Supervised Learning to classify city from urban scenes

Rasesh Kumar Rout

May 5, 2020

Contents

1 Abstract **2**

2 Introduction and Goals **2**

3 Related Work : A Brief Background **2**

4 Brief Description of Type of Work **5**

5 Data **6**
5.1 Why Dataset size should be enough? 7

6 Methods **8**
6.1 Convolutional Neural Network 8
6.2 Bag of Visual Words 9

7 Results and Visualization **10**
7.1 ConvNet 10
7.2 Bag of Visual Words 16

8 Discussion and Future Work **21**

9 Timetable **21**

1 Abstract

We present an approach to use the high definition aerial and street view images of urban areas of big cities like New York, San Francisco and Rome and classify those images to the corresponding cities. We propose to extract features from such images and feed it into a Convolutional Neural Network to predict the city. We will evaluate the performance of our ConvNet using the accuracy of classification and the F1 scores it generates. As an alternative approach, we extract SURF features and follow a bag of visual words method to classify the images as well.

2 Introduction and Goals

Geo-tagging of images has always been a challenging problem in the area of Computer Vision and Pattern Recognition. The idea is to use the cameras that record/snap the images and find the angles/direction from where the image could have been taken to predict the geographical location based on the camera's location. In my project, I won't be focusing on the geo-tagging rather extracting features out of these images and trying to predict what city it is from.

As part of this project, I want to develop an approach that is able to use the extracted features from the images to predict the City of the Urban Scene.

3 Related Work : A Brief Background

While conducting survey on the work done on the dataset, I was able to figure out that most of the authors use the Urban Scenes dataset for extracting facades from the building images and using them for matching or geo-tagging purposes but not much has been done on the classification front.

The authors of [1] proposed a regularity based approach facade matching between the Aerial and Street view images from the Google Urban Dataset. The authors calculate a motif cost function that takes into account various contexts like shape, color, texture, and location proximity whose maximum score gives the best matching for an image.

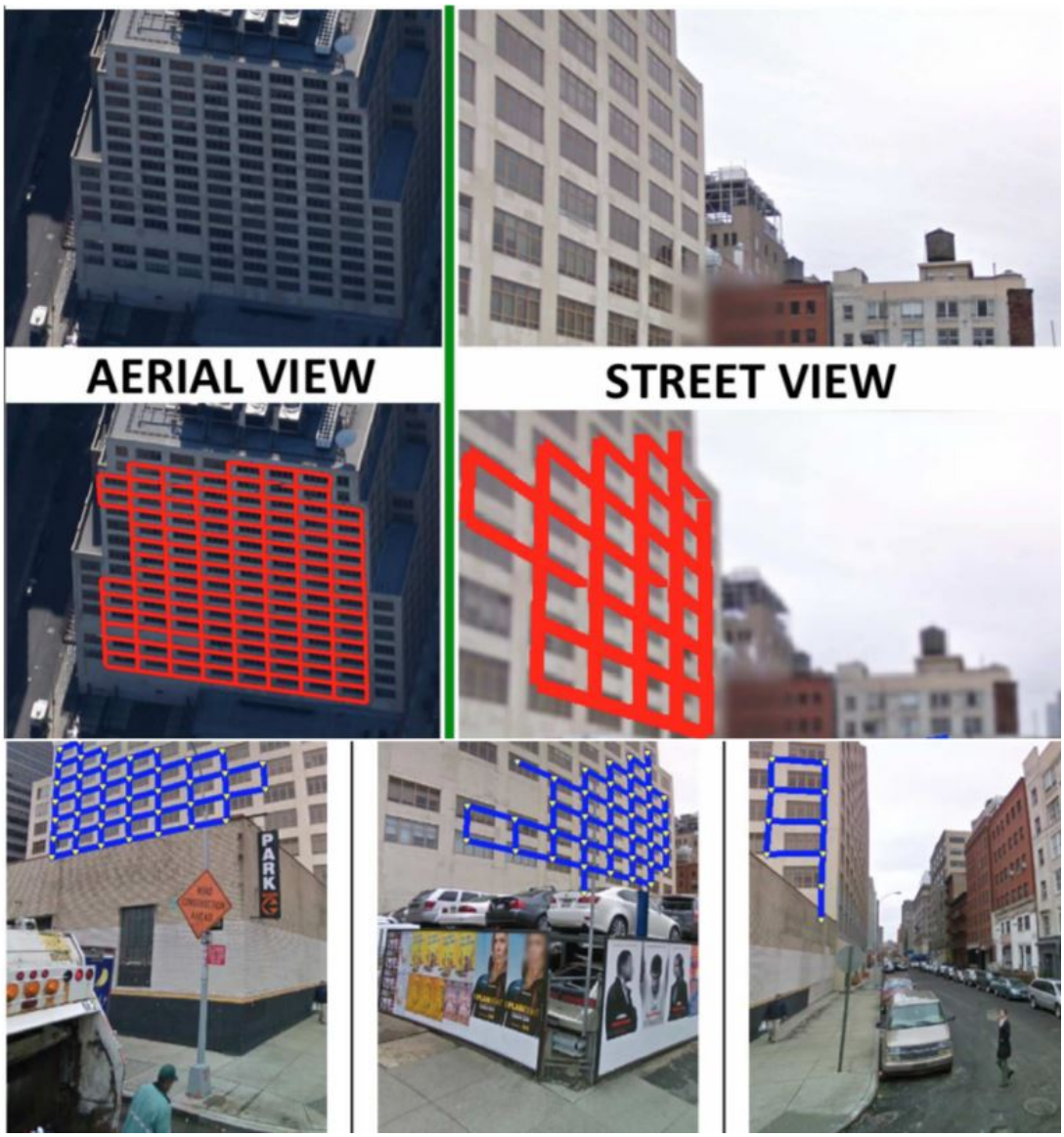


Figure 1: Regularity-Driven Building Facade Matching between Aerial and Street Views

The work done by authors of [2] focuses on detecting repeated patterns in the building facades to find location and direction of the cameras and geo-tag those images accordingly. The authors use a feature based RANSAC method to identify lattices which exhibit scale-invariant features.

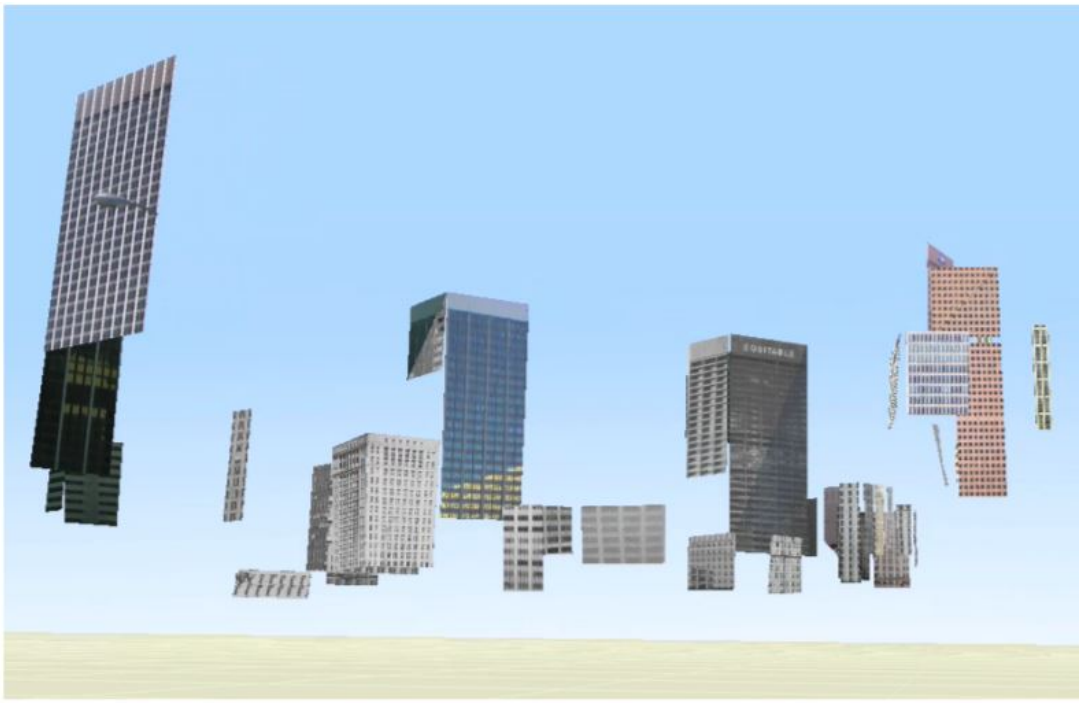


Figure 2: Detecting and Matching Repeated Patterns for Automatic Geo-tagging in Urban Environments

The authors of [3] built a custom Generative Adversarial Network(GAN) to learn various architecture styles of different cities and generated pictures of buildings that do not exist in reality.

Figure 3: City-GAN: Learning architectural styles using a custom Conditional GAN architecture

Taking inspiration from the works of [3], I plan on building a Convolutional Neural Network from scratch that is able to learn different features from the cities present in my dataset. On top of that, I'll be implementing a Bag of Visual Words method consisting of SURF features to aid in the classification as a 2nd method of approach.

4 Brief Description of Type of Work

The proposed methods would be Supervised learning method where the end goal would be classification of the images in the dataset into the city they belong to.
First method will be a Convolutional Neural Network that learns the training images made by augmenting the data. The accuracy is judged based on how the network classifies the unaugmented original data.
Second method will use SURF method to extract features from the images and a bag of visual words will be formed out of it via K-Means clustering. A linear SVM will be used to classify the images based on these features. The accuracy is judged based on how the network classifies the unaugmented original data.

To summarize, the approaches will be Supervised learning methods to perform classification of the input data into 3 classes.

5 Data

The dataset I'll be using is the Google Urban Scenes dataset. The datset consists of high resolution images of the Aerial and Street View of different urban areas of major cities.

We have data taken for three different cities, namely:

- New York
- San Francisco
- Rome

For each city, we have:

- nearly 500 Aerial view images
- nearly 100 Street view images

Each image is of size 5616x3744.

Figure 4: Aerial View Image of New York City

Figure 5: Aerial View Image of Rome

5.1 Why Dataset size should be enough?

I feel this dataset is sufficient for our Machine Learning task because we have just 3 classes that we need to classify the images into, namely the three cities. Each class has approximately 600 images each taken from Aerial and Street viewpoints. Also we are performing training a CNN and the size of data we have should be enough to get a good trained model.

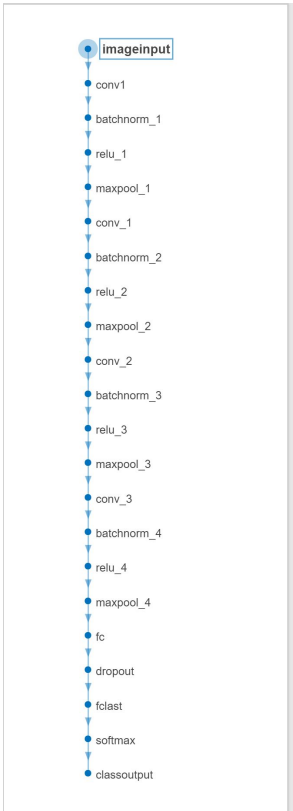
Due to unaivailibility of full data, I used augmentation to increase the size of training data. I was provided with Aerial View images(25 from New York, 15 from San Francisco and 14 from Rome). To accomplish the Machine learning task, I used 14 images from all classes, which is the minimum number of any of the classes. I performed augmentation on top of them to generate 15 times the initial size and performed the tasks.

6 Methods

6.1 Convolutional Neural Network

The approach I'm proposing is a Convolutional Neural Network based image classification. The reason for going with this approach is that CNNs have proven to give good accuracy for image classification [4] problems as the number of parameter keep growing for Neural Networks with the number of layers by introducing a sliding window of smaler dimensions over the data.
The idea is to feed the the Urban Scenes images which have been augmented to the train the CNN. Once the training is complete we use the original images as the testing data and check the classification accuracy for each class.

The Architecture of the CNN used is as follows:
$Input \rightarrow [CONV \rightarrow BatchNorm \rightarrow ReLU \rightarrow MaxPool] * 4 \rightarrow FC \rightarrow Dropout \rightarrow FC \rightarrow SoftMax \rightarrow Classification$
The first two Conv layers have 40 5x5 filters and second two have 80 5x5 filters. The first Fully Conected layer has 100 activations. In the classification layer, we use cross entropy as the loss function.
The network layers description with their learnable parameters are shown in the below figure.



ANALYSIS RESULT

	Name	Type	Activations	Learnables
1	imageinput 234x351x3 images with 'zerocen…	Image Input	234×351×3	-
2	conv1 40 5x5x3 convolutions with strid…	Convolution	234×351×40	Weights 5×5×3×40 Bias 1×1×40
3	batchnorm_1 Batch normalization with 40 cha…	Batch Normalization	234×351×40	Offset 1×1×40 Scale 1×1×40
4	relu_1 ReLU	ReLU	234×351×40	-
5	maxpool_1 2x2 max pooling with stride [2 2]…	Max Pooling	117×175×40	-
6	conv_1 40 5x5x40 convolutions with stri…	Convolution	115×173×40	Weights 5×5×40×40 Bias 1×1×40
7	batchnorm_2 Batch normalization with 40 cha…	Batch Normalization	115×173×40	Offset 1×1×40 Scale 1×1×40
8	relu_2 ReLU	ReLU	115×173×40	-
9	maxpool_2 2x2 max pooling with stride [2 2]…	Max Pooling	57×86×40	-
10	conv_2 80 5x5x40 convolutions with stri…	Convolution	55×84×80	Weights 5×5×40×80 Bias 1×1×80
11	batchnorm_3 Batch normalization with 80 cha…	Batch Normalization	55×84×80	Offset 1×1×80 Scale 1×1×80
12	relu_3 ReLU	ReLU	55×84×80	-
13	maxpool_3 2x2 max pooling with stride [2 2]…	Max Pooling	27×42×80	-
14	conv_3 80 5x5x80 convolutions with stri…	Convolution	25×40×80	Weights 5×5×80×80 Bias 1×1×80
15	batchnorm_4 Batch normalization with 80 cha…	Batch Normalization	25×40×80	Offset 1×1×80 Scale 1×1×80
16	relu_4 ReLU	ReLU	25×40×80	-
17	maxpool_4 2x2 max pooling with stride [2 2]…	Max Pooling	12×20×80	-
18	fc 100 fully connected layer	Fully Connected	1×1×100	Weights 100×19200 Bias 100×1
19	dropout 25% dropout	Dropout	1×1×100	-
20	fclast 3 fully connected layer	Fully Connected	1×1×3	Weights 3×100 Bias 3×1

Figure 6: CNN architecture

We have a total of 210 images from each class after performing 15 random rotational and scaling augmentation on the 14 base images which have been scaled down to avoid

the out of memory GPU errors.

For training the ConvNet, we split the data into Training, Validation and Testing in the percentage ratio of 70:20:10.

We perform further testing on the unaugmented images as well(14 from each class).

The data splitting and training is done for 10 random splits and the mean accuracy and results are taken note of. The initial learning rate was set at 0.0005. A single training cycle was run for 15 epochs with a batch size of 10.

6.2 Bag of Visual Words

In this approach, we propose to extract SURF(Speeded-Up Robust Features) features from the input images and feed them to a Support Vector Machines for classification by creating a bag of visual words. The reason for selecting the SURF feature extraction is the insensitivity of the method to different orientations of the images giving highly distinctive features and also it is faster than SIFT(Scale-invariant Feature transform) feature extraction. Our dataset has images taken at various angles from Aerial view and I feel SURF fits the bill here. The steps in this approach are outlined as [5]:

- Step 1: Prepare data

  We use the same augmented images as the previous CNN approach. 210 augmented images for each of the three classes. The data is split into Training and Testing as 70:30. We do 5 random splits of the data and evaluate the mean results finally.

- Step 2: Feature extraction

  In this step, we use SURF feature extraction method and extract robust features from the input images and select the 50% strong features out of the set of all extracted features. We narrow down to the number 50 by experimenting on various percentages of features and how fast the K-Means clustering converges for them.

- Step 3: Create bag of visual features

  In this step we use k-means clustering with 500 clusters to group the previously extracted SURF features into clusters. Each cluster defines an exclusive feature/visual word.

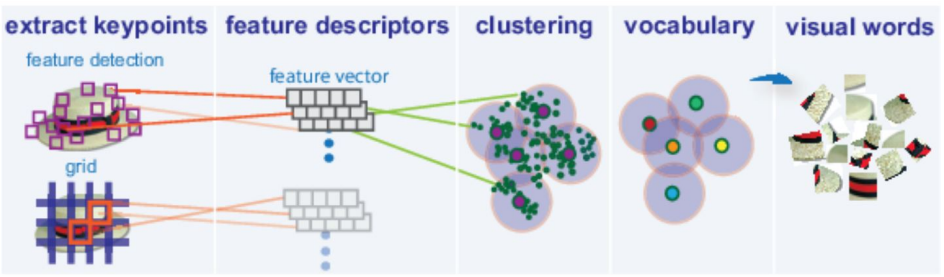


Figure 7: Creating bag of visual words

- Step 4: Classification

  The we train a multi-class(3-class) classifier using the error-correcting output codes (ECOC) framework with binary support vector machine (SVM) classifiers. The classifier function uses the bag of visual words created in the previous step to encode images in the image set into the histogram of visual words. The histogram of visual words are then used as the positive and negative samples to train the classifier.

- Step 5: Note the test accuracy

  Mean accuracy and other results are noted down for the 5 random splits of data.

7 Results and Visualization

7.1 ConvNet

The accuracy vs loss plot of one of the training iterations is shown below:

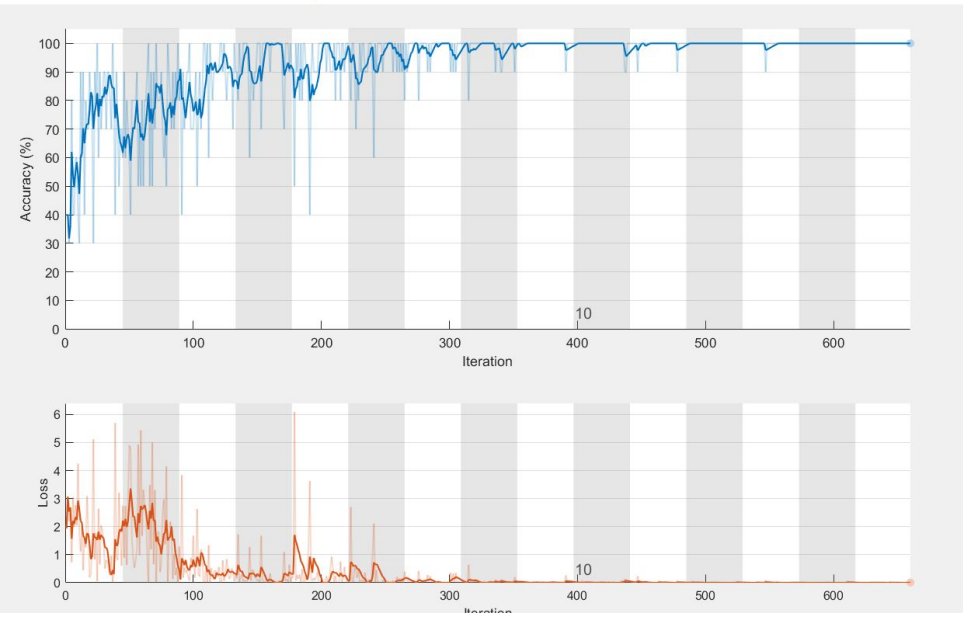


Figure 8: Plot of accuracy vs loss over 15 epochs

Below figure shows the runtime and full details of one iteration of training:

```
Loading Train, Test and Validation Filenames and Label Data...Done in 0.72 seconds
Loading original image Filenames and Label Data...Done in 0.06 seconds
Training on single GPU.
Initializing input data normalization.
|=========================================================================================|
|  Epoch  |  Iteration  |  Time Elapsed  |  Mini-batch  |  Mini-batch  |  Base Learning  |
|         |             |   (hh:mm:ss)   |   Accuracy   |     Loss     |      Rate       |
|=========================================================================================|
|       1 |           1 |       00:00:03 |       40.00% |       1.9331 |          0.0005 |
|       2 |          50 |       00:00:09 |       50.00% |       5.3643 |          0.0005 |
|       3 |         100 |       00:00:15 |       90.00% |       0.5450 |          0.0005 |
|       4 |         150 |       00:00:20 |      100.00% |       0.0946 |          0.0005 |
|       5 |         200 |       00:00:26 |       90.00% |       0.0932 |          0.0005 |
|       6 |         250 |       00:00:31 |      100.00% |       0.0284 |          0.0005 |
|       7 |         300 |       00:00:36 |      100.00% |       0.0018 |          0.0005 |
|       8 |         350 |       00:00:42 |      100.00% |       0.0064 |          0.0005 |
|      10 |         400 |       00:00:48 |      100.00% |       0.0096 |          0.0005 |
|      11 |         450 |       00:00:53 |      100.00% |       0.0003 |          0.0005 |
|      12 |         500 |       00:00:58 |      100.00% |       0.0019 |          0.0005 |
|      13 |         550 |       00:01:04 |      100.00% |       0.0065 |          0.0005 |
|      14 |         600 |       00:01:09 |      100.00% |       0.0031 |          0.0005 |
|      15 |         650 |       00:01:15 |      100.00% |       0.0007 |          0.0005 |
|      15 |         660 |       00:01:16 |      100.00% |       0.0007 |          0.0005 |
|=========================================================================================|
Trained in in 86.88 seconds
Training on single GPU.
```

Figure 9: One training iteration data

The below table shows the various mean accuracy results from the ConvNet approach for 10 random splits of Training.

Dataset	Mean Accuracy	Mean Std. Dev.
Training(Augmented)	98.90	0.0115
Validation(Augmented)	93.39	0.1270
Testing(Augmented)	88.91	0.1877
Testing(Original)	82.67	0.2916

Table 1: Accuracy results

The classification matrices are shown below:

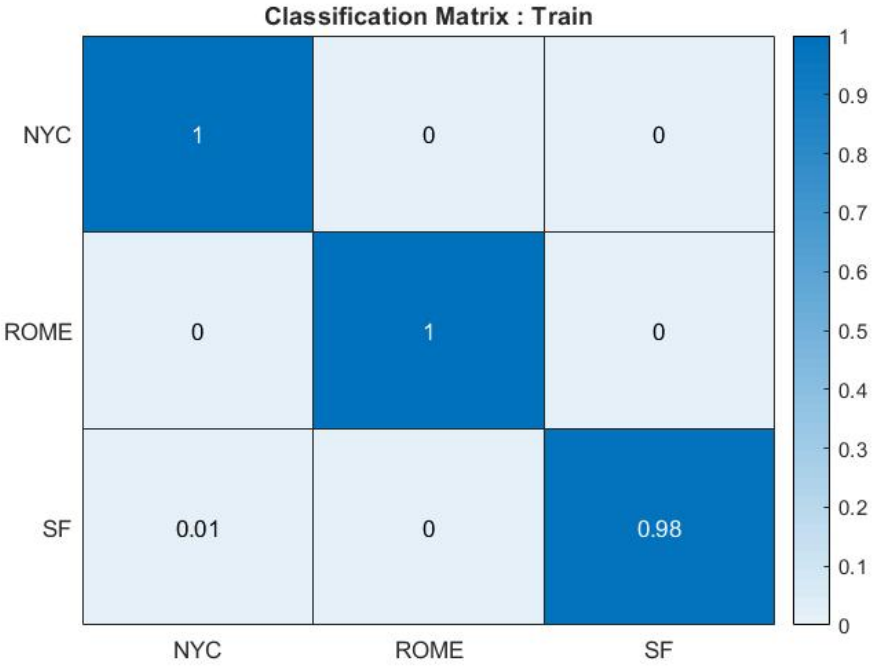


Figure 10: ConvNet: Training classification

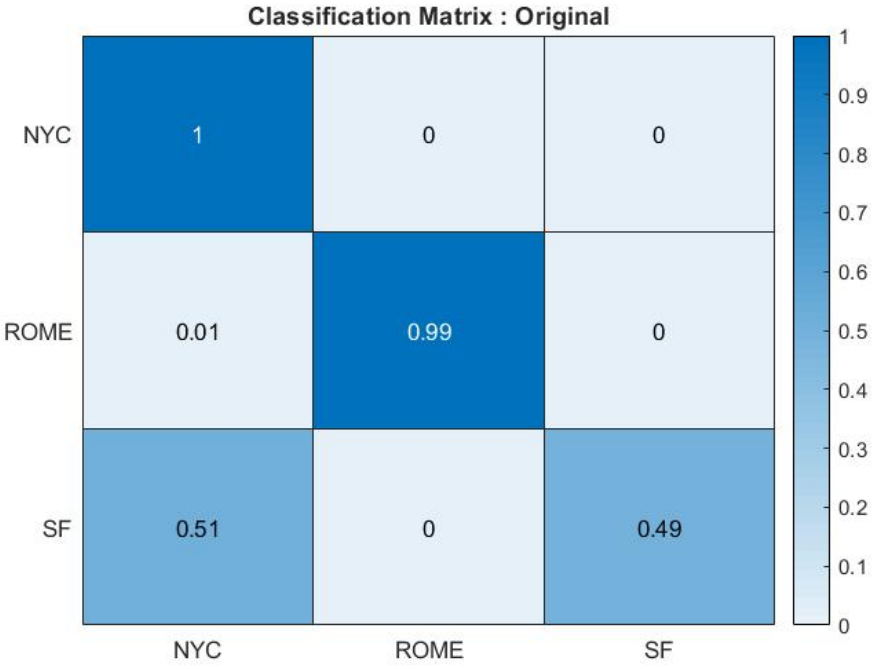


Figure 11: ConvNet: Testing classification

As we can observe from the Testing classification matrix, New York and Rome were classified properly whereas San Francisco had a bad classification rate as most of the San

Francisco images were wrongly classified as New York. The training performed really good on the augmented dataset.

Class	Accuracy
NYC	100
Rome	99
San Francisco	49

Table 2: Mean Accuracy results per class on original images

Class	Precision	Recall
Rome vs NYC	1	0.99
SF vs Rome	1	0.949
SF vs NYC	1	0.49

Table 3: Precision and Recall values for each class combination on Original images

The tSNE visualization of the last Fully Connected layer for Training and Testing gives us explanation of the above results as we can see distribution of datapoints. It's clearly visible that in the testing tSNE distribution, Rome is well separated from the other two classes whereas NYC and SF have some overlap.

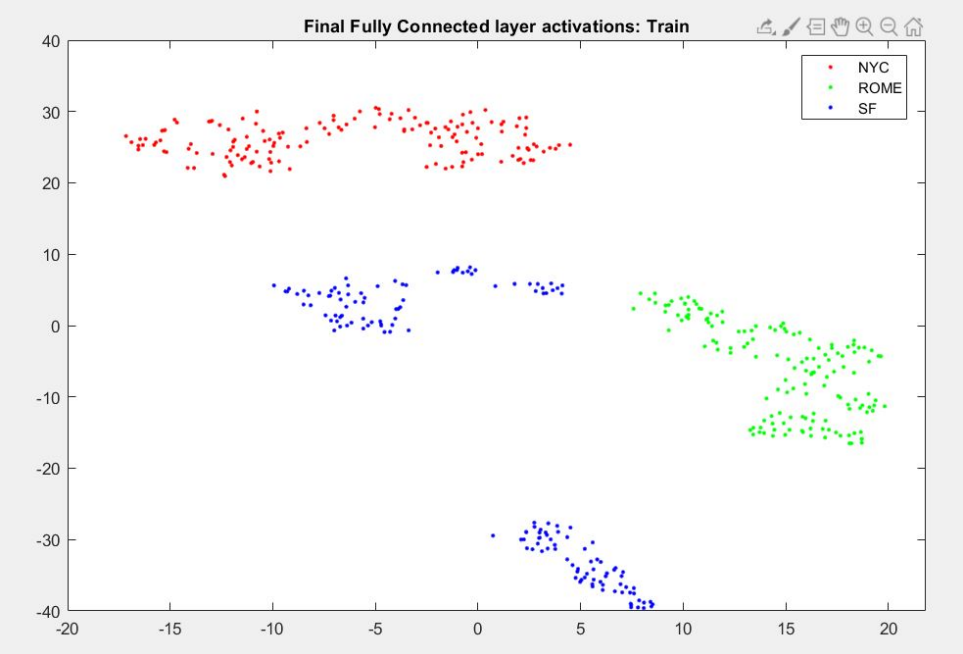


Figure 12: ConvNet: Training tSNE

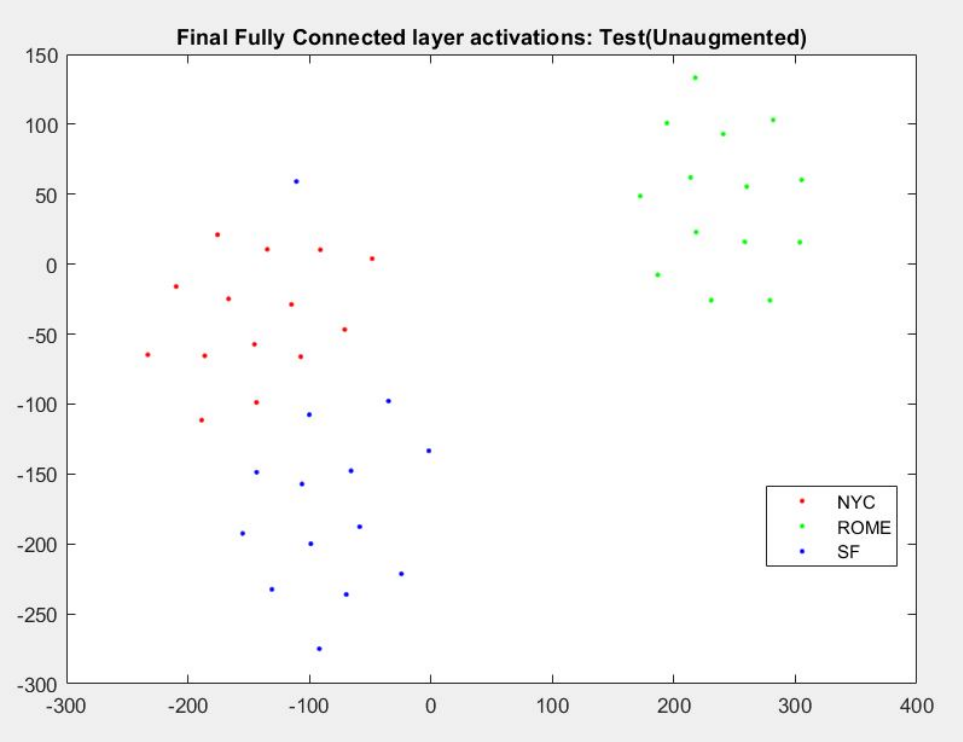


Figure 13: ConvNet: Testing(Original) tSNE

To understand what kind of features are learned in the network, we visualized three of the conv layers in our network, the first, third and fourth ones. The below figures show the strongest activation channels from each of the three conv layers.

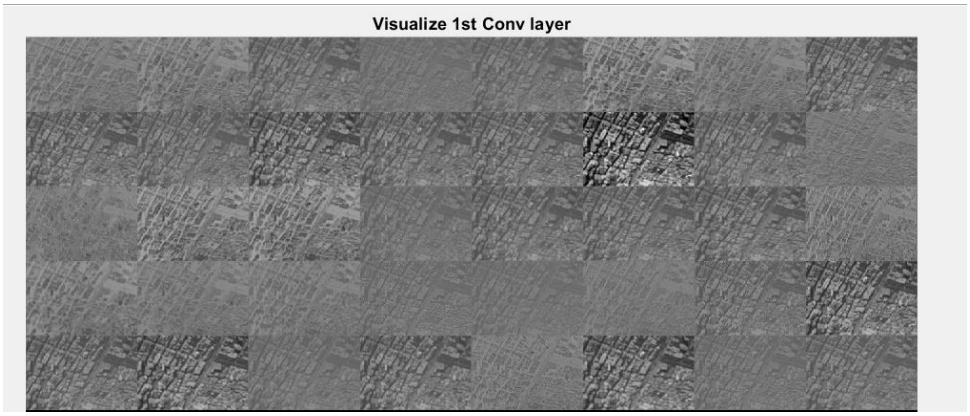


Figure 14: 1st Conv layer all activation channels

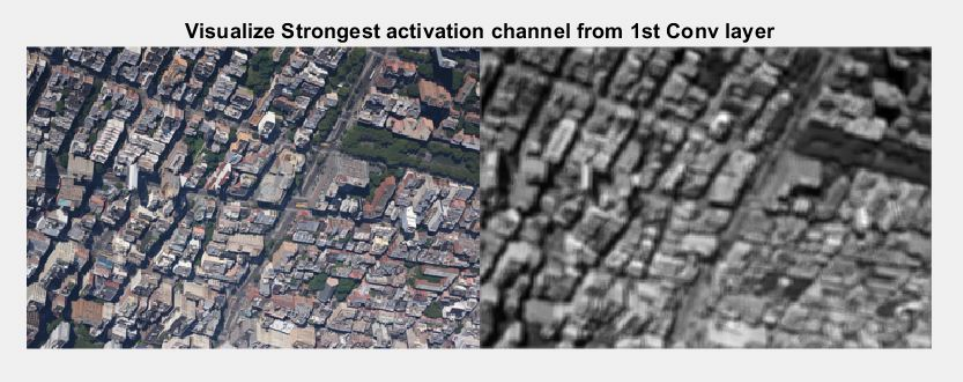


Figure 15: 1st Conv layer strongest activation

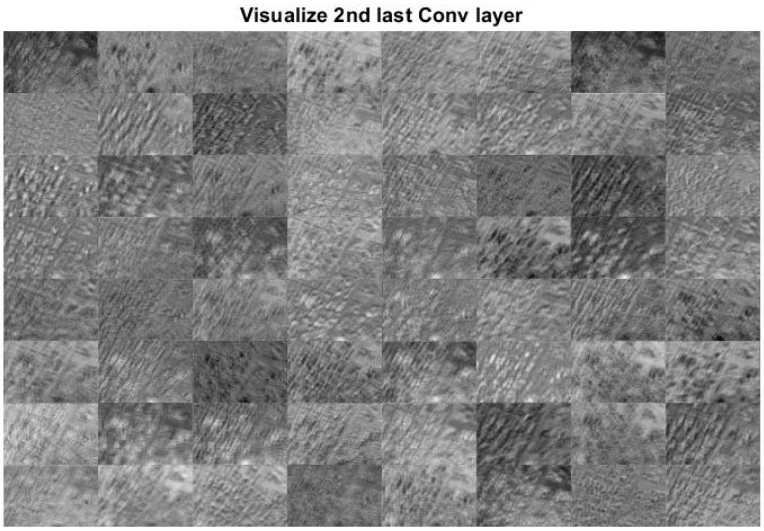


Figure 16: 2nd last Conv layer all activation channels

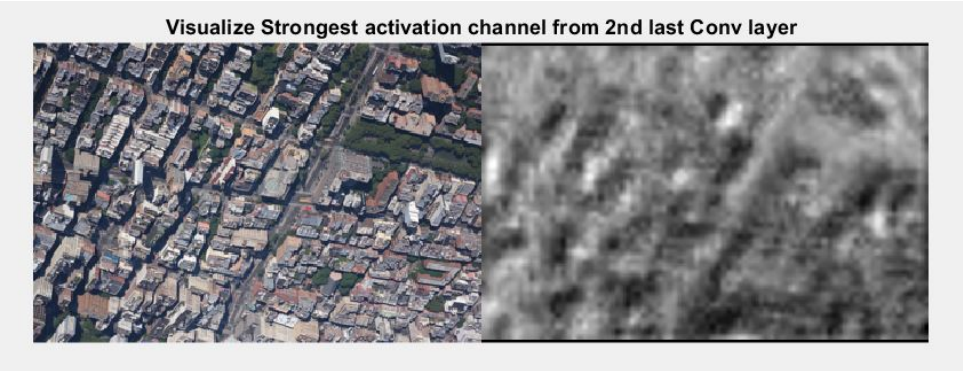


Figure 17: 2nd last Conv layer strongest activation

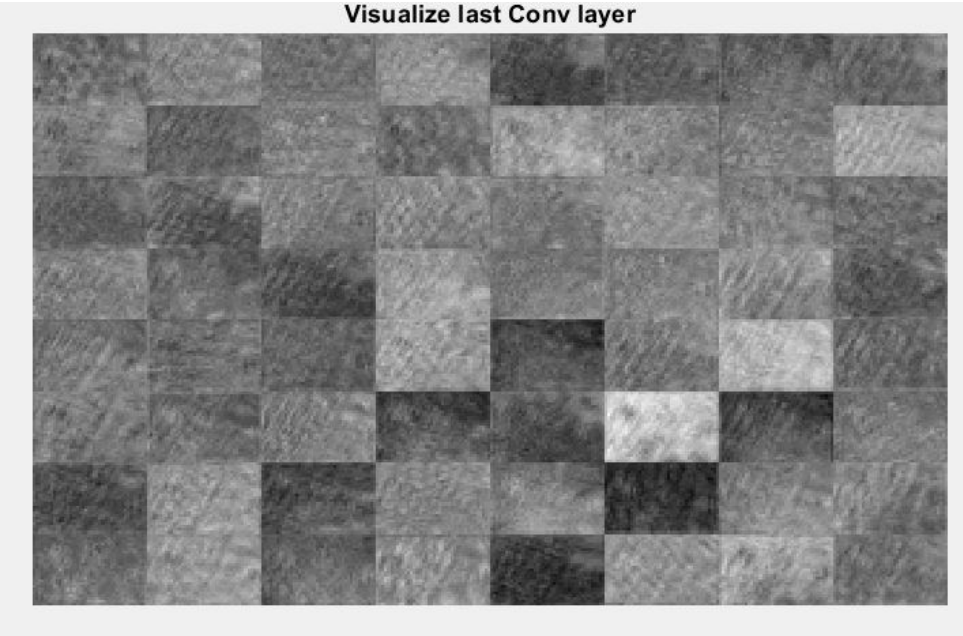


Figure 18: Last Conv layer all activation channels

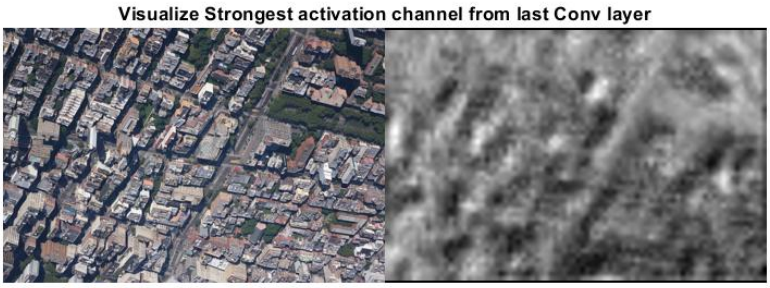


Figure 19: Final Conv layer strongest activation

As we can see from the strongest activation channels of the three conv layers in our ConvNet, the network is trying to learn the building structures more and more as we go into deeper networks. The network tends to learn the placement of these building structures in every input image for a city trying to learn more distinctive features.

7.2 Bag of Visual Words

Visualization of the kind of features extracted from the images via SURF is shown below:

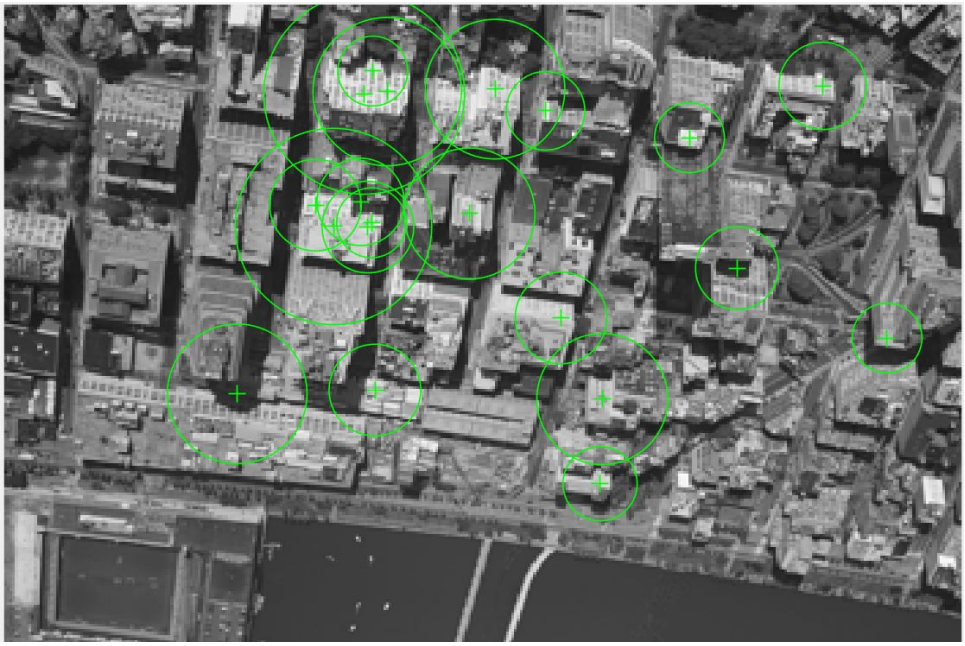


Figure 20: Extracted Surf features

As we can see in the above figure, the SURF method extracts the features from mainly the building structures and facades that are visible from Aerial View. The extra objects like water bodies aren't selected from the figure.

For finding the how much fraction of the strong features extracted via SURF gives good accuracy and converges faster for the clustering step. The resulting values and histograms showing the distribution of features in 500 clusters made by K-Means can be seen as below:

% Strong features	Accuracy	Std Dev	Converge in iterations
30	79	0.3109	48
50	81	0.2707	15
80	76	0.2970	14
90	69	0.3205	43

Table 4: Strong features selection

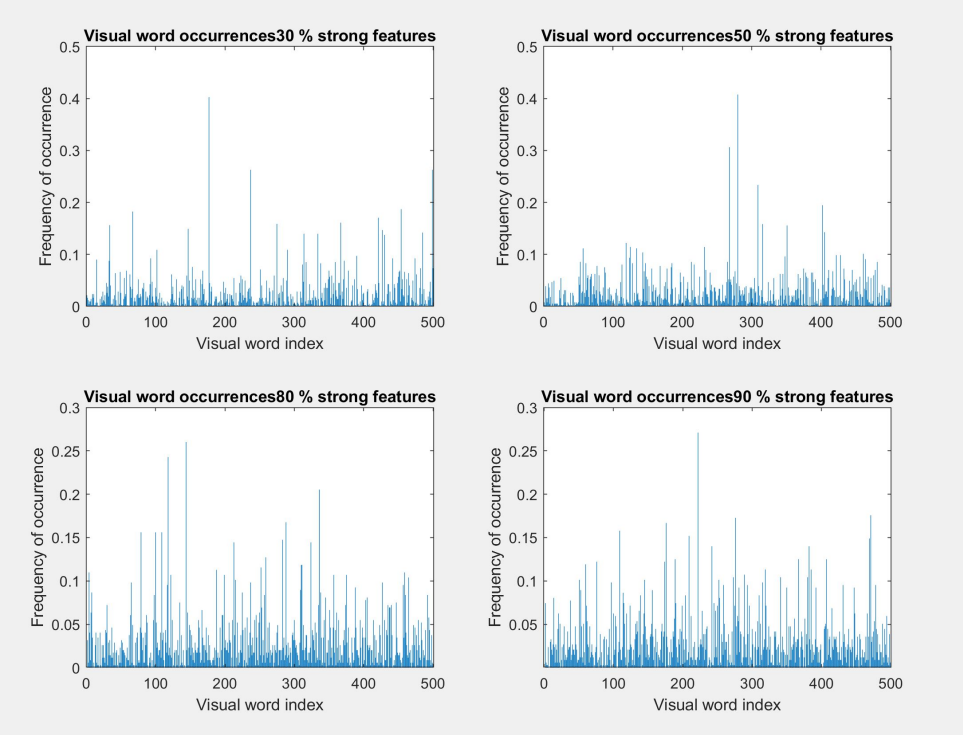


Figure 21: Histogram of visual feature bag

Based on the results above, we select 50% strong features from the SURF extracted features to do classification. For the k-means used to create cluster of visual words, we select value of 500 clusters. The results based on the parameters are tabled below for 5 random splits:

Data	Accuracy	Std Dev
Training(Aug)	92.36	0.1269
Testing(Aug)	88.15	0.1837
Testing(Orig)	76.67	0.3215

Table 5: Mean Accuracy results

The classification matrices for training and testing are shown as below:

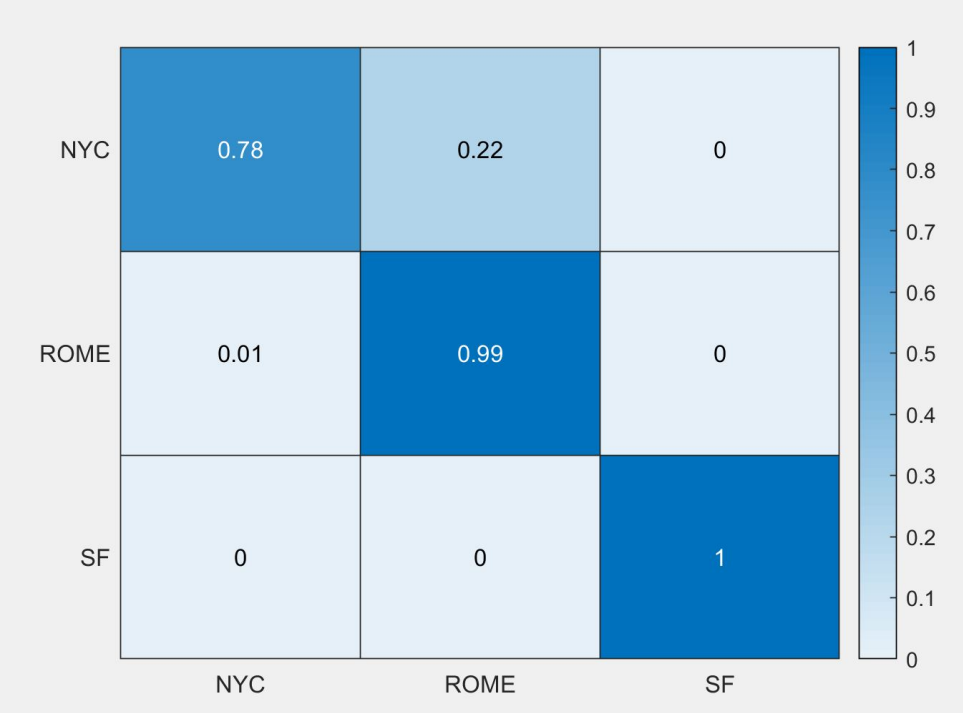


Figure 22: Bag of Visual Words: Training classification

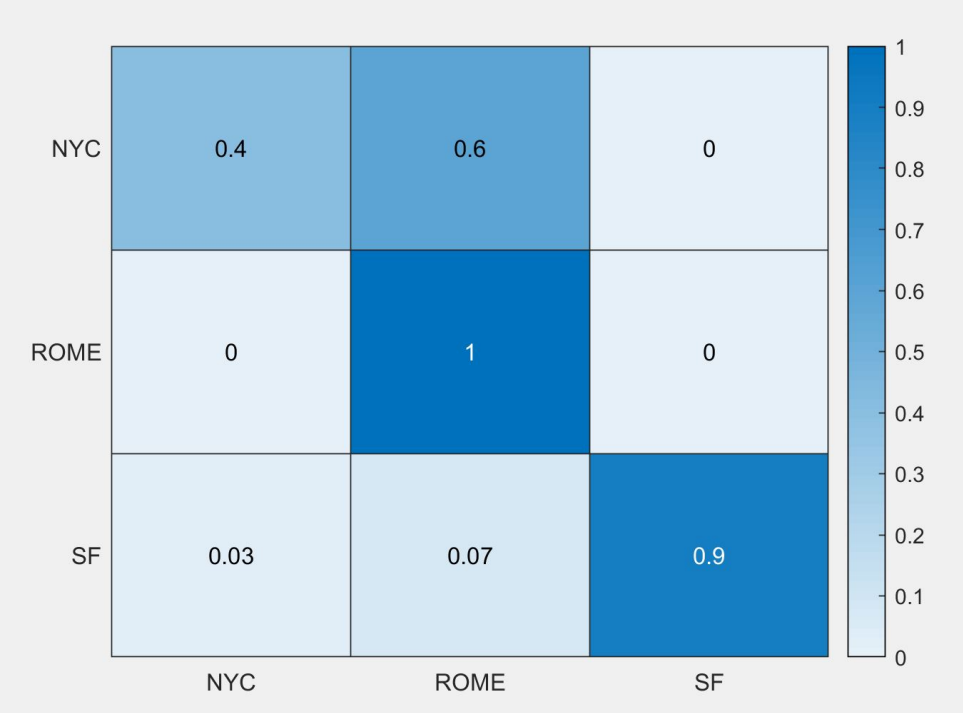


Figure 23: Bag of Visual Words: Testing classification

As we can observe from the classification matrices in Training with the augmented data, we were able to achieve good classification rates but in testing on the original images, we get good classification classes Rome and San Francisco whereas New York

City was way off in classification and was commonly wrongly classified as Rome. The individual class accuracy are reported below:

Class	Accuracy
NYC	40
Rome	100
San Francisco	90

Table 6: Mean Accuracy results per class

Class	Precision	Recall
Rome vs NYC	0.625	1
SF vs Rome	0.9	0.926
NYC vs SF	0.933	0.4

Table 7: Precision and Recall values for each class combination on Original images

8 Discussion and Future Work

The following observations can be documented from our work:

- The city classification from urban scenes is not a trivial problem.
- While architectural and visual wise Rome was way different, New York and San Francisco had similar structures.
- Our ConvNet architecture slightly edges out the Bag of Visual Words approach.

Challenges faced:

- The original large image sizes were resulting in GPU out of memory errors because of which I had to scale it down.
- Initial proposed approach of facade extraction was not completed because of unavailability of Ground Truth data where we had to manually label the facades to create a ground truth for each image.

Future work that can be done:

- Use facade extraction as originally intended and let the network learn the facade features.
- Derive a proper ranking mechanism to choose which facades should be used for the task as facades from buildings tend to be similar in texture.

9 Timetable

The final timetable over the course of completion of the project is shown below:

Table 8: A Simple weekly timetable for the project.

Checkpoint	Info
Checkpoint 1 (4/14):	Implement proposed CNN
Checkpoint 2 (4/21):	Implement the SURF feature extraction and bag of Visual Words approach.
Checkpoint 3 (4/24):	Fine tune the network and classification.
Checkpoint 4 (4/27):	Verify the classification results
Checkpoint 5 (4/28):	Final deliverable!

bottomrule

References

[1] M. Wolff, R. T. Collins, and Y. Liu, "Regularity-driven building facade matching between aerial and street views," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1591–1600, 2016. 2

[2] G. Schindler, P. Krishnamurthy, R. Lublinerman, Y. Liu, and F. Dellaert, "Detecting and matching repeated patterns for automatic geo-tagging in urban environments," *2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–7, 2008. 3

[3] M. Bachl and D. C. Ferreira, "City-gan: Learning architectural styles using a custom conditional gan architecture," *CoRR*, vol. abs/1907.05280, 2019. [Online]. Available: http://arxiv.org/abs/1907.05280 4, 5

[4] Wikipedia contributors, "Convolutional neural network," https://en.wikipedia.org/wiki/Convolutional_neural_network, [Online; accessed 23-March-2020]. 8

[5] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," *Workshop on Statistical Learning in Computer Vision*. 9