Landmark Recognition Using CNN and Feature Matching

Ruiqi Wang
Group Member: Ruiqi Wang, Haoda Li

December 2019

1 Introduction

When going over old albums, it can be difficult for people to recall the names of specific tourist attractions visited years ago. An automatic landmark recognition system would help people classify their photos and label the photos containing landmark. In addition, it would allow people to retrieve more information when they find images of some interesting places they want to go.

1.1 Problem

The input to our landmark recognition system is an image of any kind of objects. The output of our system is a label indicating whether the image contains a landmark or not, and if the image contains a landmark, providing the class(landmark_id) of the input image.
The source code is available in our Github Repository. [1]

1.2 Related Work

Google Landmark Dataset v2 [6] is currently the most diverse and comprehensive landmark dataset, including over 5 million images of more than 200 thousand different landmarks. A subset of this dataset is selected as our training and test dataset.
Image classification has been a popular research problem, convolutional neural net (CNN) architectures, such as Inception[9] and ResNet[3], have been built for large-scale image classification. Our landmark recognition system is based on VGG16[8] with architecture adapted to our task.
Geometric verification is performed on the output of the convolutional neural net model to ensure that an image is indeed a landmark. Scale-Invariant Feature Transform (SIFT) [4] and Speeded-up Robust Feature (SURF) [1] algorithms are used for feature extraction. We use Fast Approximate Nearest Neighbor(FLANN)[5] for keypoint matching, which is an algorithm to accelerate the matching of features, instead of exhaustively matching between all descriptors of two images, it constructs a randomized kd-tree and matches a query point to an approximate nearest neighbor based on decision boundary.
The correspondence between the query image and the images in candidate classes is then verified using affine transformation with Random Sample Consensus (RANSAC) [2], which is an iterative method to make robust estimation of the transformation matrix between two images given the set of matched keypoints, so that an optimal amount of inliers is kept through this transformation.

2 Method

2.1 Overview

The first part of our pipeline is to use a ConvNet model to classify input images. Instead of following our original plan in the proposal that we first categorized images into general types of architectures and then

[1] `https://github.com/lihd1003/CSC420Project`

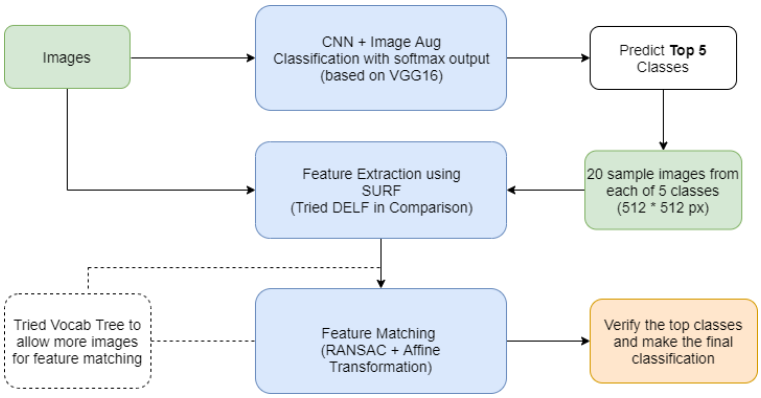

Figure 1: overall pipeline of our method [group work]

determined their specific classes, we classified the images directly into the landmark classes. The justification for this change is that we conduct transfer learning with the pre-trained model trained on ImageNet, which has its parameters tuned for classifying images into the classes of ImageNet and the general categories we want to classify our images into are included in those ImageNet classes.
The second part is to further verify whether an image contains a landmark or not, based on the prediction of the ConvNet and feature matching using affine transformation with RANSAC.

2.2 Dataset and ConvNet [Haoda Li] [2]

Our training set is obtained from Google Landmark Dataset v2. The top 1000 classes with the most images are selected and 200 images are sampled from each class. Then the images are split into 80% training set and 20% validation set. The test set contains 250 landmark images, which are randomly sampled from the top 1000 classes excluding our training and validation set. The rest 150 non-landmark images are randomly sampled from our own photos, online images and Indoor Scene Recognition Dataset [7]. The non-landmark set is a mixture of unrelated objects with buildings and indoor scenes, so that we can better measure the classification results on things similar to landmarks.
The architecture of our model is based on VGG16 with some modifications to accommodate for our task. First, 128×128 px is used as the size of our input and the size of the average pooling layer (in between feature extraction layer and classification layer) is reduced from 7×7 to 3×3 accordingly. Second, dropout layers are removed to keep all the features. Third, batch normalization is done after each convolution layer. Finally, the size of the fully connected layers is modified based on the output of the convolution layers and the dimension of the output.
Transfer learning is used to speed up the training process. We used the parameters for feature extraction layers from the model pre-trained on ImageNet (from `torchvision` package). And we trained our model by freezing all convolutional layers except the last three.

2.3 Image Augmentation [3]

Since images from each class have large variation (different perspective, illumination conditions, inside or outside, etc), we adopted image augmentation methods during the training process to increase the generalizability of our model. By inspection on the training images, I decided to augment the dataset using cropping and scaling, rotation, perspective transformations, change of brightness and blocking.
Rotation and perspective transformations are used to imitate change of viewpoints. Cropping and scaling is used to mimic close or distant view. It is also able to reduce distractors in the images taken from a distant view, since landmarks are usually close to the center of an image, by cropping the sides of the image,

[2] See `vgg16.py, datasets.py, run_model.py` for implementation
[3] See `dataset.py, image_augmentation.py and Appendix 2 for examples` for implementation

unrelated objects are removed from the scene. Blocking is used for imitating possible occlusions and also for reducing distractors. And change of brightness is used to compensate for different illumination conditions.

2.4 Geometric Verification [4]

In order to further verify the ConvNet output, we add a geometric verification step to our pipeline. We sample 20 candidate images (512×512 px) for each of the top 5 classes predicted and did feature extraction on the query images and the candidate images with SURF and feature matching with the FLANN algorithm. Then we apply affine transformation on the query image to compare its similarity with the candidate images. RANSAC is performed to find the best fit for the transformation. The max iteration for RANSAC is set to be 300 and the error threshold for inliers is set to be 4.
We experimented with two feature detection methods: SIFT and SURF. SIFT algorithm applies laplacian of Gaussian filter to an image at different scales and selects keypoints among the local maxima of the difference of Gaussian. SURF uses instead a squared filter which can be computed faster with an integral image. SIFT constructs a 128 dimensional feature descriptor from a gradient-based orientation histogram of a small patch centered around the keypoint while SURF constructs a 64 dimensional descriptor based on the sum of Haar-wavelet response.
These two methods were selected since they are robust against changes in scale, illumination and noise. The SIFT/SURF feature descriptors are extracted from images of the candidate classes. A ratio of the distance to the top two matches (0.7) is set for selecting valid matches. The matches of keypoints found by using SIFT and SURF were compared, the positions of keypoints are similar for the selected matches after applying threshold. For efficiency during testing, SURF is selected as the feature extraction methods since comparing lower dimensional vectors require less computation.
I then tried FLANN matching algorithm to boost the speed. This algorithm is compared with the brute-force matching algorithm to ensure that we do not lose much accuracy.
The next step is to verify matches between images by fitting transformations with RANSAC. Compared with finding perspective transformations (4 points method and homography), RANSAC has the most robust estimation for affine transformations, given the large variation and low resolution of our images. Therefore, we chose to perform affine transformation with RANSAC.

2.5 Combination [group work]

Two methods were proposed to combine our work to get better result. The first is to re-rank the top 5 classes by the number of candidate images matched. The second is to discriminate between landmark and non-landmark [Haoda].
We experimented with these two methods (See results in Section 3.2) and finalized this step by (i) setting 14 as the threshold of number of matched keypoints for classifying images as in the same class, then (ii) setting 6 as the threshold of number of matched images for verifying that an image belongs to a class. If a query image is not classified to be in any of the top 5 classes with the two threshold, it is classified as non-landmark. If it is verified as landmark, we output the top class predicted by the neural network.
The thresholds were selected by sampling 50 training images from 5 classes. The average number of matched keypoints among images from the same class with that among images from different classes was calculated to set threshold (i). Based on this threshold, we then calculated for each sampled image the number of matched images in the same class to set threshold (ii).

2.6 Bag of Visual Words [5]

A limitation of our pipeline is that we only use a randomly sampled subset of images for geometric verification in order to control the query time and accommodate for the space constraint. The samples may not fully represent the whole class. By constructing bags of visual words for the classes, we will be able to save space by only storing a set of feature descriptors and an indexing file for each class, and thus allow more images to be compared with the query image.

[4] `See ransac_matching.py and ransac_metrics.py for implementation, SIFT, SURF and FLANN used openCV`
[5] `See vocab_train.py, vocab_predict.py for implementation and Appendix 3 for the result of an example`

During query time, we can find the most similar images in each class for the query image by comparing their weighted global feature descriptors. This will also speed up the process for doing RANSAC, since the images are more relevant, we will be able to find a set of inliers to do transformation with fewer iterations.
Unfortunately, a more complicated structure(e.g. hierarchical vocabulary tree) is needed for storing and fast query all features to fit the scale of our dataset. Due to time constraints, I was not able to finish this step and include it in our pipeline.

3 Results and Discussion

3.1 Results of Image Augmentation

The comparison of the effect of the image augmentation methods on training is shown in Table 1 below. The model are trained on class 0 to class 99, with 100 image samples each class, and ran 12 iterations. The validation set is the 100 classes in our original validation set.
The dataset after augmenting with random change of brightness increases the accuracy on our validation set and with cropping and resizing the validation accuracy is close to that with original data. However, random perspective transformations and blocking does not improve the generalizability of the ConvNet as expected. One possible explanation for this outcome is that we used pre-trained model trained on ImageNet and is already fine-tuned for linear transformation of features. Also, the random mask generated by random blocking may not be a good representation of the occurrence of real-world occlusion or noisy objects.

3.2 Results of Geometric Verification

The accuracy(correct prediction / number of test set images) for different methods are shown in Table 2. We used threshold on the ConvNet prediction as our baseline. If the highest confidence is smaller than 0.45, we regard the query image as non-landmark. This gives an accuracy of 56.75% [Haoda].
Then we tested on the two combination methods. The first is that the accuracy would be improved after reranking the top 5 classes. However, reranking results in an even lower accuracy (53.75%). The second is to focus on where the ConvNet did not perform well, more specifically, to discriminate between landmark and non-landmark. This improves the accuracy of our pipeline to 63%.
The results showed the advantage of using geometric verification to compare the similarity of two objects in a global scale. While they also showed the limitation of our method in the cases that require the discrimination between landmarks sharing more similar characteristics. By inspection, we found that the failure of geometric verification is due to the limitation of the feature detection algorithms. Although SIFT and SURF have the advantage of detecting invariant features across different variation(scaling, illumination) of an object, they are not able to judge which objects the features are selected from. A lot of background keypoints (e.g. trees, flowers) are matched across images from different classes.

3.3 Discussion on Results

(See examples in Appendix 1) In general, our pipeline have high accuracy in predicting landmark with more distinct features (e.g. pinnacles, decorations, statues), taken in full view (which have more keypoints for matching) and having few objects other than the landmark.
Besides failing for landmark lacking distinct features, such as natural scenes, our pipeline also makes errors in two interesting scenarios. It has false negative in street views (e.g. classify Sheung Wan in Hong Kong as non-landmark). This is because such kind of images are also noise in other classes, so the ConvNet has low confidence in prediction. Meanwhile, images in such classes are often taken in different area and share few common keypoints so it fails in the geometric verification step.
In the contrary, it has false positive in scenes that share common features with landmark (e.g. classify a street car as landmark since it has "dome" and "column" or a bookshelf as landmark since it looks like a brick wall). One reason for the failure in such cases is that the input images to our pipeline are of low resolutions so that subtle details (e.g. characters on books and over the store) are lost.

Augment	Train Accuracy	validation Accuracy
No Argumentation	79.39 %	51.25 %
Random Perspective	61.57 %	50.46 %
Random Brightness	**76.40 %**	**53.07 %**
Random Crop and Resize	**68.26 %**	**51.20 %**
Random Blocking	51.19 %	49.23 %

Table 1: Image Augmentation

Feature matching	Classification rate
Threshold on neural network regard any prediction < 0.45 as non-landmark	56.75%
RANSAC affine transformation and re-ranking by number of matched images	53.75%
RANSAC affine transformation and threshold by number of matched images	63%

Table 2: Classification Rates [group work]

4 Challenges

We were faced with several challenges during the development of our pipeline.
The biggest challenge comes from the scale of our dataset. Due to space constraints, we were only able to store images of lower resolution, which resulted in some unexpected outcome during feature matching stage. The training images for the ConvNet is only 128×128 px. In order to extract sufficient amount of keypoints to be used for matching, we re-downloaded images with higher resolution and had to make a trade-off between the size and amount of images.
A problem also arose when we did transformation on images using matched keypoints. Since images for the landmark are taken from various viewpoints which involves 3D transformation and projection of the points in the scene, our initial thought was to use RANSAC with homography. However, the RANSAC algorithms were not able to return a stable transformation. I first thought that it is the issue with inlier ratio, so I tried setting larger max iteration based on the empirical and expected inlier ratio, adjusting the error tolerance for counting a point as inlier and also reducing the threshold for inlier ratio. However, the results were not satisfying.
It took me a while to realize that after resizing (convolution with filters), the depth information in the images is more or less lost and since many photos of landmark are taken from a distant view, a small difference in pixel position would result in large difference in depth. Thus, a homography transformation may not be a good representation of the changes in viewpoint. Then I tried with affine transformation, which gave a more stable transformation and the keypoints after transformation are mostly on landmark.
It was also surprising that the re-ranking method did not improve the accuracy of our pipeline and even gave worse result than setting a hard cutoff to the ConvNet output. Fortunately, we came up with the idea of focusing on one weak point of the ConvNet and use our geometric verification step to discriminate between landmark and non-landmark.

5 Conclusions

We successfully built an image recognition pipeline for landmark detection by combining ConvNet with classical image understanding methods. By applying image processing methods such as linear transformations, noise addition and color shifting to augment our dataset, I found the proper methods to increase the generalizability of our ConvNet. By using SIFT and SURF feature detection algorithms and affine transformation

with RANSAC to discriminate landmark and non-landmark, the accuracy of our pipeline is improved. In the future, I would further expand the bag of visual words module to construct a hierarchical vocabulary tree for our large-scale dataset. This would allow faster feature retrieval for matching and avoid loss of information by only sampling images from candidate classes during geometric verification. We can also try to integrate the vocabulary tree with the ConvNet, so that we can use feature descriptors with more subtle details and increase the accuracy of the ouputs.

References

[1] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded up robust features". In: *European conference on computer vision*. Springer. 2006, pp. 404–417.

[2] Martin A. Fischler and Robert C. Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Commun. ACM 24, 6 (June 1981), 381-395* (1981). arXiv: 358669.358692.

[3] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[4] David G Lowe. "Distinctive image features from scale-invariant keypoints". In: *International journal of computer vision* 60.2 (2004), pp. 91–110.

[5] Marius Muja and David Lowe. "Flann-fast library for approximate nearest neighbors user manual". In: *Computer Science Department, University of British Columbia, Vancouver, BC, Canada* (2009).

[6] Kohei Ozaki and Shuhei Yokoo. "Large-scale Landmark Retrieval/Recognition under a Noisy and Diverse Dataset". In: *arXiv preprint arXiv:1906.04087* (2019).

[7] Ariadna Quattoni and Antonio Torralba. "Recognizing indoor scenes". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2009, pp. 413–420.

[8] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[9] C. Szegedy et al. "Going deeper with convolutions". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015, pp. 1–9. DOI: 10.1109/CVPR.2015.7298594.

6 Appendix I: Results

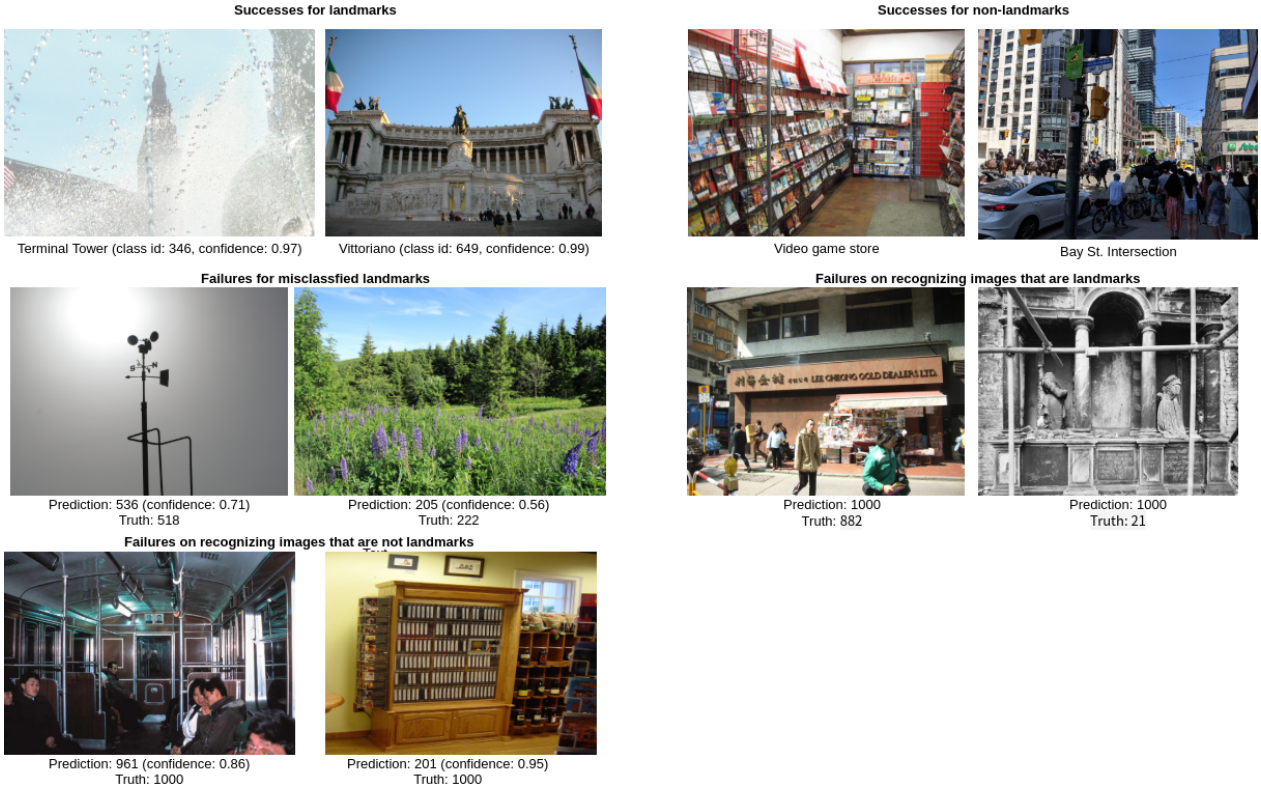

Figure 2: Results

7 Appendix II: Image Argumentation Methods

Figure 3: Block

Figure 4: Brightness

Figure 5: Crop

Figure 6: Perspective

8 Appendix III: Query in Bag of Visual Words

Figure 7: Query Image

Figure 8: Candidate 1

Figure 9: Candidate 2