Is This Restaurant Good for Kids ?

Christina Bogdan
Center for Data Science, NYU
New York City
ceb545@nyu.edu

Vincent Chabot
Center for Data Science, NYU
New York City
vec241@nyu.edu

Urjit Patel
Center for Data Science, NYU
New York City
up276@nyu.edu

I. Introduction

Our goal throughout this project was to predict high level attributes related to venues based on pictures uploaded by customers on the Yelp platform. Typically, we were interested in predicting whether a restaurant would be good for dinner, good for groups and/or good for kids. Each business could potentially have several true labels. There were mainly two points of interest we had to deal with to solve this question :

1) Inferring very general attributes from images/caption analysis (rather than the most usual case of simply classifying the images based on the objects represented)
2) The labels to predict were only business-level labels. However, each image might represent a different business-level level (for example, a picture of a cocktal might indicate that a place was good for groups, but an image of a meal might indicate that that business was also 'Good for Dinner'. This lack of image-level labels made our task semi-supervised.

II. Business Understanding

COULD you tell us, by looking at this picture, if this restaurant looks rather good for kids? Would you rather recommend it as takeaway or for dinner? This is typically the kind of high level, context information customers would like to know before choosing the next restaurant to go to. The kind of questions you could ask a friend that you know already went to this place for example. But what if your favorite recommendation platform / research engine / social network could answer all your questions? Of course, it is not easy to find data that carry such high level information. But both the large amount of pictures uploaded by customers and possibilities offered by machine learning and deep learning techniques makes it interesting to try to answer such questions.

III. Data Understanding

Our initial data consisted of two JSON files and a dataset of images. We highlight information about the datasets below:

Yelp business file

- JSON file of 77445 rows corresponding to 77445 businesses
- Contains fifteen columns. Full details can be found at the bottom of **this** page under the *business* section. We used three columns in our project:
  - *business id*: Unique identifier for each business
  - *categories*: Describes the type of business. We did not directly use this field as a feature, but used it to filter down our data.
  - *attributes*: Our label. The attributes field contains many general subcategories indicating whether the business accepts credit cards, the ambiance of the business, whether it is good for breakfast/lunch/dinner, etc. From these, we wanted to predict all attributes indicating what the business was "Good for". There were five binary attributes relating this - **Good for Breakfast, Good for Lunch, Good for Dinner, Good for Kids, Good for Groups**, and **Good for Dancing**. Originally, we attempted to predict all five of these attributes. Upon seeing the skew of the data, however, we decided to only predict three: **Good for Dinner, Good for Groups, Good for Kids**

Yelp photo file

- JSON file of 200143 rows and 4 columns
- Maps business IDs to photo IDs
- Contains two photo-level features:
  - *label*: Either food, drink, menu, outside, inside. Relates to the content of the image.
  - *caption*: The caption, if any, attached to the image. We used this as a feature in one of our classifiers.

Image dataset

- 200,003 images in JPG format

A. Data Cleaning & Engineering

We parsed our two JSON files in Python using Pandas. This caused some issues for us - when parsing the business JSON file, the *attributes* field was represented by a sub-dictionary in our original dataframe:

	attributes	business_id	categories	city
0	{u'Take-out': True, u'Drive-Thru': False, u'Ou...	5UmKMjUEUNdYWqANhGckJw	[Fast Food, Restaurants]	Dravosburg
1	{u'Happy Hour': True, u'Accepts Credit Cards':...	UsFtqoBl7naz8AVUBZMjQQ	[Nightlife]	Dravosburg

To get our data into the right format, we needed to discard all of the unneeded attributes and create columns out of the ones that were relevant to us:

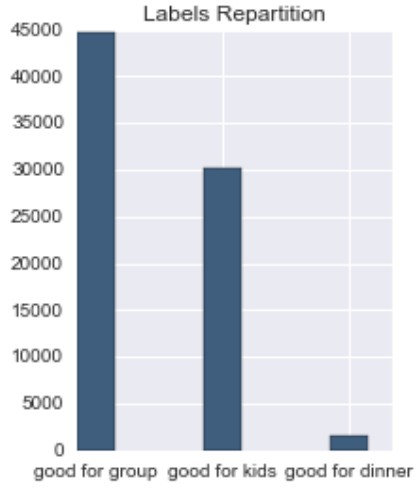
	business_id	Good_for_kids	Good_for_groups	dinner
100	D8y00MIChgtJMhxP4zMJDw	1	1	0
101	0r61RRNbGqCFVqNV_vp9bw	0	1	0
102	wddLqUkQAbslEaoRwD4Mgg	1	1	1
103	JUiR59o5IJ1LaeR3NtiOlw	1	1	0
104	PZjmI2VgR0aEWBx3C7d8Jg	1	1	0

Once this was done, we joined our parsed business dataset to the auxiliary photos dataset, to associate each set of labels with a photo. After this was done, we filtered our photos to only include images that came from businesses with the category **Restaurants, Food, Bars, Nightlife, Fast Food**, and **Coffee & Tea**. Our reasoning was that the labels that we were trying to predict were only relevant to businesses within these categories.

Originally, our plan was to predict all five 'Good For' fields. However, most of our data fell into the three attributes 'Good for Groups', 'Good for Kids', and 'Good for Dinner'.

	photo_id	business_id	caption	label
0	---L4adlY4jfEwo3cnWPcw	WltRLuH8Np_btPnoGaF5XA	NaN	none
1	--00Uxeaz2kzZm3aAuJOVw	1-One6Szx5eQQKfZeVKJWw	Traditional cribs, convertible cribs, twin or ...	none
2	--5rZFJBzhXlpcGkgMb51w	PrYKua8LhcYYjMmmlPPkhA	hmmm... Thai iced tea w/ simple sryyrup on the...	drink
3	--852IDNHxJLs5FTlzvMCQ	9IT4dXLpGAXSGiSQVdMLpA	NaN	none
4	--8N4jtEwQwGFYpc3F589A	s9IJSI3AgG0GqCS6hggXiA	Robyn feeding a mountain lion!	none

Even among these three labels, the distribution of the data was highly skewed:



To prepare our image data, we rescaled each image to be of size **96 x 96** and converted it from RGB to YUV format before storing it in matrix format. After this was done, we performed entry-wise normalization of the pixels on the train, validation, and test data such that for each pixel x_{ij},:

$$x'_{ij} = \frac{x_{ij} - \mu_{tr}}{\sigma_{tr}}$$

Where μ_{tr} and σ_{tr} are the mean and standard deviations of the mean and standard deviation across the training set.

B. Data Insights

Examining the correlation between the three target labels we chose (Figure 1) reveals some interesting information about our data. It may be noted in particular that the 'Good for Kids' and 'Good for Groups' labels are the most correlated, and their correlation is negative. Indeed, we'll see **later** in our analysis of the image captions that the most relevant words for 'Good for Groups' labeled places correspond to attributes such as "night", "alcohol", "pool", ... which of course do not correspond to places we would recommend for kids.

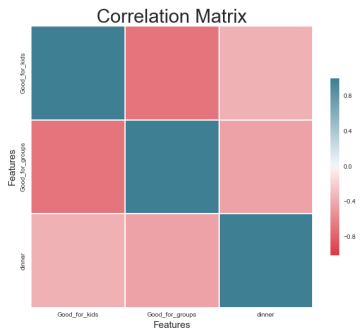


Fig. 1: Correlation matrix between the labels

IV. Global Framework

The global framework we designed is illustrated on Figure 2 In order to solve the fact that most of the picture we had had several labels (the same labels as the restaurant they were associated to), we first isolated the images from businesses that had only one true label. We trained a model on the captions of these images in order predict individual labels on the remaining images associated with multi labels businesses. This would turn our initial problem into a fully supervised problem with every picture only having now one label, that we believed would be the most relevant for that picture. We would then solve that supervised problem by training independently both an image and a caption based model. We then averaged the predicted probability from each model and attributed the label that had the maximum probability. From the single image level predictions, we could eventually predict the business level multiple attributes as follow: as soon as a business would have a picture with a given predicted label, we would predict that label for the business as well.

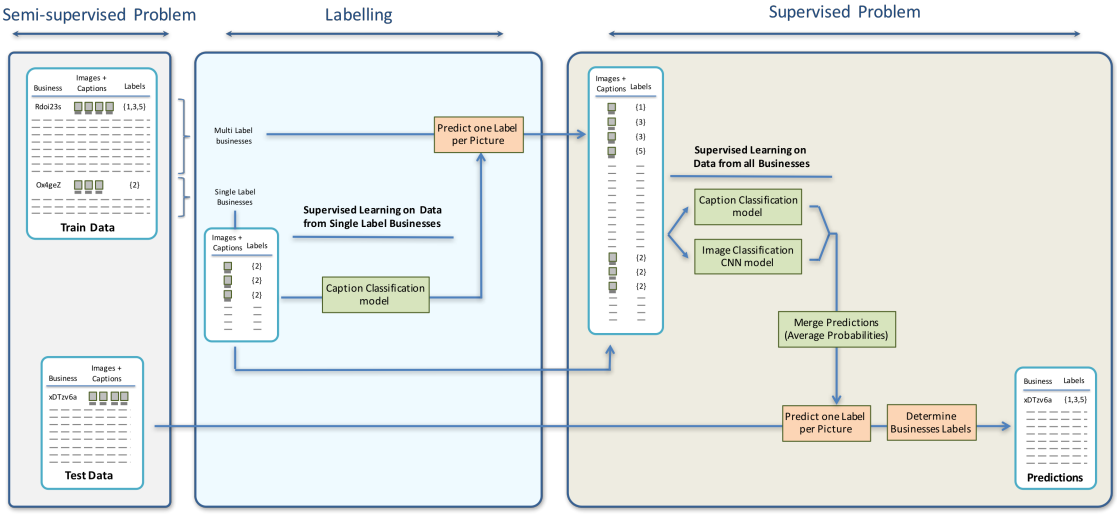
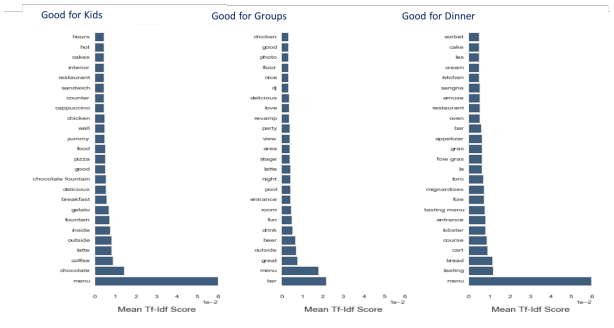


Fig. 2: Global Framework

V. Retrieving Individual Image Labels

By only using images that were associated to single label businesses, we could perform fully supervised classification. We used the captions on these images to train a model using tf-idf. We then used this model to determine a unique picture-level label for the pictures associated to businesses with multiple labels. The figure below displays top contributing words for each class (stop words removed):



We also tried various classifiers and among them, Linear SVC, SGD and Multinomial models gave the best accuracy:

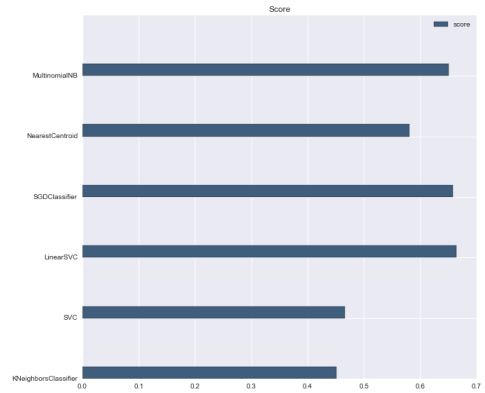


Fig. 3: Performance with various models

Our final results are shown below:

Classifier	MultinomialNB
Train Accuracy	0.8943
Test Accuracy	0.6662

VI. Image Classification via Caption & Image Analysis

Once we recovered individual labels for each image, we trained two fully supervised models separately before aggregating predictions from both to produce a final classification:

1) A text classifier on the images captions, using the exact same methodology as before
2) A Convolutional Neural Network
   We outline this stage of our model below:

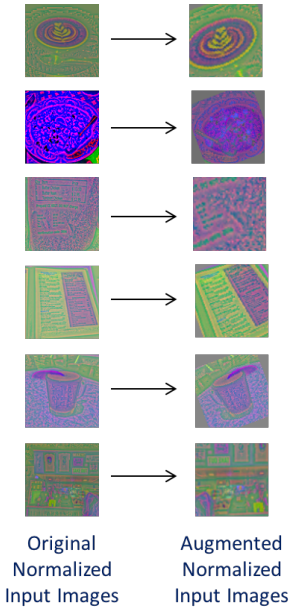
A. CNN Methodology

We implemented our convolutional neural network using the Torch package in the Lua programming language. We ended up using a ten layer model (7 convolutional layers + 3 linear layers), though we originally attempted a larger model and found that it did not work well with our data. A visualization of our network architecture can be found in the **appendix**.

The main issue that we faced when building our CNN was dealing with the skew of the data. After running our $\tilde{7}$0,000 images through the text classifier to get image-level labels, we had approximately 30,000 images in the 'Good for Groups' class, 22,000 in 'Good for Kids', and 700 in 'Good for Dinner'. We attempted to overcome this by performing selective data augmentation on the two classes with reduced size. That is, we randomly applied visual transformations to the smaller classes in our training data and added these transform images to our data set in order to increases the number of images for those classes. Our augmentation techniques are summarized below:

- **Rotation**: Rotate the input by some random angle
- **Scaling**: Randomly rescale image up or down before re-cropping to fit model input size
- **Translation**: Randomly shift image left, right, up, or down by some number of pixels
- **Contrast**: Modify the contrast of the input image
- **Horizontal Flip**: Horizontally flip input image

For each image, we set a 50-80% probability of applying a particular transformation. We experimented with many different augmentation sizes, and ultimately decided to add **1000** images to the 'Good for Kids' class and **600** images to 'Good for Dinner'. We did not want to add too many augmented images to the data set for fear of overfitting, and the large volume of images in the 'Good for Groups' and 'Good for Dinner' classes made the augmentation only have moderate success - if we had less total images than maybe it would have worked better. Some samples of our data augmentation are shown below:

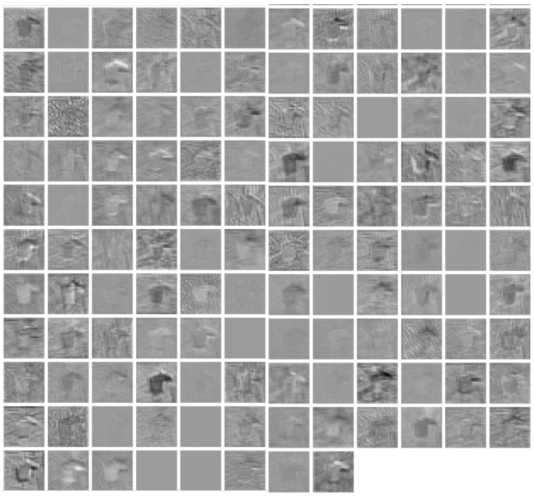


Data Augmentation

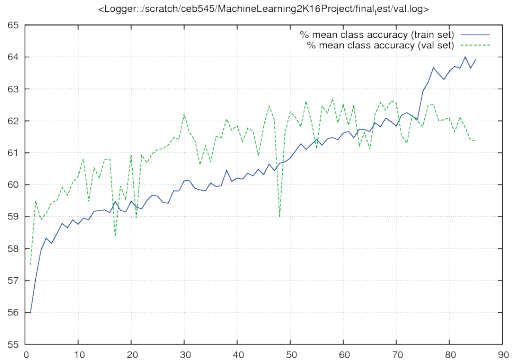
Our aim with the CNN was to extract spatially invariant, high-level features from the images that represent the very general classification of being good for groups, kids, or dinner. Each convolutional layer has three stages that work to accomplish this: convolution, application of a non-linearity, and pooling. The work done in each of these three stages is summarized below:

1) **Convolutional Stage**: In this stage, we compute the product of a learned "filter" (represented by a weights matrix) with local regions of the image. This product allows us to identify what sections of an image are relevant to what we are attempting to predict. At each convolutional layer we define the number of feature maps we want to output - this is how many filters we are applying over each image in this layer.
2) **Non-Linearity**: The application of a non-linearity to the data allows the network to learn beyond linear combinations of the data. Without applying a non-linearity to the output of the convolutional stage, we could represent all layers in the CNN as one linear combination.
3) **Pooling**: The pooling stage downsamples the output of the previous two stages - this is important for introducing spatial invariance and reducing computational costs.

The output of one of the convolutional layers for one of our images is shown below:



With our best model, we achieved **67.11%** accuracy on our training data and **60.21%** accuracy on our test data, which resulted after running our model for approximately 90 epochs. The convergence of the model's accuracy over time is shown in the figure below:



VII. Final Prediction of Photo Labels

Both our caption model and CNN assigned class probabilities for each photo. To create a final prediction for a particular photo, we average these two scores, and then took the maximum score of this average as the label for the photo.

VIII. Final Prediction of Business Labels

We assigned labels to each business based on the predicted labels of the images associated to that business. We formed the prediction for a business by taking the union of predictions across all images.

IX. Evaluation

We use Precision, Recall and F1 score to evaluate the performance of our model. Indeed, Precision is very important for our use case because customer satisfaction is crucial for recommendation platform. The satisfaction of customers won't be particularly affected if you do not recommend a certain place, whereas it would highly affect their trust in your service if you were for instance to recommend a good place for kids when in fact it is not.

The chart below shows our results in terms of true/false positives/negatives obtained with our model. As we can see, we were able to have a low number of false positives. Nevertheless, this was performed at the cost of a quite high number of false negatives

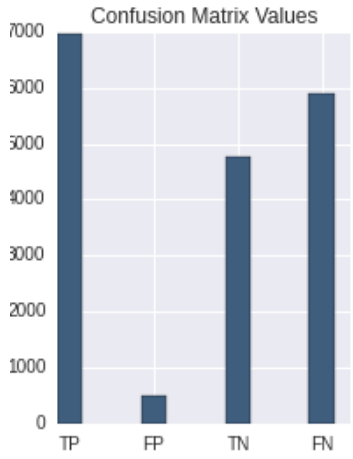


Fig. 4: Summary of Confusion Matrix values

Finally we used the F1 score to assess the overall performance of our model, which is computed as follow:

$$F1score = 2\frac{Precision.Recall}{Precision + Recall}$$

With:

$$Precision = \frac{TruePositive}{TruePositives+FalsePositives}$$

$$Recall = \frac{TruePositives}{TruePosivies+FalseNegatives}$$

We summarize our final results below:

Precision	0.9318
Recall	0.5405
F1 score	0.6841

X. Conclusion & Future Work

Ultimately, we were able to improve on our random baseline with our joint CNN and caption-based model. If we had more time, we would have used another decision function to infer business labels from our prediction photo labels, instead of just taking the union of predicted labels across all photos. We would have also explored other methods to combine our predictions from our image and caption based models. Finally, we would have included our implementation of αSVM [1], which we believe would have helped us in our attempts to generate image-level labels.

XI. Acknowledgments

We would like to thank our advisor Kush R. Varshney for the strong support and always very helpful advice he provided us throughout our project. He always provided us very thoughtful responses to our questions and was had great suggestions for us throughout our project.

References

[1] αSVM for Learning with Label Proportions http://felixyu.org/pdf/felix_yu_pSVM_icml2013.pdf

XII. Appendix

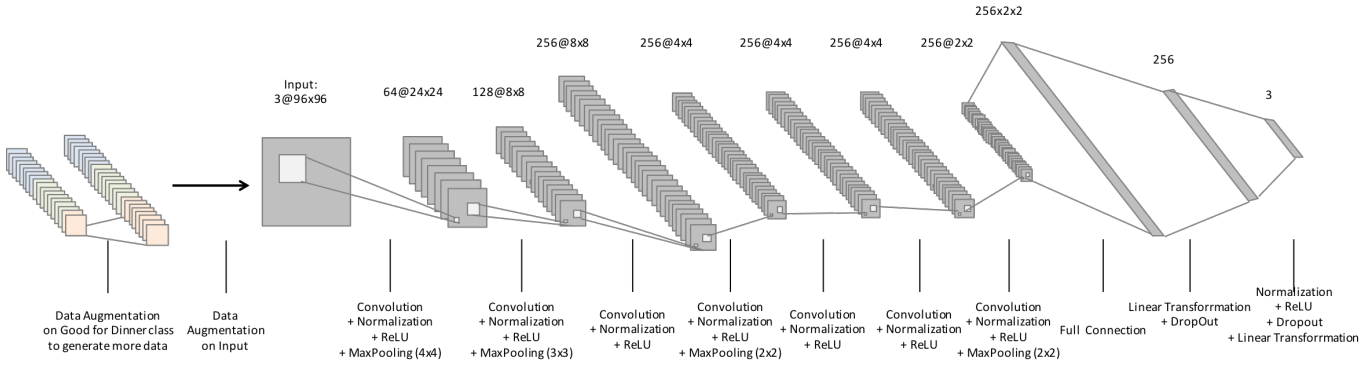


Fig. 5: Architecture of the Convolutional Neural Network we used

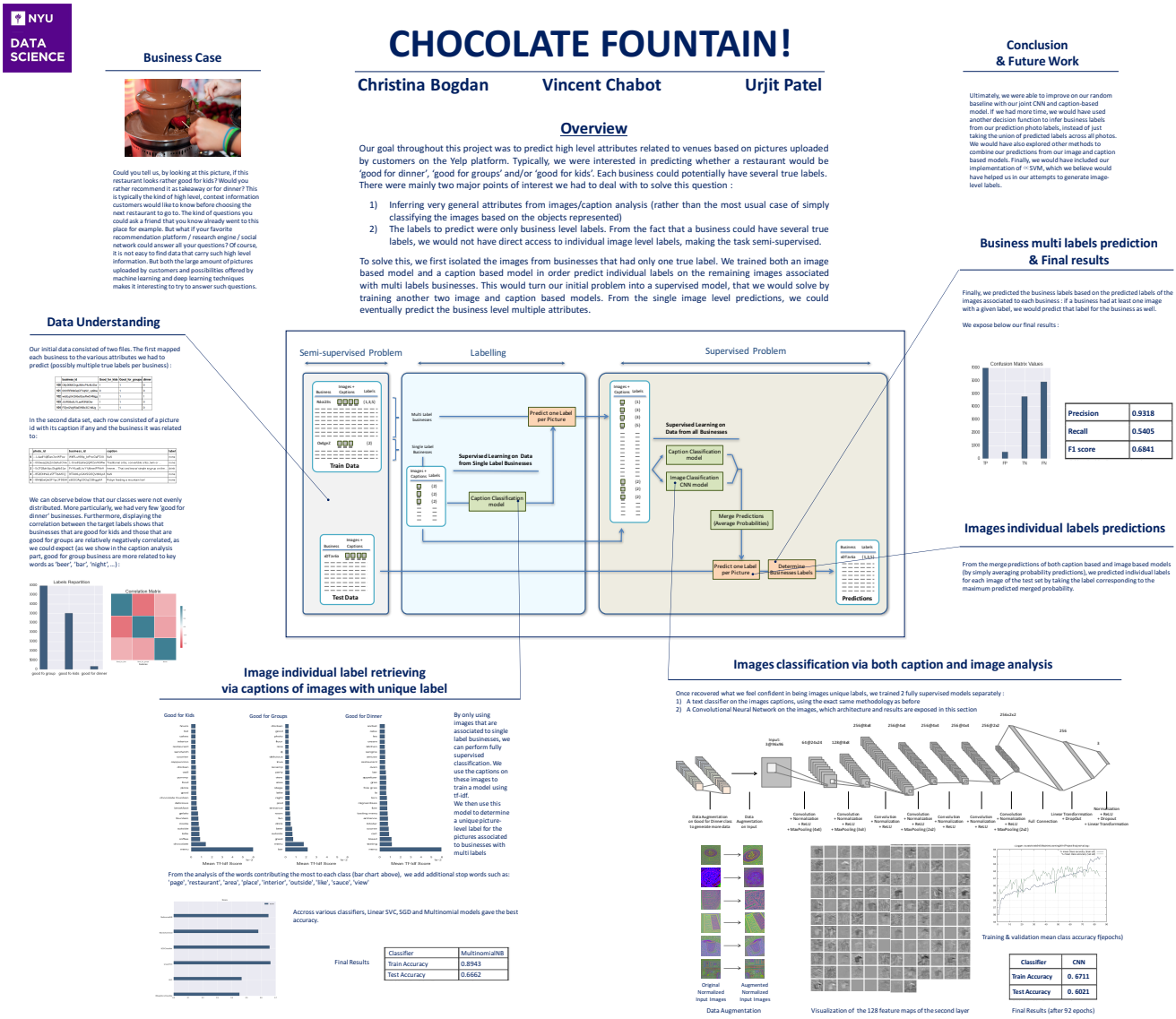


Fig. 6: Final Poster for the NYU CDS Projects Presentation Event