# Final project
# Analysis of Deep Learning Architectures

Harvard CS 109B, Spring 2017

*Submission of: Group 26*

*May 2017*

## Importing Libraries

```
# Libraries

library(gam)          # Additive Models
```

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

```
## Loading required package: foreach
```

```
## Loaded gam 1.14
```

```
library(splines)      # splines fit
library(ggplot2)      # Graphing
library(ggthemes)     # Graphing Themes
library(grid)         # For multiplot
library(gridExtra)    # For multiplot
library(reshape2)     # to reshape dataframes
library(tidyr)        # to combine dataframe
```

```
##
## Attaching package: 'tidyr'
```

```
## The following object is masked from 'package:reshape2':
##
##     smiths
```

## SBuild up data arrays

### The reference Labels from the test set

```
Labels = subset(read.csv("labels.csv"), select=Action:Other)
```

### Function to transform the probabilitites to categories

```
# Define Function
categ = function(filename) {

  # Specify the data range from the file
  item = subset(read.csv(filename) , select = Action:Other)
```

```
  # Create a copy for editing
  clone = item

  # Execute per row
  for (i in 1:nrow(item)){

    # Execute per column
    for (j in 1:ncol(item)){

      # Apply cuttof probability
      if (item[[i,j]] < 0.5) {

        # Modify the array
        clone[[i,j]] = 0

      }

      else {clone[[i,j]] = 1}


    }

  }

  return(clone)

}
```

### Function to perform a multiple plot

```
# Multiple plot function (Source : "Cookbook for R"
#         "http://www.cookbook-r.com/Graphs/Multiple_graphs_on_one_page_(ggplot2)/")
#
# ggplot objects can be passed in ..., or to plotlist (as a list of ggplot objects)
# - cols:   Number of columns in layout
# - layout: A matrix specifying the layout. If present, 'cols' is ignored.
#
# If the layout is something like matrix(c(1,2,3,3), nrow=2, byrow=TRUE),
# then plot 1 will go in the upper left, 2 will go in the upper right, and
# 3 will go all the way across the bottom.
#
multiplot <- function(..., plotlist=NULL, file, cols=1, layout=NULL) {
  library(grid)

  # Make a list from the ... arguments and plotlist
  plots <- c(list(...), plotlist)

  numPlots = length(plots)

  # If layout is NULL, then use 'cols' to determine layout
  if (is.null(layout)) {
    # Make the panel
```

```
    # ncol: Number of columns of plots
    # nrow: Number of rows needed, calculated from # of cols
    layout <- matrix(seq(1, cols * ceiling(numPlots/cols)),
                     ncol = cols, nrow = ceiling(numPlots/cols))
  }

 if (numPlots==1) {
    print(plots[[1]])

  } else {
    # Set up the page
    grid.newpage()
    pushViewport(viewport(layout = grid.layout(nrow(layout), ncol(layout))))

    # Make each plot, in the correct location
    for (i in 1:numPlots) {
      # Get the i,j matrix positions of the regions that contain this subplot
      matchidx <- as.data.frame(which(layout == i, arr.ind = TRUE))

      print(plots[[i]], vp = viewport(layout.pos.row = matchidx$row,
                                      layout.pos.col = matchidx$col))
    }
  }
}
```

### Function tocalculate accuracy based on the categories

```
# Define Function
accuacy = function(categories){

  # Intiate the counter for accuracy
  Count = rep(0 , ncol(categories))

  # Loop over rows
  for(i in 1:nrow(categories)) {

    # Loop over columns
    for (j in 1:ncol(categories)) {

      # Check for similarity
      if (Labels[i, j] == categories[i,j]){

        # Modify the counter
        Count[j] = Count[j] + 1

      }

    }

  }

  return(Count/nrow(Labels))
```

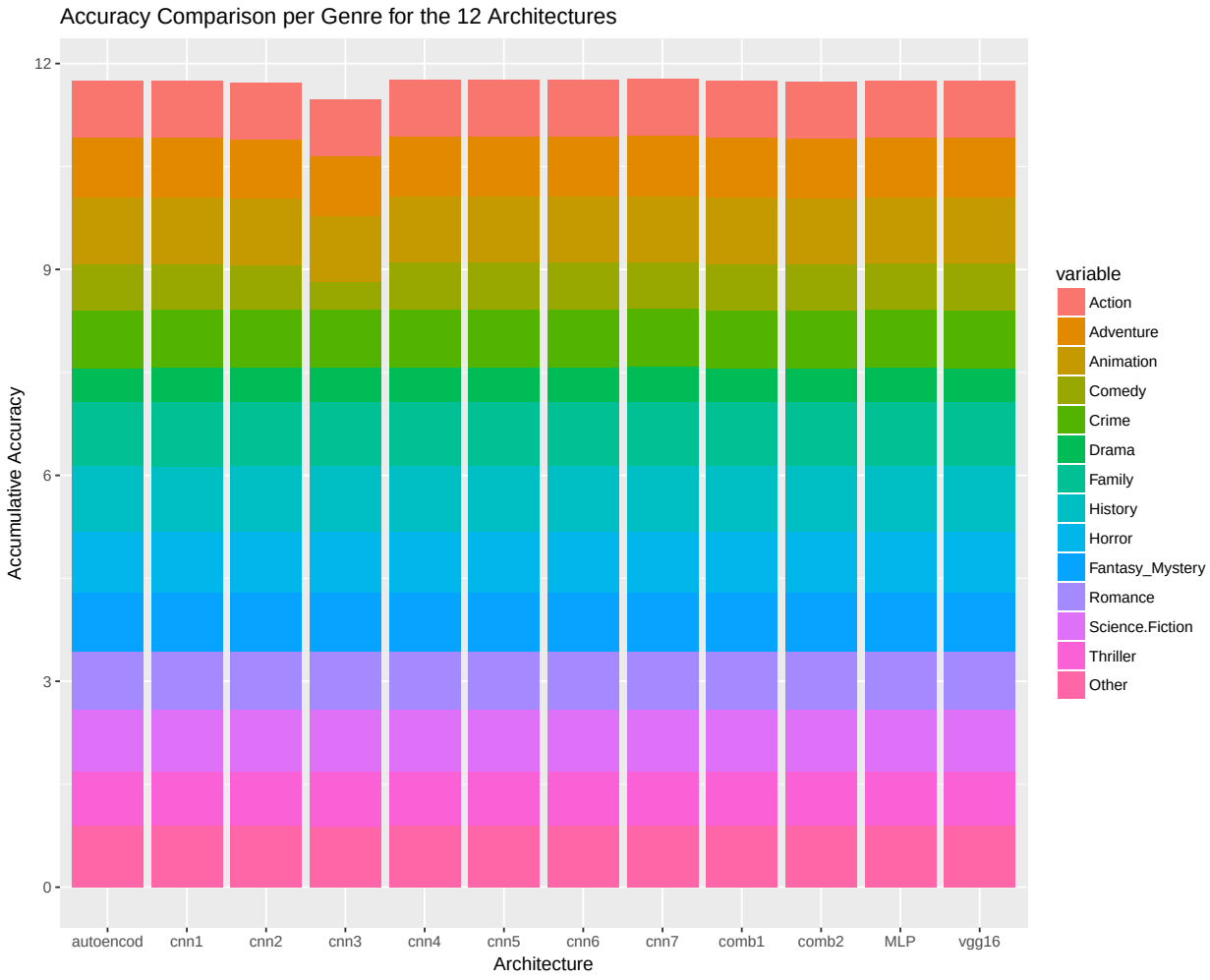
```
}
```

### Execution for the 12 architecture

```
# Perform for the fist file (as a seed)
acc1 = accuacy(categ("1pred.csv"))

# Loop for the other files
for (i in 2:12) {

  # onstruct file name
  fname = paste(toString(i) ,"pred.csv" , sep = "")

  # Calculate the accurracy
  acc = accuacy(categ(fname))

  # Add to the Array
  acc1 = rbind(acc1 , acc)

}
```

### Visualization

```
# Construct Data Frame
Accuracy = data.frame(acc1,  row.names = seq(1,12))

# Assign column names matching genres
colnames(Accuracy) = colnames(Labels)

# Assign Architecture names
Accuracy$Archs = c("cnn1" , "cnn2" , "cnn3" , "cnn4" , "cnn5" , "cnn6" , "cnn7" , "MLP" ,
                   "autoencod", "vgg16" , "comb1" , "comb2")
# Re-arrange for plotting
Acc = melt(Accuracy , id.vars = "Archs")

# Plot
ggplot(Acc , aes(x = Archs , y = value )) +
  geom_bar(aes(fill = variable) , alpha = 1.0 , stat = "identity") +
  xlab("Architecture") +
  ylab("Accumulative Accuracy") +
  ggtitle("Accuracy Comparison per Genre for the 12 Architectures")
```




# Sample heatmap for two architectures

- Heatmap will represent the proximity between the actual class and the predicted class
- All values are scaled between 0 and 1
- A small sample of 100 movie were considered

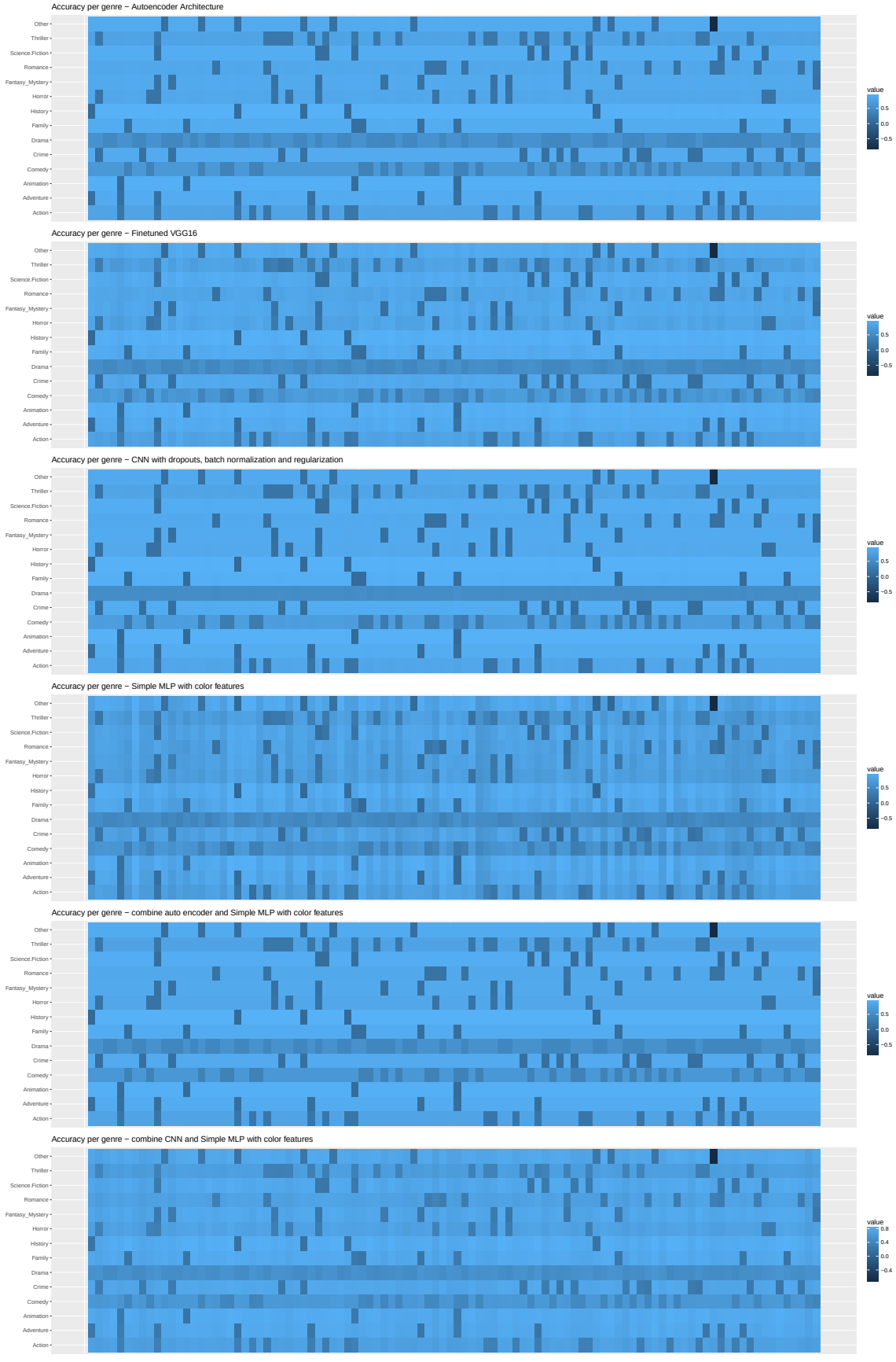
## Open Data File

```
# Auto encoder architecture
meas_enc = subset(read.csv("9_prox.csv") , select = Action:Other)

# Fine-tunned VGG16 architecture
meas_vgg = subset(read.csv("10_prox.csv") , select = Action:Other)

# CNN with Dropout, Batch Normalization and regularization
meas_cnn = subset(read.csv("7_prox.csv") , select = Action:Other)

# Simple MLP for the color features
meas_simple = subset(read.csv("8_prox.csv") , select = Action:Other)
```

```
# Combined autoencoder and simple MLP
meas_comb1 = subset(read.csv("11_prox.csv") , select = Action:Other)

# Combined CNN and simple MLP
meas_comb2 = subset(read.csv("12_prox.csv") , select = Action:Other)

# Add an index
id = seq(1,100)

# Attach the index to the data frames:
# ------------------------------------

meas_enc$id = id

meas_vgg$id = id

meas_cnn$id = id

meas_simple$id = id

meas_comb1$id = id

meas_comb2$id = id
```

```
p1 <- ggplot(melt(meas_enc , id.vars = "id") , aes(x = id , y = variable)) +
         geom_tile(aes(fill = value)) + ylab ("") + xlab("") +
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank()) +
  ggtitle("Accuracy per genre - Autoencoder Architecture")


p2 <- ggplot(melt(meas_vgg , id.vars = "id") , aes(x = id , y = variable)) +
         geom_tile(aes(fill = value)) + ylab ("") + xlab("") +
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank()) +
  ggtitle("Accuracy per genre - Finetuned VGG16")

p3 <- ggplot(melt(meas_cnn , id.vars = "id") , aes(x = id , y = variable)) +
         geom_tile(aes(fill = value)) + ylab ("") + xlab("") +
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank()) +
  ggtitle("Accuracy per genre - CNN with dropouts, batch normalization and regularization")

p4 <- ggplot(melt(meas_simple , id.vars = "id") , aes(x = id , y = variable)) +
         geom_tile(aes(fill = value)) + ylab ("") + xlab("") +
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank()) +
  ggtitle("Accuracy per genre - Simple MLP with color features")
```

```
p5 <- ggplot(melt(meas_comb1 , id.vars = "id") , aes(x = id , y = variable)) +
         geom_tile(aes(fill = value)) + ylab ("") + xlab("") +
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank()) +
  ggtitle("Accuracy per genre - combine auto encoder and Simple MLP with color features")

p6 <- ggplot(melt(meas_comb2 , id.vars = "id") , aes(x = id , y = variable)) +
         geom_tile(aes(fill = value)) + ylab ("") + xlab("") +
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank()) +
  ggtitle("Accuracy per genre - combine CNN and Simple MLP with color features")



multiplot(p1, p2 , p3 , p4 , p5 , p6 , cols = 1)
```


Accuracy per genre − Autoencoder Architecture
Accuracy per genre − Finetuned VGG16
Accuracy per genre − CNN with dropouts, batch normalization and regularization
Accuracy per genre − Simple MLP with color features
Accuracy per genre − combine auto encoder and Simple MLP with color features
Accuracy per genre − combine CNN and Simple MLP with color features
Other
Thriller
Science.Fiction
Romance
Fantasy_Mystery
Horror
History
Family
Drama
Crime
Comedy
Animation
Adventure
Action
value
0.5
0.0
−0.5
0.8
0.4
−0.4

## Plot Average Accurracy accross the 12 architectures

- The accuracy will be evaluated per genre basis

```
# Calculate the average
accavg = colMeans(acc1)

# Construct a data frame
Avg = data.frame(accavg)

# Agg Genres
Avg$genres = colnames(Labels)

# Plotgeneration
ggplot(Avg , aes(genres , accavg)) + geom_bar(fill = "steelblue" , stat = "identity") +
  ylab("Average Accurracy") +
  ggtitle("Average Accuracy per Genre")
```