Project 3

Nikita Bondarenko, Yue Gao, Ruochen Liu, Xuehan Liu, Xiaowo Sun

March 24, 2017

Group Member: Nikita Bondarenko, Yue Gao, Ruochen Liu, Xuehan Liu, Xiaowo Sun

Instructions: When testing new data, put new images in test folder and put sift_features_test.csv in data.

Step 0: load libraries and specify directories

```
#install.packages("caret")
#install.packages("gbm")
#install.packages("randomForest")
#install.packages("plyr")
#install.packages("xgboost")
#install.packages("fastAdaboost")
#install.packages("deepboost")
#install.packages("EBImage")
#install.packages("e1071")
#install.packages("kernlab")
#install.packages("OpenImageR")

library("caret")
```

```
## Loading required package: lattice
```

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

```
library("gbm")
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

```
## Loading required package: splines
```

```
## Loading required package: parallel
```

```
## Loaded gbm 2.1.1
```

```
library("randomForest")
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
```

```
##     margin
```

```r
library("plyr")
library("xgboost")
library("fastAdaboost")
library("deepboost")
```

```
##
## Attaching package: 'deepboost'

## The following object is masked from 'package:survival':
##
##     heart
```

```r
library("EBImage")
library("e1071")
library("kernlab")
```

```
##
## Attaching package: 'kernlab'

## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```r
library("OpenImageR")
```

```
##
## Attaching package: 'OpenImageR'

## The following objects are masked from 'package:EBImage':
##
##     readImage, writeImage
```

```r
setwd("~/GitHub/spr2017-proj3-group7/doc/") # Here replace it with your own path or manually set it in
source("../lib/feature.R")
source("../lib/train.R")
source("../lib/test.R")
```

Provide directories for raw images. Training set and test set should be in different subfolders.

```r
experiment_dir <- "../data/" # This will be modified for different data sets.
img_train_dir <- paste(experiment_dir, "raw_images/", sep="")
img_test_dir <- paste(experiment_dir, "test/", sep="")
```

Step 1: Summary of trained models

baseline model

GBM + 5000SIFT

```r
load("../output/baseline.RData")
print(baseline$bestTune)
```

```
##    n.trees interaction.depth shrinkage n.minobsinnode
## 50     250                10       0.1             10
```

```
print(mean(baseline$finalModel$train.error))
```

```
## [1] 0.3003612
```

other models that we have tried using HOG features

Test Errors: GBM: 0.146 Random Forest:0.142 SVM Linear: 0.122 SVM Radial: 0.118 xgBoost: 0.136

advanced model

Finally we decided to use SVM Linear model because of its low test error and stability.

```
#load SVM here
load("../output/advanced.RData")
print(advanced$bestTune)
```

```
##   C
## 4 2
```

```
print(advanced$finalModel)
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 2
##
## Linear (vanilla) kernel function.
##
## Number of Support Vectors : 507
##
## Objective Function Value : -944.8283
## Training error : 0.132
```

Step 2: Extract new features from test images

For our project, we construct 54 HOG features. Save the constructed features to the output subfolder.

```
feature_sift <- read.csv("../data/sift_features_test.csv")
feature_sift <- t(feature_sift)
tm_feature_test <- system.time(feature_hog <-feature(img_test_dir,export=T))
#load("../output/HOG.RData")
```

Step 3: Make prediction

baseline model

Only feed the baseline training model with SIFT data.

```
tm_test_bs <- system.time(pred_test_bs <- test(baseline, feature_sift))
save(pred_test_bs, file="../output/pred_test_bs.RData")
write.csv(pred_test_bs, file="../output/pred_test_bs.csv")
```

advanced model

Only feed the advanced training model with HOG data.

```r
tm_test_ad <- system.time(pred_test_ad <- test(advanced, feature_hog))
save(pred_test_ad, file="../output/pred_test_ad.RData")
write.csv(pred_test_ad, file="../output/pred_test_ad.csv")
```

Step 4: Summarize Running Time

```r
cat("Time for constructing testing features=", tm_feature_test[1], "s \n")
```

```
## Time for constructing testing features= 63.44 s
```

```r
cat("Time for making prediction=", tm_test_ad[1], "s \n")
```

```
## Time for making prediction= 0.03 s
```

Step 0.5: Train Models

```r
#labels <- read.csv("../data/labels.csv")
#labels <- labels[,1]
#x <- "The feature you choose"
#models <- Train(x, labels) # Return the models and errors.
```