Project 3 Group 1

Wanting Cui

```
if(!require("xgboost")){
  install.packages("xgboost")
}
```

```
## Loading required package: xgboost
```

```
## Warning: package 'xgboost' was built under R version 3.4.4
```

```
if(!require("ggplot2")){
  install.packages("ggplot2")
}
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.4.2
```

```
if(!require("reshape2")){
  install.packages("reshape2")
}
```

```
## Loading required package: reshape2
```

```
## Warning: package 'reshape2' was built under R version 3.4.3
```

```
library(xgboost)
library(ggplot2)
library(reshape2)

source("../lib/xgboost_cv.R")
```

Step 0: specify directories.

Provide directory for extracted features.

```
feat.dir <- "../output/features/"
```

Step 1: set up controls for evaluation experiments.

In this chunk, ,we have a set of controls for the evaluation experiments.

- (T/F) cross-validation on the training set
- (number) K, the number of CV folds
- (T/F) run evaluation on an independent test set

```
run.cv=FALSE # run cross-validation on the training set
K <- 5  # number of CV folds
run.test=TRUE # run evaluation on an independent test set
```

Step 2: import training features and labels.

```
sift2 <- read.csv(paste0(feat.dir, "SIFT_train1.csv"), header = TRUE)
load(paste0(feat.dir, "hog_train1.RData"))
lbp2 <- read.csv(paste0(feat.dir, "lbp_train1.csv"), header = TRUE)
train2 <- cbind(sift2, hog2, lbp2)
train2 <- data.matrix(train2)

lab_tr <- read.csv(paste0(feat.dir, "label_train1.csv"))
lab_tr <- data.matrix(lab_tr)
```

Step 3: Train a classification model with training images

Model selection with cross-validation

- Do model selection by choosing among different values of training model parameters. Max.depth (depth of each tree), eta (shrinkage), nrounds.

```
if(run.cv){
  shl_tr <- xgboost_cv(train = train2, lab = lab_tr, nrou = 300,
                       list_max.depth = c(3, 5, 10, 20),
                       list_eta = c(0.03, 0.3, 0.5, 0.8),
                       name = "SIFT + HOG + lbp", fold = K)
  err.cv1 <- shl_tr[[1]]
  save(err.cv1, file = "../output/XGBOOST_results/err_shl.RData")


  shl_tr2 <- xgboost_cv(train = train2, lab = lab_tr, nrou = 200,
                       list_max.depth = c(2, 3, 4),
                       list_eta = c(0.4, 0.5),
                       name = "SIFT + HOG + lbp 2", fold = K)
  err.cv2 <- shl_tr2[[1]]
  save(err.cv2, file = "../output/XGBOOST_results/err_shl2.RData")

}
```

Visualize cross-validation results.

```
if(run.cv){
   print(shl_tr[[2]])

  jpeg("../output/XGBOOST_results/XGBOOST & SIFT + HOG + lbp0.jpeg")
  plot(shl_tr[[2]])
  dev.off

  print(shl_tr2[[2]])

  jpeg("../output/XGBOOST_results/XGBOOST & SIFT + HOG + lbp2.jpeg")
  plot(shl_tr2[[2]])
  dev.off

}
```

- Train the model with the entire training set using the selected model (model parameter) via cross-validation.

```
tm_train=NA
param <- list("objective" = "multi:softmax",
                    "num_class" = 4,
                    "eta" = 0.5, "max.depth" = 4)
tm_train <- system.time(bst <- xgboost(data = train2, label = lab_tr, params = param, nrounds = 80, verb
save(bst, file="../output/XGBOOST_results/bst.RData")
```

Step 5: Make prediction

Feed the final training model with the completely holdout testing data.

```
tm_test=NA
if(run.test){
  sift1 <- read.csv(paste0(feat.dir, "SIFT_test1.csv"))
  load(paste0(feat.dir, "hog_test1.RData"))
  lbp1 <- read.csv(paste0(feat.dir, "lbp_test1.csv"))
  test <- cbind(sift1, hog1, lbp1)
  test <- data.matrix(test)

  tm_test <- system.time(pred <- predict(bst, test))
  save(pred, file="../output/XGBOOST_results/pred_test.RData")

  lab_te <- read.csv(paste0(feat.dir, "label_test1.csv"))
  mean(pred != lab_te)
}
```

```
## [1] 0.09333333
```

Summarize Running Time

Prediction performance matters, so does the running times for constructing features and for training the model, especially when the computation resource is limited.

```
cat("Time for training model=", tm_train[3], "s \n")
```

```
## Time for training model= 25.68 s
```

```
cat("Time for making prediction=", tm_test[3], "s \n")
```

```
## Time for making prediction= 0.05 s
```