Model_CNN

YU Hong

2018/12/10

Model CNN

Data Preparation

```
library(keras)
batch_size <- 128
num_classes <- 10
epochs <- 15
```

Input image dimensions

```
img_rows <- 28
img_cols <- 28
```

The data, shuffled and split between train and test sets

```
mnist <- dataset_mnist()
x_train <- mnist$train$x
y_train <- mnist$train$y
x_test <- mnist$test$x
y_test <- mnist$test$y
```

Redefine dimension of train/test inputs

```
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)
```

Transform RGB values into [0,1] range

```
x_train <- x_train / 255
x_test <- x_test / 255
cat('x_train_shape:', dim(x_train), '\n')
```

```
## x_train_shape: 60000 28 28 1
```

```
cat(nrow(x_train), 'train samples\n')
```

```
## 60000 train samples
```

```
cat(nrow(x_test), 'test samples\n')
```

```
## 10000 test samples
```

Convert class vectors to binary class matrices

```
y_train <- to_categorical(y_train, num_classes)
y_test <- to_categorical(y_test, num_classes)
```

Define Model

```
model <- keras_model_sequential() %>%
  layer_conv_2d(filters = 32, kernel_size = c(3,3), activation = 'relu',
                input_shape = input_shape) %>%
  layer_conv_2d(filters = 64, kernel_size = c(3,3), activation = 'relu') %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_dropout(rate = 0.25) %>%
  layer_flatten() %>%
  layer_dense(units = 128, activation = 'relu') %>%
  layer_dropout(rate = 0.5) %>%
  layer_dense(units = num_classes, activation = 'softmax')
summary(model)
```

```
## ___________________________________________________________________________
## Layer (type)                     Output Shape                  Param #
## ===========================================================================
## conv2d_1 (Conv2D)                (None, 26, 26, 32)            320
## ___________________________________________________________________________
## conv2d_2 (Conv2D)                (None, 24, 24, 64)            18496
## ___________________________________________________________________________
## max_pooling2d_1 (MaxPooling2D)   (None, 12, 12, 64)            0
## ___________________________________________________________________________
## dropout_1 (Dropout)              (None, 12, 12, 64)            0
## ___________________________________________________________________________
## flatten_1 (Flatten)              (None, 9216)                  0
## ___________________________________________________________________________
## dense_1 (Dense)                  (None, 128)                   1179776
## ___________________________________________________________________________
## dropout_2 (Dropout)              (None, 128)                   0
## ___________________________________________________________________________
## dense_2 (Dense)                  (None, 10)                    1290
## ===========================================================================
## Total params: 1,199,882
## Trainable params: 1,199,882
## Non-trainable params: 0
## ___________________________________________________________________________
```

Compile model

```
model %>% compile(
  loss = loss_categorical_crossentropy,
  optimizer = optimizer_adadelta(),
  metrics = c('accuracy')
)
```

Train model

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

Result

```
scores <- model %>% evaluate(
  x_test, y_test, verbose = 0
)
scores
```

```
## $loss
## [1] 0.03093762
##
## $acc
## [1] 0.9913
```