University of Connecticut

IMAGE PROCESSING: YELP-PHOTO CLASSIFICATION

TEAM MATRIX

Janani Giridhar, Kaushik Vijayakumar, Sriram Rajagopalan, Thejes Venkataraman, Tejas Naren TN, Vineeth Raghav

CONTENTS

EXECUTIVE SUMMARY

Yelp is a multinational company which hosts online reservation services and user reviews about the local businesses and restaurants around the world. It has one of the largest online user community and hosts over 121 million user reviews, with tens of millions of photos uploaded. Yelp extensively deploys analytics for its business solutions and to enhance its user experience, for instance, to recommend most helpful and relevant reviews for its users.

Every restaurant in the Yelp community is assigned a set of predefined tags based on image uploaded by the end user. Our objective was to classify these restaurants with the predefined tags by processing the images without human intervention. This project involves a seven-layered approach. Each layer in the process has several functionalities which would be detailed in the report with suitable illustrations.

The project starts with the feature extraction process for each photo and proceeds to aggregate these features at the business level. These business level features act as inputs to the binary classification models that are used for prediction.

The enlisted are the results achieved in this project

- XGBoost and Random Forest models were selected to yield the desired results
- The Accuracy for each model

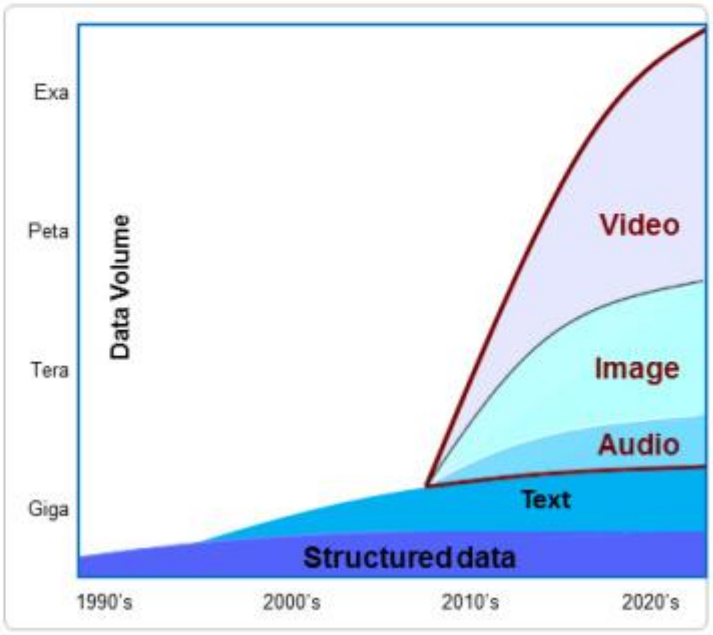
XGBoost	78.76%
Random Forest	78.35%
Averaged Model	78.84%

- Averaged the predictions of XGBoost and Random Forest to take the best of both the models and achieve a marginally better prediction.

This algorithm can be used to process the images uploaded in the Yelp community and automate the process of classifying a restaurant to the predefined tag.

INTRODUCTION

Image Processing is a technical analysis of the various complex aspects and characteristics of an image by deploying algorithms. It can be considered as a type of signal processing where the input is an image and output is the characteristics of the image. With the image and video category of data increasing over the years as shown in the graph below, it becomes extremely important to research on image processing techniques so that we can make work easier in the near future where there will be plenty of digital data to be processed.



There are wide range of applications using digital image processing which can be used across different industries starting from healthcare, Forensic sciences, astronomy, defense and so on. One such application is to classify images of restaurants and food items in the social media pages and websites. Since we are largely dependent on reviews and comments to determine the place for lunch/dinner, the scope for algorithms to automatically classify a restaurant under a certain category increases as well. This classification will quickly provide the customer with an idea about what can be expected from the place.

BUSINESS PROBLEM

Currently, when the Yelp users submit a review, they can upload pictures alongside the reviews. In an era of photo-centric social storytelling, Yelp's users upload an enormous number of photos alongside their reviews. However, when the users upload a review they need to manually select restaurant labels with appropriate categories such as good for lunch/dinner, serves alcohol, outdoor seating available, takes reservations etc. Since selecting the labels is optional, many users prefer not to select the labels which leaves many of the restaurants only partially categorized. The other major problem is that even if the tagging is made mandatory, there is a high chance of the restaurants being incorrectly tagged.

The solution of this project is to build a model that reads a dataset of user submitted photos and automatically tags restaurants with one or more labels relevant to the restaurant or business.

DATA & PACKAGE DESCRIPTION

In the given Yelp dataset, there were 234,846 photos. These photos provide rich local business information across categories. In this dataset, we have been provided with photos that belong to a business and we were asked to predict the business attributes. There are 9 different attributes in this problem:

0: good for lunch

1: good for dinner

2: restaurant takes reservations

3: outdoor seating

4: restaurant is expensive

5: serves alcohol

6: has table service

7: ambience is classy

8: good for kids

The data which has been used for image classification is present in 6 files. The data descriptions are given below

- *photos.tgz* - photos of the training set
- *photo_to_biz_ids.csv* - maps the photo id to business id
- *train.csv* - main training dataset. Includes the business id's, and their corresponding labels.

Data Dictionary

- business: Provides the unique id of the business
- photo_id: Provides the unique id of a photo
- labels: Provides us the unique label id in which the photos have been segregated.

Packages Explanation

Packages are collections of R functions, data, and compiled code in a well-defined format. The directory where packages are stored is called the library. R comes with a standard set of packages.

Apart from the mxnet and imageR package which has been primarily used for advanced Image Processing, there are some other packages as well which have been used during the project. These Packages are explained below.

1. ***stringR*** ¬- This package in R is used for performing common string operations such as Character manipulation, Pattern matching and manipulating whitespace.
2. ***readR-*** This package in R is used for reading rectangular data like 'csv', 'tsv', and 'fwf'. It is used for flexibly parsing many types of data.
3. ***plyr*** – This package in R in used for splitting up a big data structure into homogeneous pieces, applying a function to each piece and combining all the results back together.
4. ***tools*** - This package in R primarily provides us the required set of tools for package development, administration and documentation.
5. ***devtools*** – This package in R is used for installing a package which is not present in Cran library. This was used to install a new package in R.
6. ***imageR*** – This is an advanced image processing package used in R. This package was used for some of the key aspects in the project such as displaying and plotting images, working with pixel neighborhoods, resizing, warping, cropping and denoising.

7. ***mxnet*** - This package is not available in the Cran library and has been downloaded from GitHub repository. This is a R package that brings fast GPU computation and state-of-art deep learning. This package is a neural net which allows us to flexibly configure state-of-art deep learning models backed by the fast CPU and GPU back-end.
8. ***randomForest*** – This package in R implements Breiman's random forest for classification and regression. It can also be used in unsupervised mode for assessing proximities among data points.
9. ***xgboost*** – This package is used for performing gradient boosting. It has both linear model solver and tree learning algorithms. This makes xgboost at least 10 times faster than existing gradient boosting implementations

SOLUTION METHODOLOGY:

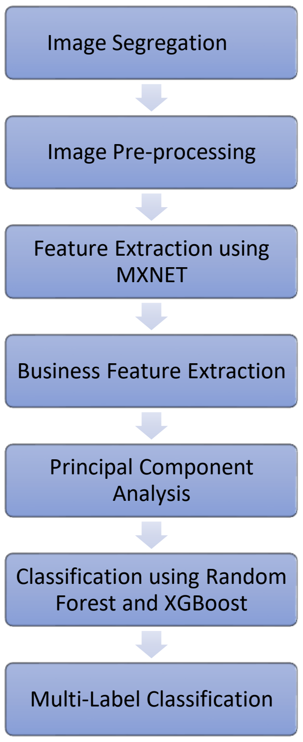


Image Segregation
Image Pre-processing
Feature Extraction using MXNET
Business Feature Extraction
Principal Component Analysis
Classification using Random Forest and XGBoost
Multi-Label Classification

`

The solution segment has been divided into 7 different parts as shown in the chart above with each part representing a significant step in achieving the final result.

Since there were 2000 different restaurants with more than 234,846 images which had to be processed, our initial task was to seggregate these images into respective business category. Each business was seggregated as a folder containing the images and these were processed based on the respective business ids. Secondly, the images had to pre-processed before being subjected to feature extraction. There were various sizes and pixels of images that were present for each business. We pre-processed the images in such a way that all the images were re-sized based on the shorter width and also was trasnformed into a pixel size of 224*224.

In the third phase, we extracted the features from the images using the MXNet package. There were 1000 columns of features extracted for each image corresponding to a business with probablity values ranging between 0 and 1. Each feature was given a particluar weightage depending upon the charateristics of an image at every pixel. The next step was to convert the features of individual images into business level features. There were many images corresponding to each business and all the features extracted from these images must be aggregated into a single feature representing the business. We calculated the mean of the features of all the images in such a way the resultant is a unique feature respresenting a business. The result of the fourth step gives 2000 unique features, each acting as a representative of all images pertaining to a business.

In the fifth phase, we performed Principal Component Analysis on the business features. Since there were 2000 businesses each having 1000 features represented as continuous variables, we performed dimensioanlity reduction using PCA. We were able to retain 91% of the variation using 500 variables which was exactly half the number of original list of variables.

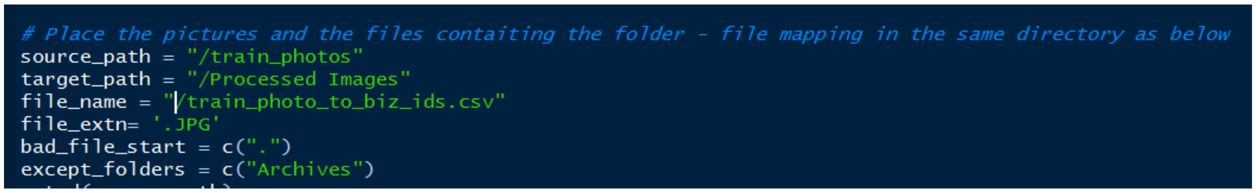
Finally, we built various classification models to classify the business features into various categories. The Random Forest and XGBoost models had the best results when compared to others. Hence we decided to combine the results through a weighted average approach and finally classify the features into multiple categories. The final result had the images of a restaurant processed and based on the results, the restaurant was classified into multiple labels based on the charateristics of images.

The initial challenge was that all the user submitted photos for all the businesses (restaurants) are placed under one directory. The classification of the business and the photo (file name) is provided in a separate csv file. However, while training a model, it is cumbersome and inefficient to read the file every time to classify the business based on the photo.

To overcome this challenge, we segregated the images as the first phase of pre-processing based on the business-to-photo mapping provided in the csv file. As a result, there is a subdirectory created for every business under which all its photos are placed.

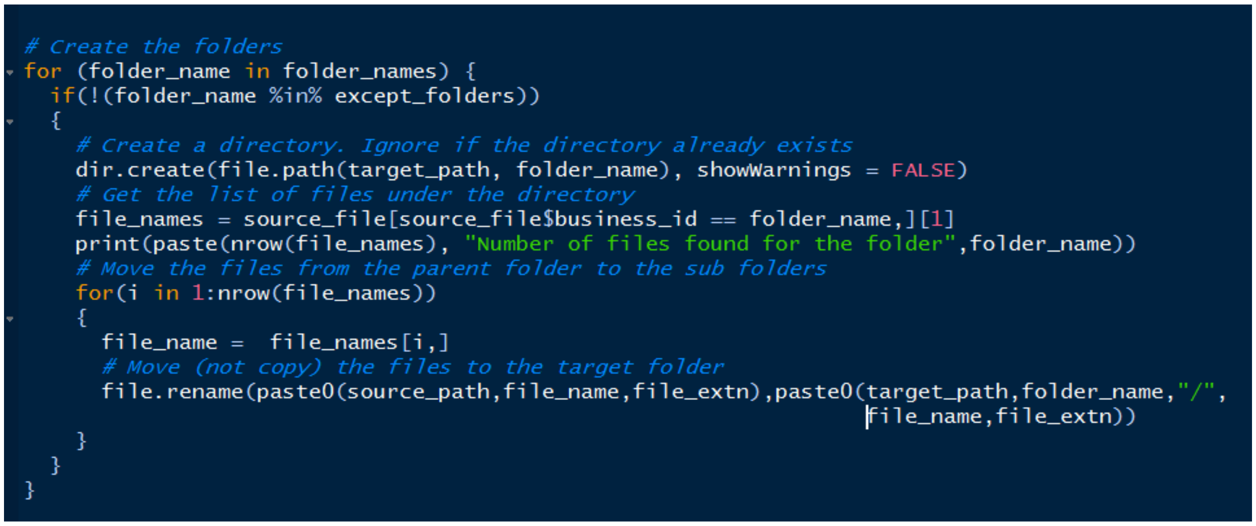
It is also observed that there are several invalid photos in the dataset which begins with "._" which needs to be excluded while building the model. Hence, we have excluded these photos during this segregation process, and these photos can be removed at the end of the step manually from the parent directory.

Step 1: R code snippet to specify the source and target directories.



```r
# Place the pictures and the files contaiting the folder - file mapping in the same directory as below
source_path = "/train_photos"
target_path = "/Processed Images"
file_name = "/train_photo_to_biz_ids.csv"
file_extn= '.JPG'
bad_file_start = c(".")
except_folders = c("Archives")
```

Step 2: R code snippet to segregate the photos based on its business by reading the csv file

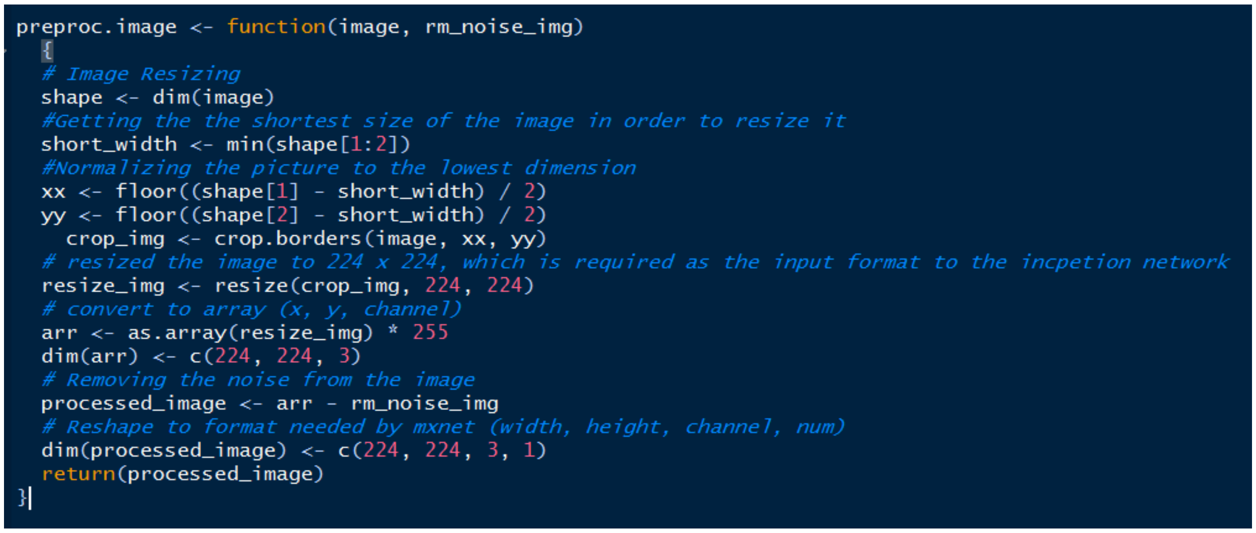


```r
# Create the folders
for (folder_name in folder_names) {
  if(!(folder_name %in% except_folders))
  {
    # Create a directory. Ignore if the directory already exists
    dir.create(file.path(target_path, folder_name), showWarnings = FALSE)
    # Get the list of files under the directory
    file_names = source_file[source_file$business_id == folder_name,][1]
    print(paste(nrow(file_names), "Number of files found for the folder",folder_name))
    # Move the files from the parent folder to the sub folders
    for(i in 1:nrow(file_names))
    {
      file_name =  file_names[i,]
      # Move (not copy) the files to the target folder
      file.rename(paste0(source_path,file_name,file_extn),paste0(target_path,folder_name,"/",
                                                                  file_name,file_extn))
    }
  }
}
```

IMAGE PROCESSING:

Image Pre-Processing:

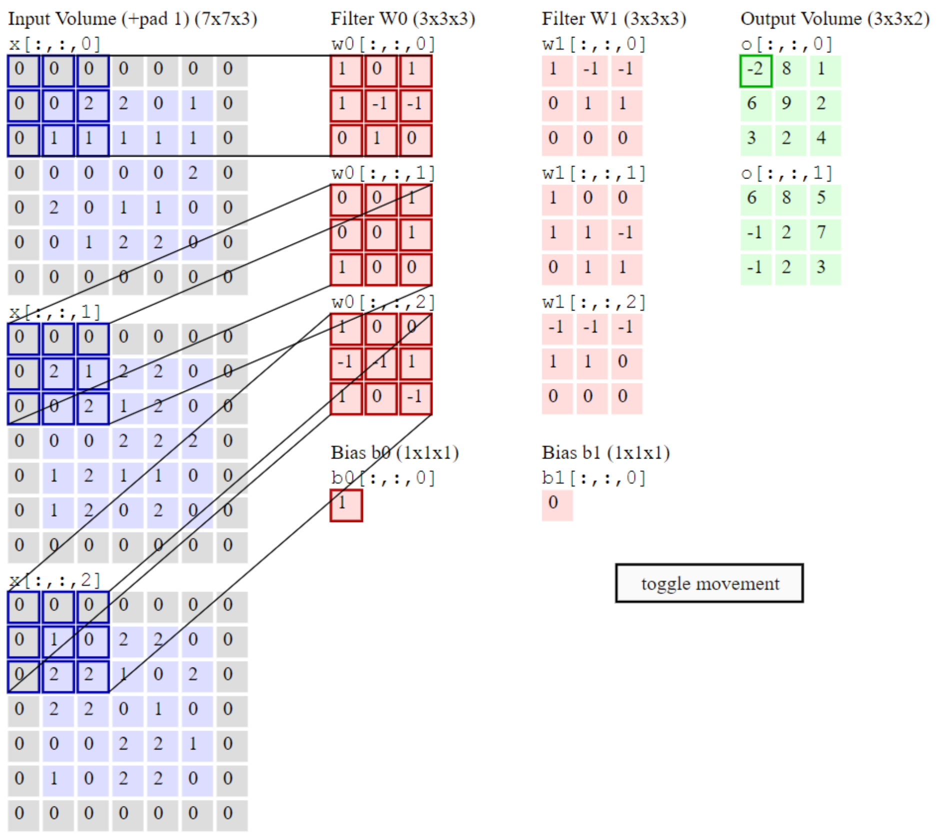
Step 1: Since the images were uploaded by multiple users, the resolution of the pictures was different for every photo. For to model to train well, it is necessary that all the photos have the same configuration and yet preserves the features, to extract the useful features. So, after analysis we picked the ideal configuration of the images as 224 x 224 which is recommended for the library mxnet.



```
preproc.image <- function(image, rm_noise_img)
  {
  # Image Resizing
  shape <- dim(image)
  #Getting the the shortest size of the image in order to resize it
  short_width <- min(shape[1:2])
  #Normalizing the picture to the lowest dimension
  xx <- floor((shape[1] - short_width) / 2)
  yy <- floor((shape[2] - short_width) / 2)
    crop_img <- crop.borders(image, xx, yy)
  # resized the image to 224 x 224, which is required as the input format to the incpetion network
  resize_img <- resize(crop_img, 224, 224)
  # convert to array (x, y, channel)
  arr <- as.array(resize_img) * 255
  dim(arr) <- c(224, 224, 3)
  # Removing the noise from the image
  processed_image <- arr - rm_noise_img
  # Reshape to format needed by mxnet (width, height, channel, num)
  dim(processed_image) <- c(224, 224, 3, 1)
  return(processed_image)
}
```

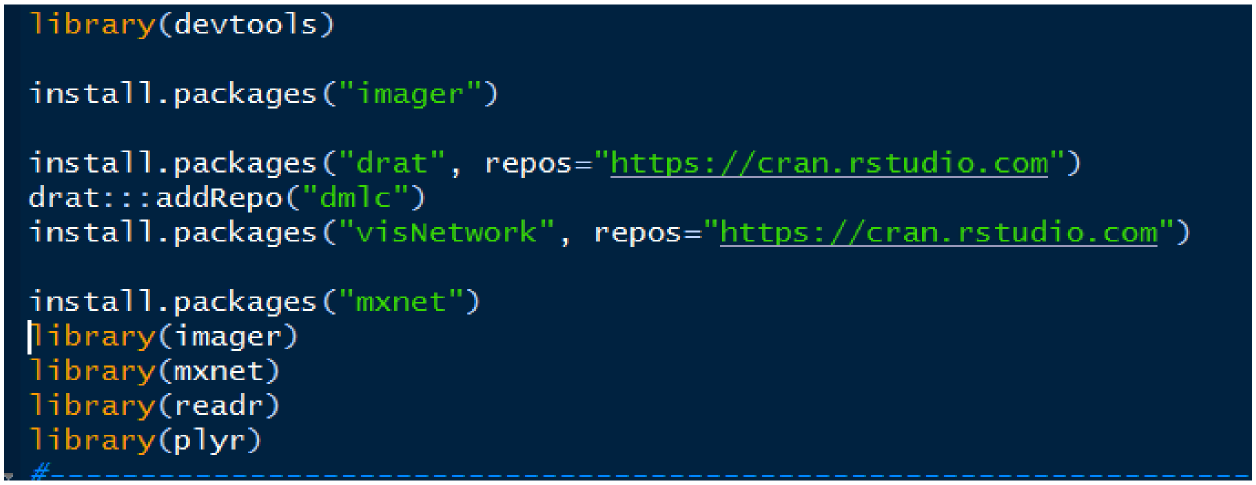
Image Processing with MXnet:

The first step in the Image processing is Feature extraction. Feature extraction is a kind of dimensionality reduction process which converts the higher dimensional data like images into lower dimensional space. It converts the images into meaningful vectors that preserve the important information pertaining to the images. We have made use of the Inception BN neural network to extract the features. This model has been pre trained on the ImageNet database which has more than 14 Million Photos. The Mxnet package has all the required functions that are required to handle the Inception BN model.The images are converted into a pixel matrix, which is the input to the neural network. The dimension reduction process is shown in the image below.



Feature Extraction Process:

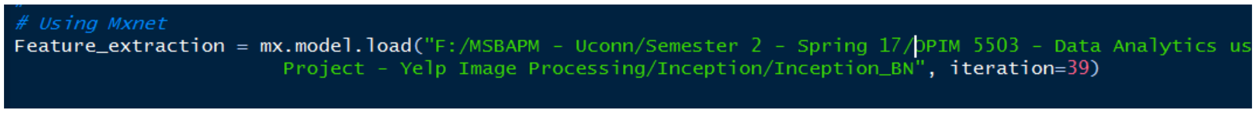
Step 1: To Install and load the respective packages used to extract features



```
library(devtools)

install.packages("imager")

install.packages("drat", repos="https://cran.rstudio.com")
drat:::addRepo("dmlc")
install.packages("visNetwork", repos="https://cran.rstudio.com")

install.packages("mxnet")
library(imager)
library(mxnet)
library(readr)
library(plyr)
#-----------------------------------------------------------------
```

Step 2: Using the INCEPTION_BN neural network

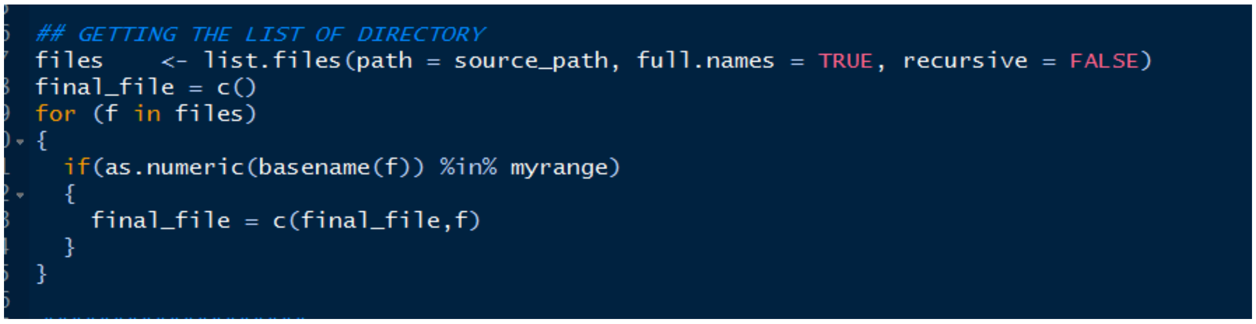
Since the *mxnet* package was downloaded separately from Github, we had the variable named *model* to hold the *Inception_BN* neural network which can be later called to extract feature from the images



```
# Using Mxnet
Feature_extraction = mx.model.load("F:/MSBAPM - Uconn/Semester 2 - Spring 17/OPIM 5503 - Data Analytics us
                        Project - Yelp Image Processing/Inception/Inception_BN", iteration=39)
```

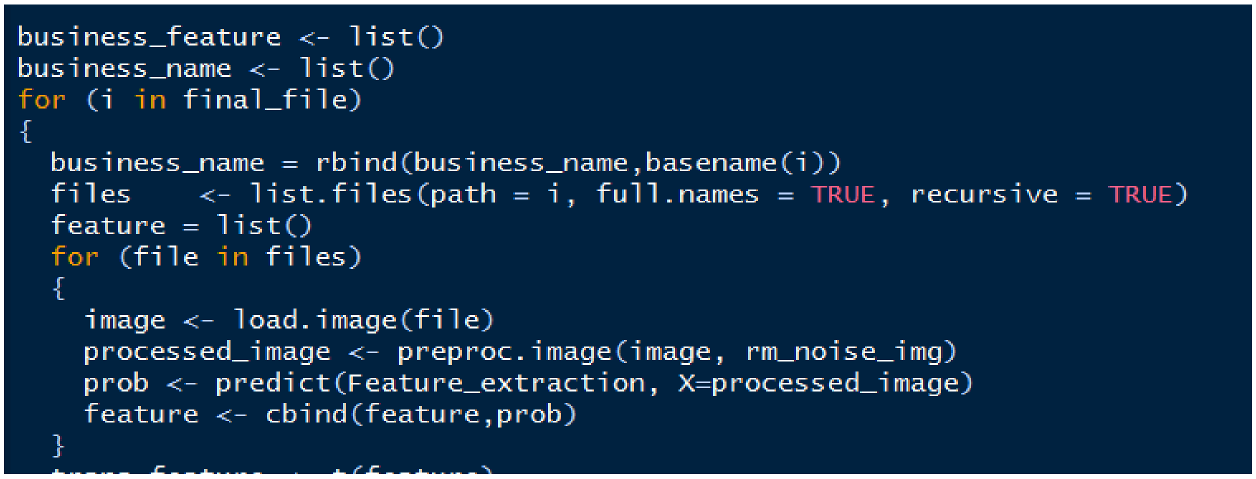
Step 3: Getting the list of directory

Since each folder represents a business, it is necessary to list the directory and process one business at a time. We used the following code to do the same



```
## GETTING THE LIST OF DIRECTORY
files    <- list.files(path = source_path, full.names = TRUE, recursive = FALSE)
final_file = c()
for (f in files)
{
  if(as.numeric(basename(f)) %in% myrange)
  {
    final_file = c(final_file,f)
  }
}
```

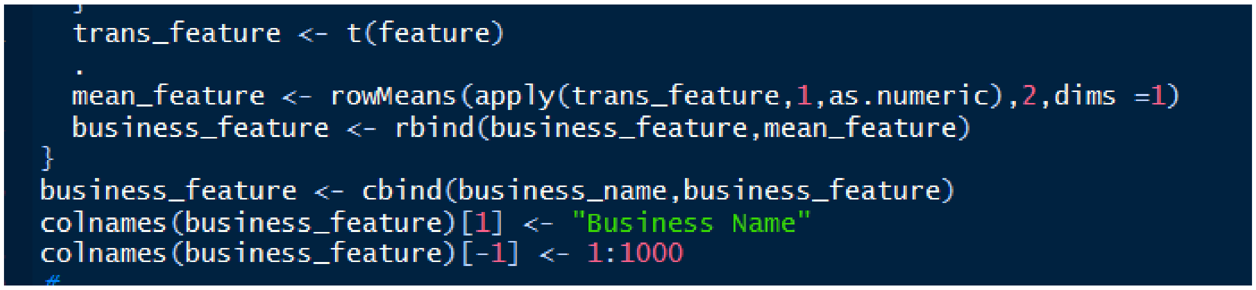
Step 4: Finally, we extracted the features from each image by using the preproc.image function and the INCEPTION_BN neural network stored in the variable feature_extraction. The code returns 1000 features for each photo that has been given as the input.



```
business_feature <- list()
business_name <- list()
for (i in final_file)
{
  business_name = rbind(business_name,basename(i))
  files    <- list.files(path = i, full.names = TRUE, recursive = TRUE)
  feature = list()
  for (file in files)
  {
    image <- load.image(file)
    processed_image <- preproc.image(image, rm_noise_img)
    prob <- predict(Feature_extraction, X=processed_image)
    feature <- cbind(feature,prob)
  }
```

Business Feature Aggregation

The number of photos in each business differs from business to business. The features extracted in the previous section, is for individual photos. Since the objective of the project is to classify the business on the whole, the features need to be aggregated at the business level. The features of all the photos associated with a business is averaged to get the average feature for that business. It is assumed that these features explain the unique characteristics of the business and can be used to predict the labels.

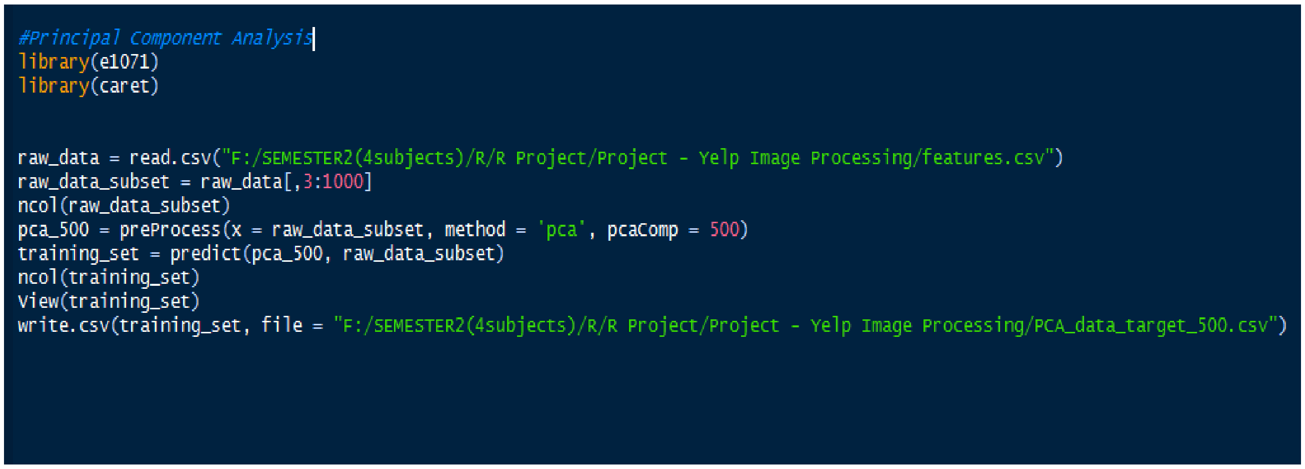


```
    trans_feature <- t(feature)
    .
    mean_feature <- rowMeans(apply(trans_feature,1,as.numeric),2,dims =1)
    business_feature <- rbind(business_feature,mean_feature)
  }
  business_feature <- cbind(business_name,business_feature)
  colnames(business_feature)[1] <- "Business Name"
  colnames(business_feature)[-1] <- 1:1000
```

Principle Component Analysis

We used principal components analysis to reduce the number of variables and avoid multicollinearity. We have performed PCA on the business features. We have 2000 businesses with 1000 features each. By performing PCA we could retain 91% of the variation using 500 variables which is exactly half the number of original list of variables.

R Code Snippet for Principal Component analysis:



```
#Principal Component Analysis
library(e1071)
library(caret)


raw_data = read.csv("F:/SEMESTER2(4subjects)/R/R Project/Project - Yelp Image Processing/features.csv")
raw_data_subset = raw_data[,3:1000]
ncol(raw_data_subset)
pca_500 = preProcess(x = raw_data_subset, method = 'pca', pcaComp = 500)
training_set = predict(pca_500, raw_data_subset)
ncol(training_set)
view(training_set)
write.csv(training_set, file = "F:/SEMESTER2(4subjects)/R/R Project/Project - Yelp Image Processing/PCA_data_target_500.csv")
```

CLASSIFICATION ALGORITHM

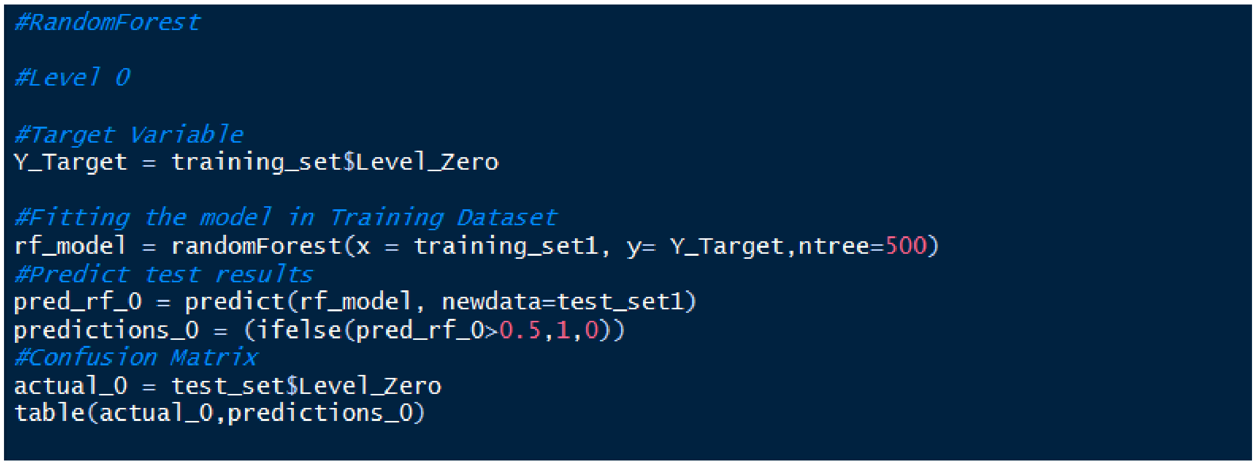
Once the variable reduction is completed using Principle Component Analysis (PCA) we have 500 independent variables and 9 binary classifier labels (one for each attribute).

Our approach was to train the data with different classification model approached using 500 independent variables to predict binary classified labels individually. Once the best model for each label is identified, the model is run with the test data to predict the classification of each label individually (as trained) and then the predictive binary response variables (labels) are cumulated together as 9 variables.

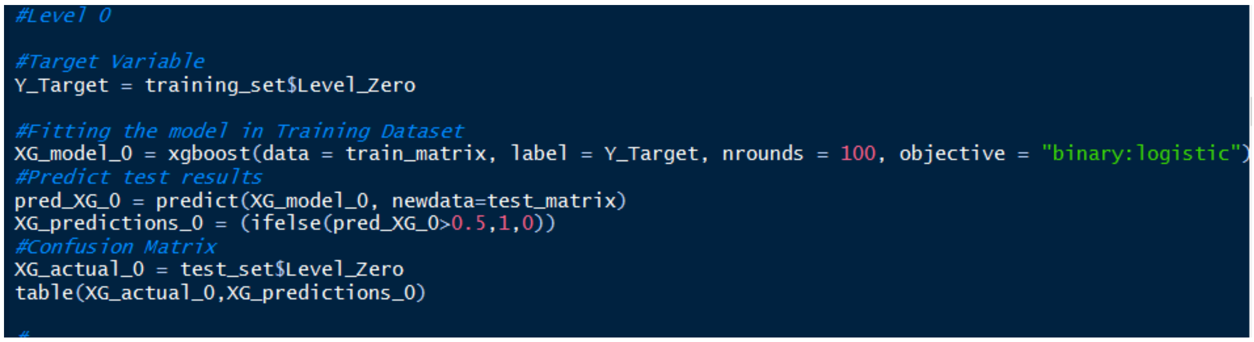
Below are the classification models used:

- SVM
- Logistic regression
- Random Forest
- XGBoost

However, on running all the classification model approaches, the Random Forest and XGBoost models yielded best results out of all our classifications models. So, we aggregated the binary prediction probabilities of Random Forest and XGBoost to arrive at the best final binary predictions. The following code were executed for Random forest and XGBoost models.



```
#RandomForest

#Level 0

#Target Variable
Y_Target = training_set$Level_Zero

#Fitting the model in Training Dataset
rf_model = randomForest(x = training_set1, y= Y_Target,ntree=500)
#Predict test results
pred_rf_0 = predict(rf_model, newdata=test_set1)
predictions_0 = (ifelse(pred_rf_0>0.5,1,0))
#Confusion Matrix
actual_0 = test_set$Level_Zero
table(actual_0,predictions_0)
```



```
#Level 0

#Target Variable
Y_Target = training_set$Level_Zero

#Fitting the model in Training Dataset
XG_model_0 = xgboost(data = train_matrix, label = Y_Target, nrounds = 100, objective = "binary:logistic")
#Predict test results
pred_XG_0 = predict(XG_model_0, newdata=test_matrix)
XG_predictions_0 = (ifelse(pred_XG_0>0.5,1,0))
#Confusion Matrix
XG_actual_0 = test_set$Level_Zero
table(XG_actual_0,XG_predictions_0)
```

Below is the prediction accuracy for each label:

labels	0-0	0-1	1-0	1-1	Accuracy
label 0	305	27	100	68	75%
label 1	213	49	62	176	78%
label 2	201	50	38	211	82%
label 3	147	92	98	163	62%
label 4	348	16	76	60	82%
label 5	132	63	34	271	81%
label 6	107	65	26	302	82%
label 7	340	29	57	74	83%
label 8	122	60	32	286	82%

RANDOM FOREST

Labels	0--0	0--1	1--0	1--1	Accuracy
label 0	293	39	75	93	77%
label 1	216	46	57	181	79%
label 2	202	49	39	210	82%
label 3	149	90	103	158	61%
label 4	337	27	62	74	82%
label 5	134	61	38	267	80%
label 6	112	60	31	297	82%
label 7	328	41	46	85	83%
label 8	132	50	42	276	82%

XG BOOST

Comparing the Random Forest and XGBoost models, both the models had the same accuracy with small deviations in their predictions. Hence their prediction probabilities where averaged to get a new averaged model. The averaged model seems to take the best of both the models and gives a marginally better prediction.

Labels	**0--0**	**0--1**	**1--0**	**1--1**	**Accuracy**
label 0	294	38	76	92	77%
label 1	217	45	57	181	80%
label 2	199	52	36	213	82%
label 3	151	88	103	158	62%
label 4	337	27	64	72	82%
label 5	134	61	38	267	80%
label 6	110	62	28	300	82%
label 7	331	38	49	82	83%
label 8	132	50	40	278	82%

AVERAGED MODEL

CONCLUSION

The proposed solution consisted of seven phases starting from Image segregation and preprocessing to classification. After the initial preprocessing of images, we extracted the features using Mxnet package and aggregated these features with respect to each business. These features represented the unique characteristics of the images corresponding to a specific restaurant. The dimensions were further reduced by using Principle Component Analysis and the resultant was used as input to build classification models. The final accuracy of 78.84% was achieved by using an average of the results from Random Forest and XGBoost classification models.

This machine learning algorithm can be incorporated by Yelp to automatically classify the restaurants into different categories.

REFERENCES

1. Introduction to Image Processing by **Frery**, Alejandro C., **Perciano**, Talita http://www.springer.com/us/book/9781447149491
2. https://www.r-bloggers.com/image-recognition-in-r-using-convolutional-neural-networks-with-the-mxnet-package/ - Using Image processing in R.
3. https://cran.r-project.org/web/packages/magick/vignettes/intro.html - Advanced Image Processing in R using the magick package.
4. https://dahtah.github.io/imager/imager.html - Extracting business features from images.
5. http://mxnet.io/tutorials/r/classifyRealImageWithPretrainedModel.html - Image Processing Techniques
6. http://cs231n.github.io/convolutional-networks/ - Feature Extraction Process
7. https://dh101.ch/2014/10/21/importance-of-image-data-and-image-processing-techniques-for-digital-humanities/ - Importance of Image processing