IMDB

Chujun\_Huang

3/2/2020

library(keras)  
  
imdb\_dir <- "C:/Users/Annie/Desktop/aclImdb"  
train\_dir <- file.path(imdb\_dir, "train")  
labels <- c()  
texts <- c()  
for (label\_type in c("neg", "pos")) {  
 label <- switch(label\_type, neg = 0, pos = 1)  
 dir\_name <- file.path(train\_dir, label\_type)  
 for (fname in list.files(dir\_name, pattern = glob2rx("\*.txt"),  
 full.names = TRUE)) {  
 texts <- c(texts, readChar(fname, file.info(fname)$size))  
 labels <- c(labels, label)  
 }  
}  
  
# cut-off reviews after 150 words  
maxlen <- 150  
# restricting the training\_data to first 100 samples  
training\_samples <- 100  
# validates on 10000 samples  
validation\_samples <- 10000  
# considering only top 10,000 words in the dataset  
max\_words <- 10000  
  
# tokenizing the words  
tokenizer <- text\_tokenizer(num\_words = max\_words) %>%  
 fit\_text\_tokenizer(texts)  
sequences <- texts\_to\_sequences(tokenizer, texts)  
word\_index = tokenizer$word\_index  
  
# Turns the list of integers into a 2D integer tensor shape (samples,maxlen)  
data <- pad\_sequences(sequences, maxlen = maxlen)  
labels <- as.array(labels)  
cat("Shape of data tensor:", dim(data), "\n")

## Shape of data tensor: 25000 150

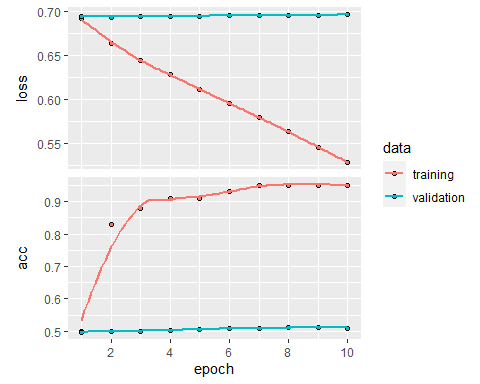
cat('Shape of label tensor:', dim(labels), "\n")

## Shape of label tensor: 25000

set.seed(123)  
indices <- sample(1:nrow(data))  
training\_indices <- indices[1:training\_samples]  
validation\_indices <- indices[(training\_samples + 1):  
 (training\_samples + validation\_samples)]  
  
train\_data <- data[training\_indices,]  
train\_label <- labels[training\_indices]  
valid\_data <- data[validation\_indices,]  
valid\_label<- labels[validation\_indices]  
  
test\_dir <- file.path(imdb\_dir, "test")  
labels <- c()  
texts <- c()  
for (label\_type in c("neg", "pos")) {  
 label <- switch(label\_type, neg = 0, pos = 1)  
 dir\_name <- file.path(test\_dir, label\_type)  
 for (fname in list.files(dir\_name, pattern = glob2rx("\*.txt"),  
 full.names = TRUE)) {  
 texts <- c(texts, readChar(fname, file.info(fname)$size))  
 labels <- c(labels, label)  
 }  
}  
sequences <- texts\_to\_sequences(tokenizer, texts)  
x\_test <- pad\_sequences(sequences, maxlen = maxlen)  
y\_test <- as.array(labels)  
  
# Using an embedding layer and classifier on the IMDB data  
model <- keras\_model\_sequential() %>% layer\_embedding(input\_dim = 10000,output\_dim = 8,input\_length = maxlen) %>%   
 layer\_flatten() %>% layer\_dense(units=1,activation = "sigmoid")  
model %>% compile(optimizer = "rmsprop",loss = "binary\_crossentropy",metrics=c("acc"))

history <- model %>%  
 fit(train\_data,train\_label,epochs=10,batch\_size=32,validation\_data = list(valid\_data,valid\_label))  
# Plot of Accuracy and Loss function of the model  
plot(history)

## `geom\_smooth()` using formula 'y ~ x'



# By observing the plot, the validation accuracy of the model is ~50% considering the first 150 words in every review with 100 samples.  
  
# Evaluating the test dataset   
model %>% fit(  
 train\_data,  
 train\_label,  
 epochs = 2,  
 batch\_size = 32)  
result <- model %>% evaluate(x\_test,y\_test)  
result

## $loss  
## [1] 0.6976663  
##   
## $acc  
## [1] 0.5072

cat("The Test accuracy of the model is ",result$acc)

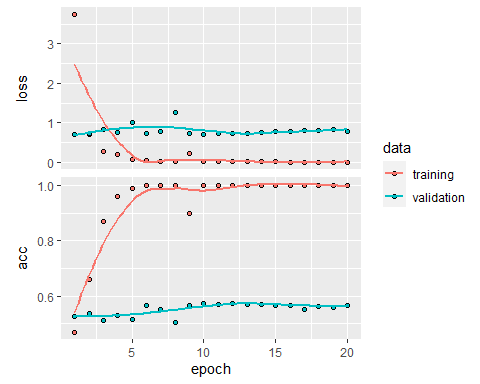
## The Test accuracy of the model is 0.5072

# Parsing the GloVe word-embeddings file  
glove\_dir = "C:/Users/Annie/Desktop/glove.6B"  
lines <- readLines(file.path(glove\_dir, "glove.6B.100d.txt"))  
  
embeddings\_index <- new.env(hash = TRUE, parent = emptyenv())  
for (i in 1:length(lines)) {  
 line <- lines[[i]]  
 values <- strsplit(line, " ")[[1]]  
 word <- values[[1]]  
 embeddings\_index[[word]] <- as.double(values[-1])  
}  
cat("Found", length(embeddings\_index), "word vectors.\n")

## Found 400000 word vectors.

# Preparing the GloVe word-embeddings matrix  
embedding\_dim <- 100  
embedding\_matrix <- array(0, c(max\_words, embedding\_dim))  
for (word in names(word\_index)) {index <- word\_index[[word]]  
if (index < max\_words) {embedding\_vector <- embeddings\_index[[word]]  
if (!is.null(embedding\_vector))embedding\_matrix[index+1,] <- embedding\_vector}}  
  
# Model construction  
model <- keras\_model\_sequential() %>%  
layer\_embedding(input\_dim = max\_words, output\_dim = embedding\_dim,input\_length = maxlen) %>%  
layer\_flatten() %>%  
layer\_dense(units = 32, activation = "relu") %>%  
layer\_dense(units = 1, activation = "sigmoid")  
  
# Loading pretrained word embeddings into the embedding layer  
get\_layer(model, index = 1) %>%  
set\_weights(list(embedding\_matrix)) %>%  
freeze\_weights()  
  
model %>% compile(optimizer = "rmsprop",loss = "binary\_crossentropy",metrics = c("acc"))  
  
history1 <- model %>% fit(train\_data, train\_label,epochs = 20,batch\_size = 32,validation\_data = list(valid\_data , valid\_label))  
plot(history1)

## `geom\_smooth()` using formula 'y ~ x'



# By observing the above plot, the validaition accuracy of the model is ~50% with 100 samples in the training dataset.THe model quickly starts overfitting with small number of traning samples. Hence with having few traning samples, performance is highly dependent on exactly which 100 samples are choosen and choosing at random.   
  
model %>% fit(train\_data, train\_label,epochs = 2,batch\_size = 32)  
  
result1 <- model %>% evaluate(x\_test,y\_test)  
result1

## $loss  
## [1] 0.86043  
##   
## $acc  
## [1] 0.55388

cat("The Test accuracy of the model is ",result1$acc)

## The Test accuracy of the model is 0.55388