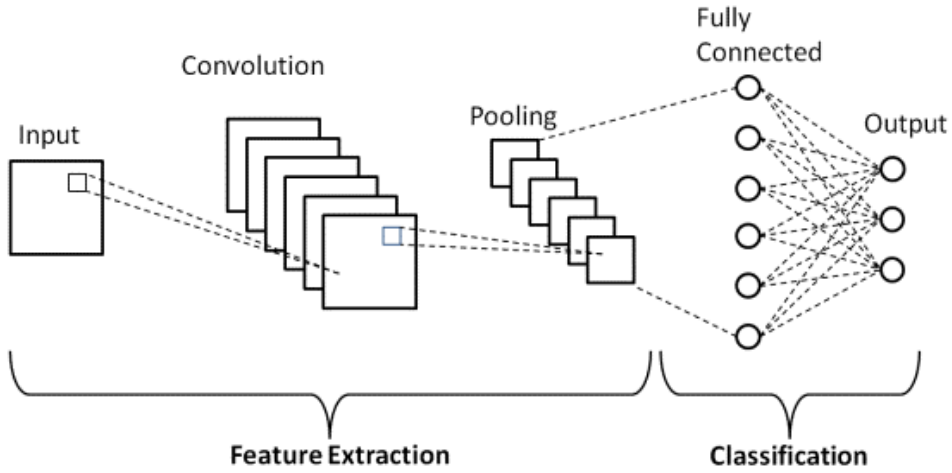


Restaurant Rating Prediction Using Convolutional Neural Networks

Group 7 - Varun Kausika, Tanushree Balaji, Akhila Guttikonda, Shubadha Kapre, Anudeep Kumar Akkana

Abstract

The problem of data scarcity is very important since data is at the core of any AI project. The size of a dataset is often responsible for poor performances in ML projects. For a small business, this could be especially meaningful since the first few projects define whether a startup can break into an industry or not. We tackle this problem for the specific case of restaurants. When prospective restaurant owners open up their business, they often want to get a sense of how well their business might perform/how customer sentiment might play out even before officially opening to the public.

CNNs have historically been used to gain insights from image data in a relatively compact way compared to other dense neural network structures. In this project, we attempt to predict the star rating of a new restaurant by combining images of various aspects (food, drink, interior, exterior) of the business into a compound score.

Introduction

Have you ever struggled with choosing one picture out of the 100 pictures you clicked to post on social media? All the time? Yes, we feel you!
Now think how difficult it is for restaurant owners to select the right images to show on their websites or Google to attract the most number of customers and get high user ratings.
We, graduate students at the University of Texas at Austin studying Business Analytics have set out to address this very problem using Convolutional Neural Networks and help restaurant owners select the best pictures to show on restaurant recommendation websites that would get them high user ratings. Our primary objective is to predict the star rating of restaurants based on

images. This is especially applicable for new business owners to predict if their restaurant will be able to catch up with the existing businesses in the market. Business owners can also use this model to select the best pictures to be put up on restaurant aggregator websites like Zomato.

Established restaurants and seasoned businesses have abundant historical data like reviews and ratings to understand which pictures would attract the most number of customers. However, it is not the same case for budding restaurants because they don't have enough historical data. The model we are building here is specifically for new restaurant businesses looking to get listed on food delivery & restaurant recommendation platforms who seek to understand what kind of thumbnails the audience gravitates towards. Our model will predict the user rating by taking the restaurant images posted by the customers of various restaurants as input. Since this is an image classification problem, we are using Convolutional Neural Networks. CNN is a Deep Learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and thus classify images based on the differences in features.

Follow along this blog to know the approach we took to tackle the problem, the platform we used, data processing, the models tried out, what our results were, and some of the challenges we faced along the way.
Happy Reading!

Data Collection

Our dataset is from Yelp, a crowd sourced website for business reviews. The size of the dataset is approximately 14 GB containing the reviews of over 150K restaurants spread over 11 cities across the US. The dataset has about 200K restaurant images including images of food, drinks and ambience and over 6M user ratings.

You can find the dataset here.

Exploratory Data Analysis and Data Pre-Processing

Before building the model on this dataset, we did some exploratory data analysis to understand the patterns in the data and remove anomalies if any.
The dataset mainly has two dataframes, one for the restaurant images and the other for user ratings and textual reviews.
The **Photos** dataframe has columns like photo_id, business_id, caption and label. The **Reviews and Ratings** dataframe has a total of 14 columns but the columns relevant to our problem are : business_id and star_rating (ranging from 1 to 5).
The business_id column present in both the dataframes helps us join the two dataframes and retrieve images and star ratings corresponding to each business.

Photos

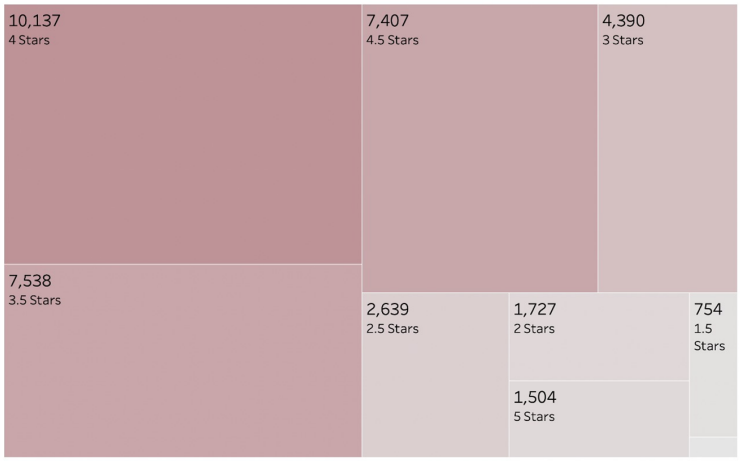
photo_id	200103
business_id	36680
caption	76404
label	5
dtype: int64	

Reviews

business_id	148447
name	114117
address	122843
city	1416
state	27
postal_code	3361
latitude	135561
longitude	131816
stars	9
review_count	1158
is_open	2
attributes	87661
categories	83160
hours	49822
dtype: int64	

We checked for missing values in both the data frames and removed the few missing values we found in the Photos dataframe. There were no significant missing values in the Ratings and Reviews dataframe.

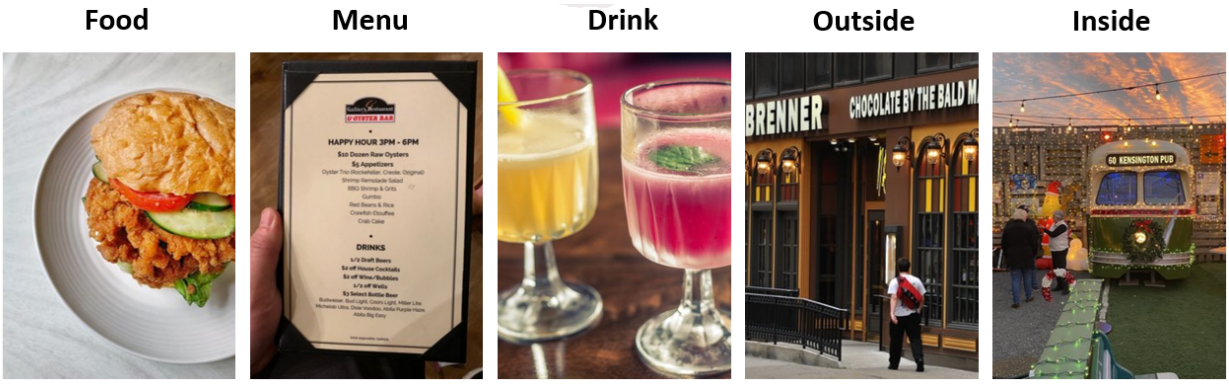
Exploratory Data Analysis revealed the data included a mix of cafes, ethnic restaurants, dessert bars, groceries and breweries. About 85% of the restaurants in the dataset had a star rating of more than 3 with ~70% of them within the star rating range of 3.5 to 4.5.



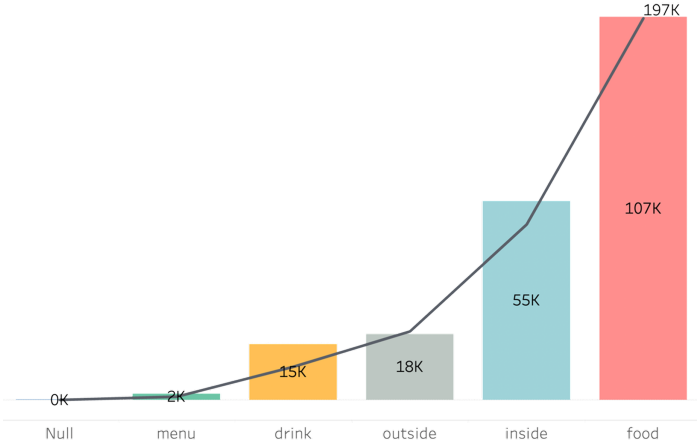
Stars	
1	0.3%
1.5	2.1%
2	4.8%
2.5	7.3%
3	12.1%
3.5	20.8%
4	28.0%
4.5	20.5%
5	4.1%

We filtered and removed all the restaurants that did not have even a single image and we are left with 36k restaurants after filtering. We then checked to see the type of images that were present for these 36K restaurants and found that the images belonged to the following five categories:

- Food
- Drink
- Ambience - Exterior
- Ambience - Interior
- Menu Images



The following graph shows the spread of the images among the five categories listed above.



We can see that the number of images for the 'Menu' category is significantly lower than the other four categories. We can argue that compared to other categories, images related to the menu would have less impact on the user's rating and also because they are text intensive, they do not fit into our current purpose of analysis. So, we decided to eliminate menu images and only include the other four categories.

After removing the Menu related images, we had a total of 195k images for 36K restaurants.

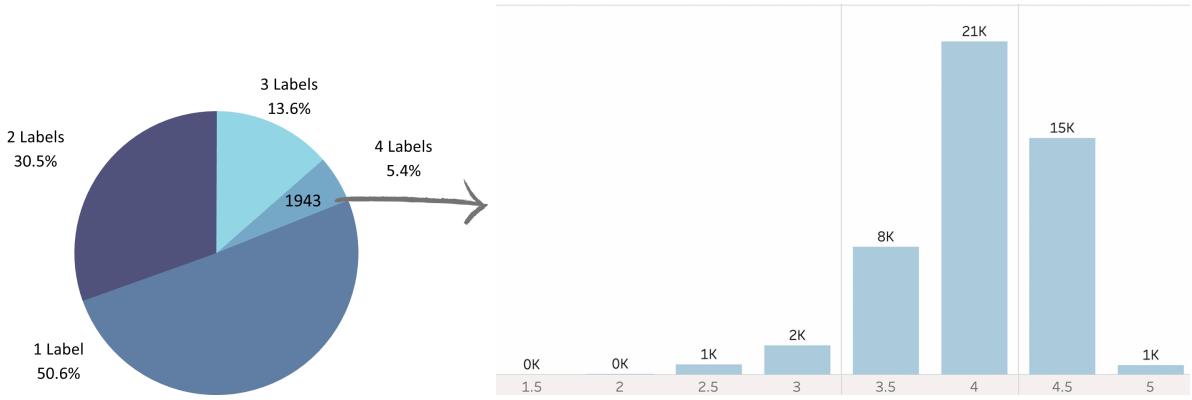
Approach

Since we have 4 categories of images for each restaurant, we used the collage based CNN classification approach. In this approach, a collage of images is given as the input to the model. To build these collages we need restaurants that have at least one image of each category.

So, we filtered for restaurants that have at least one image of food, drink, exterior and interior. We got about 2K restaurants that have all four images.

The total number of images across all these restaurants was found to be around 50K.

The following graph depicts the spread of these 50K images across different star rating categories.



As we can see, the classes are highly skewed. We have about 30K images for restaurants that have a star rating of either 3 or 4, 16K images for restaurants that have a star rating of more than 4 and only 3K images for restaurants with star rating of less than 3.

To choose the images to create collages, we first limited our samples to 4 collages per business. We then created collages for each of these 2K restaurants by randomly combining individual images from the four categories. The average star score (the target variable in our analysis) was calculated to be the average star rating of all 4 images.

For example, say a restaurant X has 3 images of food, 3 images of drinks, 2 images each for exterior and interior . The collage for this restaurant would be created by randomly combining any one of the 3 food images, any one of the 3 drink images and so on. Each combination of these collages would be unique.

Once the collages are created, we randomly select four collages for each restaurant. This gives us a total of 7.5K collages across all restaurants (4 collages for each of the 1.9K restaurants). This means our final dataset has ~7.5K images.

Building the Model

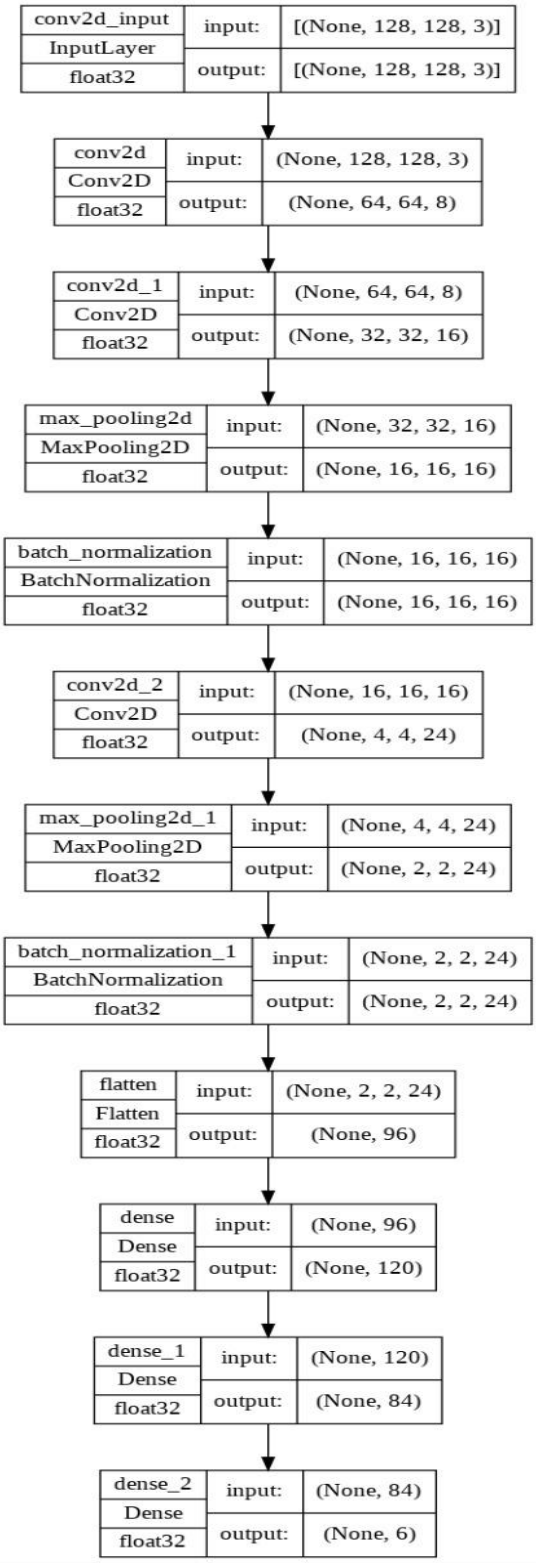
Convolutional Neural Networks

Our classifier employs a Convolutional Neural Network (CNN) which is a Deep Learning algorithm that takes in an input image, assigns importance to various aspects/objects in the image and is able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms.It is also computationally efficient. It uses special convolution and pooling operations and performs parameter sharing

Architecture

A CNN consists of multiple layers of convolutional kernels intertwined with pooling and normalization layers,which combine values and normalize them respectively. Its final step uses a fully connected multi-layer perceptron to give us the actual predicted classes for each input image.

CNN Basic Architecture



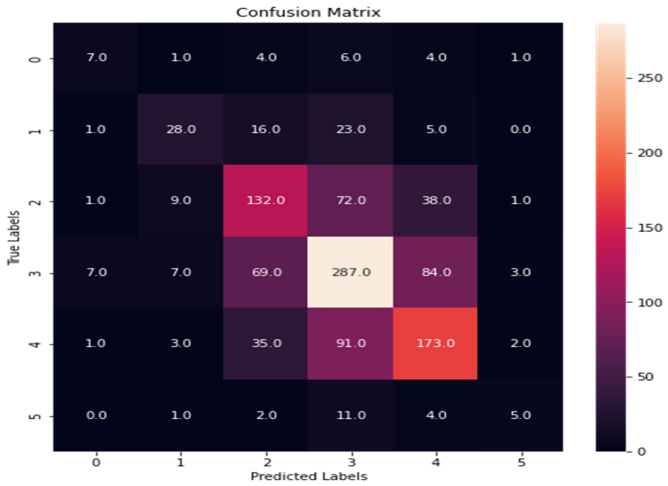
The input to the first layer is of fixed size 128 x 128 RGB image. It is convolved with 8 filters of size 2*2*3 and a stride of 2 resulting in dimension of 64*64*8. The second layer also involves in a convolution operation with 16 filters of size 2*2*8 and a stride of 2 resulting in a 32*32*16. The third layer is a pooling operation with filter size 2*2 and a stride of 2. Hence resulting image dimension will be 16*16*16.

The fourth layer also involves in a convolution operation with 24 filters of size 2*2*16 followed by a fifth pooling layer with similar filter size of 2*2 and a stride of 2. Thus,the resulting image dimension will be reduced to 2*2*24. Now we flatten the output of the convolutional layers to create a single long feature vector of size 96*1 and that is connected to 2 dense hidden layers, of sizes 120*1 and 84*1 respectively. Finally, the softmax activation function in the output layer of size 6*1. Intermediate batch normalizations and max poolings are also used appropriately as shown above.

Once the image dimension is reduced,Three Fully-Connected (FC) layers follow a stack of convolutional layers. All the layers except for the output follow the ReLu activation function.

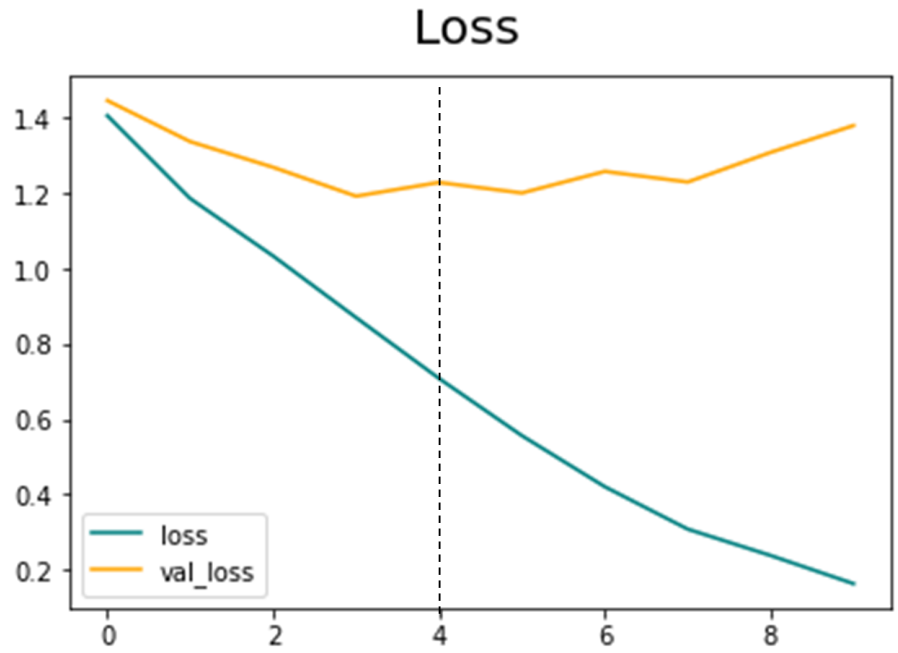
Learnings and Results

We were able to achieve a train accuracy of 67%and validation & test accuracies of over 52%, outperforming the baseline of 40% after 3 epochs of training. With this decent accuracy obtained, new business owners can feed in images of their restaurants for them to estimate and what their potential rating could be. Due to our imbalance in class sizes, the metric of choice to evaluate our model was the F1-score. By the end of this project we have learnt that text data analysis gives you an overview of user sentiment which is a stated preference while pictorial data analysis which is independent of end user is a revealed preference. By using pictorial data we will be waiving away with the process of collecting large amounts of textual data to perform sentiment analysis in the future.



	precision	recall	f1-score	support
0	0.41	0.30	0.35	23
1	0.57	0.38	0.46	73
2	0.51	0.52	0.52	253
3	0.59	0.63	0.61	457
4	0.56	0.57	0.56	305
5	0.42	0.22	0.29	23
accuracy			0.56	1134
macro avg	0.51	0.44	0.46	1134
weighted avg	0.55	0.56	0.55	1134

Finally, we would like to address overfitting vs underfitting. We experimented with various different architectures like VGG like and AlexNet like structures and found our final model had a reasonable number of parameters to train (~29000). This resulted in a significant, but not very steep increase in training performance and generalized decently to our test set. The optimal number of epochs to train was found to be 4. From the graph below, we can clearly see a tendency towards higher variance as we increase the number of epochs.



The future scope of this project could be to include ‘menu’ labeled pictures to understand how the users are responding to the restaurants that post menu pictures and also use these menu

images to predict the cuisine of the restaurant. We can also tailor our model to account for the location of the restaurant like ‘riverfront’, ‘lake view’, ‘valley view’ etc.

Code references:

Most EDA was done using Tableau Desktop

CNN code reference:
https://pub.towardsai.net/multiclass-image-classification-hands-on-with-keras-and-tensoflow-e1cf434f3467
