(rate = 0.25) %>%
  layer_conv_2d(filters = 64, kernel_size = c(3,3),
                activation = 'relu', input_shape = input_shape) %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_dropout(rate = 0.25) %>%
  layer_flatten() %>%
  layer_dense(units = 128, activation = 'relu',
              kernel_regularizer=regularizer_l1_l2(l1=1e-4,l2=1e-5)) %>%
  layer_dropout(rate = 0.25) %>%
  layer_dense(units = num_classes,
              activation = 'softmax')

mod1 %>% compile(
  loss = loss_categorical_crossentropy,
  optimizer = optimizer_adam(),
  metrics = c('accuracy')
)

cnn_history <- mod1 %>% fit(
  x_train, y_train,
  batch_size = batch_size,
  epochs = epochs,
  validation_split = 0.2
)
```

```
# Predict on the test data
predictProbs = predict(mod1, x_test)
```

```
saveRDS(predictProbs, file = "predict_mod1.Rds")
```

Process Model 1 Results

```
predictProbs = readRDS("predict_mod1.Rds")
```

```
predictClass <- matrix(NA, nrow = nrow(x_test), ncol = num_classes)
for (i in 1:nrow(x_test)) {
  classVec <- rep(0, num_classes)
  classVec[which(predictProbs[i,] == max(predictProbs[i,]))] <- 1
  predictClass[i,] <- classVec
}

testAccuracyNoDem <- 0.0
testAccuracyVeryMildDem <- 0.0
testAccuracyMildModDem <- 0.0

diffMat <- y_test - predictClass

for (i in 1:length(noDemTestIdx)) {
  testAccuracyNoDem <- testAccuracyNoDem+ifelse(min(diffMat[i,]) == 0,1,0)
}
for (i in 1:length(veryMildDemTestIdx)) {
  testAccuracyVeryMildDem <- testAccuracyVeryMildDem +
    ifelse(min(diffMat[length(noDemTestIdx)+i,]) == 0,1,0)
}
for (i in 1:length(mildModDemTestIdx)) {
  testAccuracyMildModDem <- testAccuracyMildModDem +
    ifelse(min(diffMat[length(noDemTestIdx)+length(veryMildDemTestIdx)+i,]) == 0,1,0)
}

testAccuracyNoDem <- testAccuracyNoDem/length(noDemTestIdx)
testAccuracyVeryMildDem <- testAccuracyVeryMildDem/length(veryMildDemTestIdx)
testAccuracyMildModDem <- testAccuracyMildModDem/length(mildModDemTestIdx)

resModel1 <- data.frame("No Dementia" = testAccuracyNoDem,
                        "Very Mild Dementia" = testAccuracyVeryMildDem,
                        "Mild/Moderate Dementia" = testAccuracyMildModDem)
knitr::kable(resModel1)
```

No.Dementia	Very.Mild.Dementia	Mild.Moderate.Dementia
0.975	0.9214286	0.9625

Model 2: 4 classes with one hot encoding

```
lenNoDem <- 3200
lenVeryMildDem <- 2240
lenMildDem <- 896
lenModDem <- 64

img_rows <- 128
img_cols <- 128

totalLen <- lenNoDem+lenVeryMildDem+lenMildDem+lenModDem
```

```
noDemClass <- cbind(rep(1,lenNoDem), rep(0,lenNoDem),
                    rep(0,lenNoDem), rep(0,lenNoDem))
veryMildDemClass <- cbind(rep(0,lenVeryMildDem), rep(1,lenVeryMildDem),
                          rep(0,lenVeryMildDem), rep(0,lenVeryMildDem))
mildDemClass <- cbind(rep(0,lenMildDem) ,rep(0,lenMildDem),
                      rep(1,lenMildDem), rep(0,lenMildDem))
modDemClass <- cbind(rep(0,lenModDem), rep(0,lenModDem),
                     rep(0,lenModDem), rep(1,lenModDem))
classMat <- rbind(noDemClass, veryMildDemClass,
                  mildDemClass, modDemClass)
```

```
# Predict on the test data
predictProbs = predict(mod2, x_test)
```

```
saveRDS(predictProbs, file = "predict_mod2.Rds")
```

Process Model 2 Results

```
predictProbs = readRDS("predict_mod2.Rds")
```

```
predictClass <- matrix(NA, nrow = nrow(x_test), ncol = num_classes)
for (i in 1:nrow(x_test)) {
  classVec <- rep(0, num_classes)
  classVec[which(predictProbs[i,] == max(predictProbs[i,]))] <- 1
  predictClass[i,] <- classVec
}

testAccuracyNoDem <- 0.0
testAccuracyVeryMildDem <- 0.0
testAccuracyMildDem <- 0.0
testAccuracyModDem <- 0.0

diffMat <- y_test - predictClass

for (i in 1:length(noDemTestIdx)) {
  testAccuracyNoDem <- testAccuracyNoDem+ifelse(min(diffMat[i,]) == 0,1,0)
}
for (i in 1:length(veryMildDemTestIdx)) {
  testAccuracyVeryMildDem <- testAccuracyVeryMildDem +
    ifelse(min(diffMat[length(noDemTestIdx)+i,]) == 0,1,0)
}
for (i in 1:length(mildDemTestIdx)) {
  testAccuracyMildDem <- testAccuracyMildDem +
    ifelse(min(diffMat[length(noDemTestIdx) +
                       length(veryMildDemTestIdx)+i,]) == 0,1,0)
}
for (i in 1:length(modDemTestIdx)) {
  testAccuracyModDem <- testAccuracyModDem +
    ifelse(min(diffMat[length(noDemTestIdx)+length(veryMildDemTestIdx)+
                       length(mildDemTestIdx)+i,]) == 0,1,0)
}
```

```
testAccuracyNoDem <- testAccuracyNoDem/length(noDemTestIdx)
testAccuracyVeryMildDem <- testAccuracyVeryMildDem/length(veryMildDemTestIdx)
testAccuracyMildDem <- testAccuracyMildDem/length(mildDemTestIdx)
testAccuracyModDem <- testAccuracyModDem/length(modDemTestIdx)

resModel2 <- data.frame("No Dementia" = testAccuracyNoDem,
                        "Very Mild Dementia" = testAccuracyVeryMildDem,
                        "Mild Dementia" = testAccuracyMildDem,
                        "Moderate Dementia" = testAccuracyModDem)
knitr::kable(resModel2)
```

No.Dementia	Very.Mild.Dementia	Mild.Dementia	Moderate.Dementia
0.97875	0.9785714	0.9508929	0.875