IMDB_RNN

smandumu

3/3/2020

```
library(keras)
library(reticulate)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
imdb_dir <- "D:/MSBA/Adv ML/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)
  }
}

maxlen <- 150                 # We will cut reviews after 150 words
training_samples <- 100       # We will be training on 200 samples
validation_samples <- 10000   # We will be validating on 10000 samples
max_words <- 10000            # We will only consider the top 10,000 words in the dataset


# 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
```

```
#Shape of data tensor: 25000 150
cat('Shape of label tensor:', dim(labels), "\n")
```

```
## Shape of label tensor: 25000
```

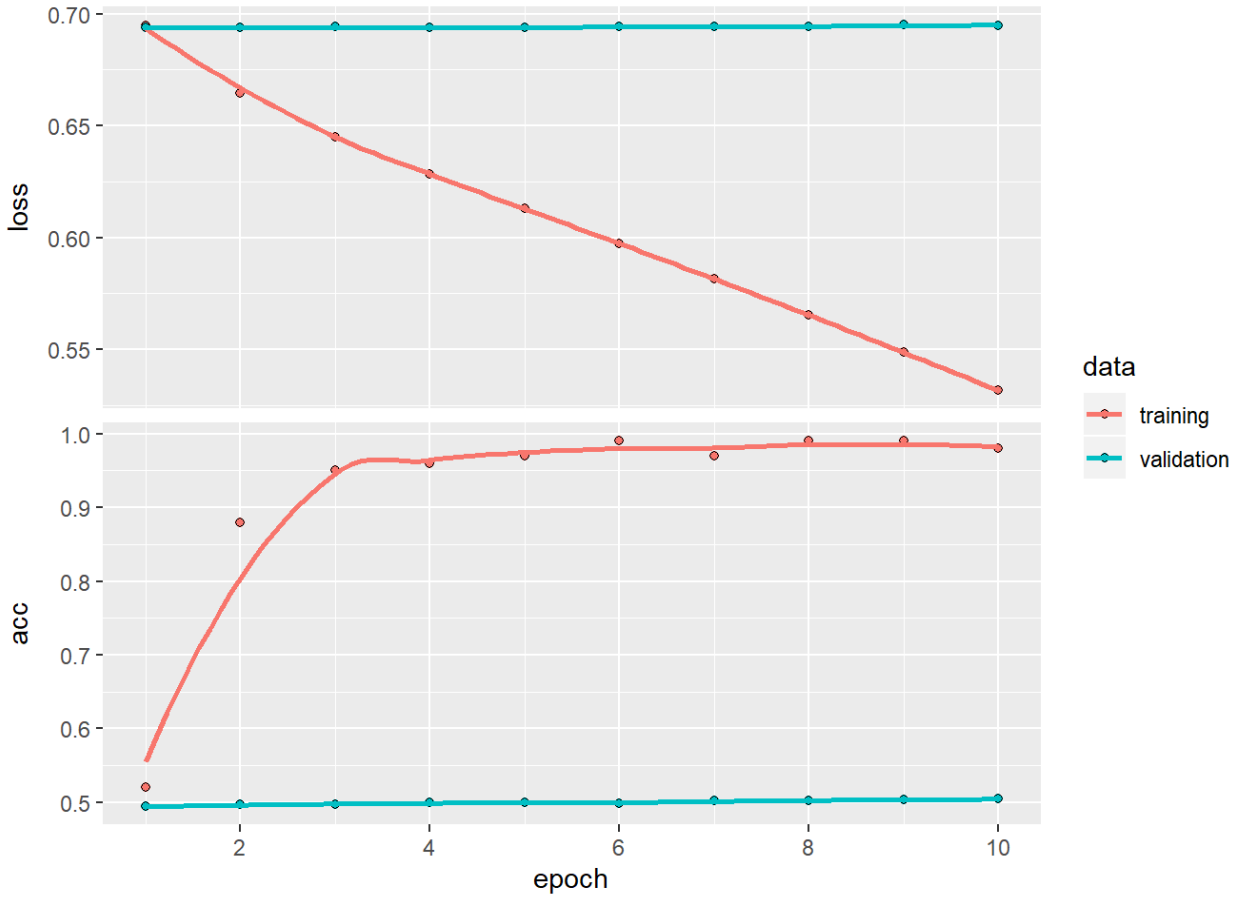
```
#Shape of label tensor: 25000
set.seed(123)
indices <- sample(1:nrow(data))
training_indices <- indices[1:training_samples]
validation_indices <- indices[(training_samples + 1):
                              (training_samples + validation_samples)]

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)
```



From the above plot we can see that the model is performing with ~50% accuracy with 100 samples in the training data While the Validation dataset has 1000 samples.

```
# validation accuracy

# 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] 0.69567
##
## $acc
## [1] 0.5
```

```
#The Test accuracy of the model is  0.50

# Parsing the GloVe word-embeddings file
glove_dir = 'D:/MSBA/Adv ML/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.
```

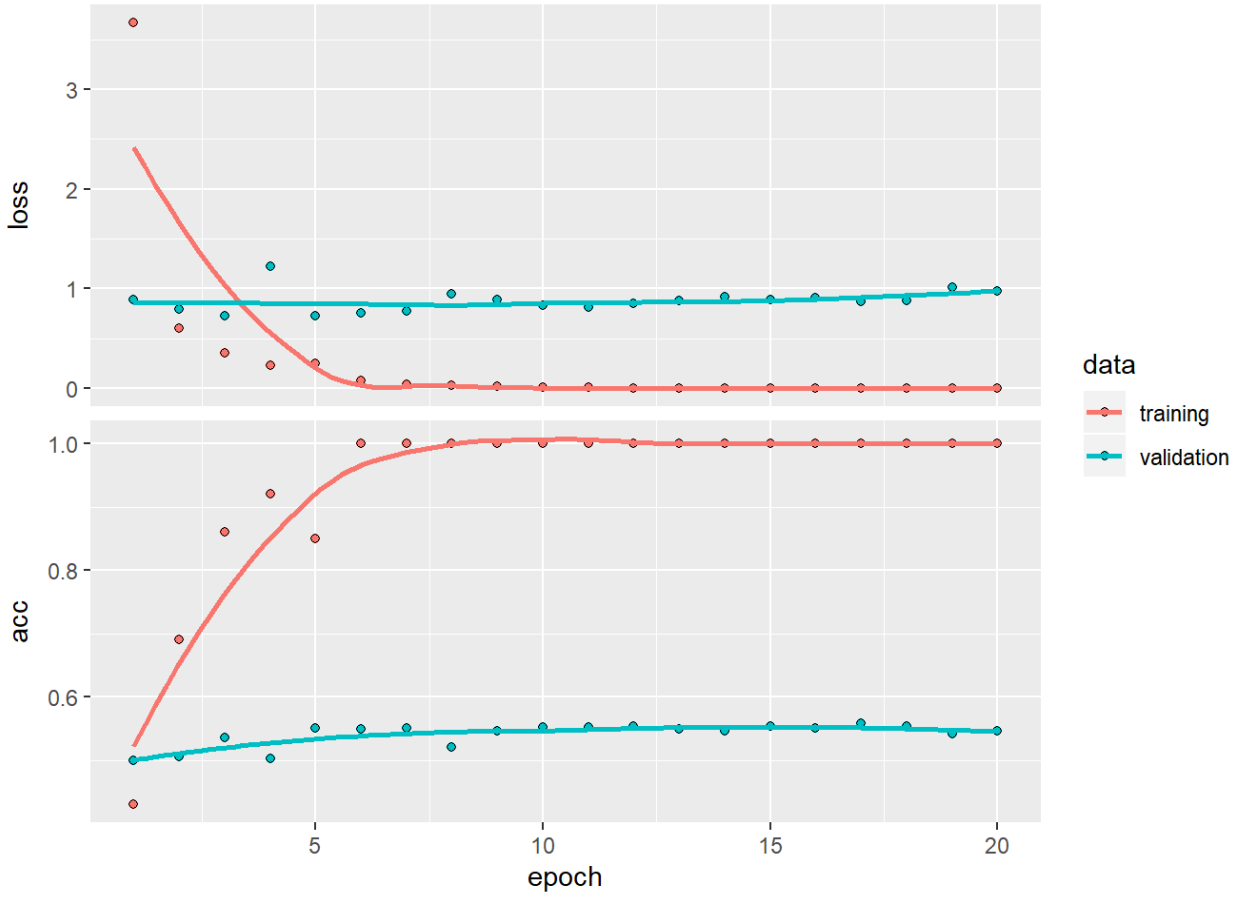
```
# Preparing the GloVe word-embeddings matrix
embedding_dim <- 100
embedding_matrix <- array(0, c(max_words, embedding_dim))
for (word in names(word_index)) {
  index <- word_index[[word]]
  if (index < max_words) {
    embedding_vector <- embeddings_index[[word]]
    if (!is.null(embedding_vector))
      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)
```



Trying to improve the model performance using pretrained networks but we observe that model still performs with ~50% accuracy that is due to less number of samples in training data.As it does not see more patterns in the data.

```
# validation accuracy is 55.81 by cutting reviews for first 150 words in every 100 samples


model %>% fit(
  train_data, train_label,
  epochs = 2,
  batch_size = 32)
result1 <- model %>% evaluate(x_test,y_test)
result1 # Test Accuracy of the model is 55%
```

```
## $loss
## [1] 0.9918165
##
## $acc
## [1] 0.54552
```