Final Project: Fine-grained localisation

Yongzhe Zhang
Faculty of Engineering and Information Technology
The University of Melbourne
Melbourne, Australia
yongzhe@student.unimelb.edu.au

Yingpeng Tan
Faculty of Engineering and Information Technology
The University of Melbourne
Melbourne, Australia
yingpengt@student.unimelb.edu.au

***Abstract* — Given the images database consisting of both labeled and unlabeled data, where the label means the geolocation of shooting the images. This report uses ResNet, VGG, and phash to primarily filter out the approximate picture. Then our approach uses SIFT for feature detection and description. Finally, using RANSAC to compare the key points among the images and returning the most similar images' geographical coordinates for the unlabeled image. We make comparison of three methods and suggests some future improvement.**

Keywords — computer vision, ResNet, VGG, SIFT, geolocation

I. INTRODUCTION

With the rapid development of the internet and smartphones, unlabeled images are readily available from many different resources through social media networks. In the face of these massive photographs, we must realize that there is a lot of information, including the parameters of the camera itself, as well as the information it captures about the outside world. As for the information on the image, in addition to detecting objects and sorting and identifying them, we are also very interested in the exact location of the photo taken. In recent years, this positioning problem has been classified as geolocation, which have evolved so fast. This is mainly due to almost every mobile device that can take pictures with a certain degree of positioning and Internet access, which provides many images with location information for research. But at the same time, for a variety of reasons that may occur, there will be many photos that do not come with the corresponding geographic coordinate information. However, people may have a need to get the geographic coordinate information of these images. For the solution of this problem, today's rapid development of computer vision for it to provide an effective solution. One possible approach is to 3D reconstruct the captured object based on existing photograph information. Then, when we get a picture that needs to identify its coordinates, we just need to determine where the image taken is relative to the 3D model [1]. In addition, another strategy to address this problem is to match the similarity of an unlabeled image to an existing labeled dataset, and, after deriving the most similar image, use the detection and comparison of point descriptors to calculate the geographic coordinate information in which the image is located. Compared with 3D model reconstruction, this method is lighter and simpler in terms of algorithm difficulty, total calculation, and memory consumption.

The main concern of this report is to predict the images geographical coordinates based on the given data. Specifically, there are 7500+1200 images token in an exhibition hall. However, 7500 images are given the geographical coordinates which are the train data, while the remaining 1200 images are not, which are the test data.

This report will introduce all the adopted methods in II.RELATED WORK. The methods include phash, VGG, ResNet, SIFT, and RANSAC. Then, it will introduce our approach in experiment part, and present the evaluation result of the experiment. Finally, this report will make conclusion and suggest future improvement.

II. RELATED WORK

One of the research we have learned so far on the subject of identifying geolocation information about the shooting location is based on video media, and although this is different from the image recognition that this report focuses on, the principles can be used for our reference: Salvador Medina et al. tackle video-geolocation through traditional image retrieval techniques based on Google Street View as the reference [2]. First, they use NetVLAD to obtain the deep learning features to represent images' similarity for each frame. Then, they use a bundle of voting based methods to aggregate the geolocation results. Finally, they found that the combination of NetVLAD and SIFT similarity can compute the best aggregation.

Also, there are many researches that concentrate on image place recognition. Chen et al. use street-level image data to show identify city-scale landmark based on mobile devices, and improve feature detection and incorporate a user's position [3]. Among all those methods, it is a common strategy that using the locations of the most visually similar images obtained from the geolocation-labeled database to estimate the location of the given particular unlabeled images.

For similarity comparison, our approach use perceptual hashing, VGG, and ResNet respectively:

A. Perceptual hashing (phash)

Given various forms of multimedia, for example, the images, perceptual hashing is an algorithm that can be used to produces the fingerprint of these images [4]. Perceptual hash is locality-sensitive, which means that if features of the given multimedia are similar, then the hash result will be analogous [4]. In practical application, this method is widely used in copyright infringement. In addition, researchers have found that perceptual hash can also be used to compare and match images in databases [5]. This method has many advantages, such as its simple and easy to implement, and the calculation speed is relatively fast, so it can be used as a good baseline. But there is also a problem with the phash algorithm, which gets a very low similarity score for partially cropped images, for example, in copyright detection, if the user continuously exposes the creative image is a partial crop operation, then the phash algorithm is not recognized [4].

B. Very Deep Convolutional Networks (VGG)

Very Deep Convolutional Networks is the name of a pre-trained convolutional neural network invented by Karen

Simonyan and Andrew Zisserman [6]. The main work is to prove that increasing the depth of the network can affect the final performance of the network to some extent. In addition, VGG has many other advantages, including: 1. The structure of the VGG is very simple, and the entire network uses convolution core sizes of the same size (3x3) and maximum pool size (2x2). The combination of several small filter (3x3) convolution layers is better than a large filter (5x5 or 7x7) convolution layer[7]. However, VGG consumes more computing resources and uses more parameters, resulting in more memory consumption, so training a VGG model usually takes longer, but fortunately there is an open pretrained model that makes it easy to use [7].

C. *Residual Neural Networks (ResNet)*

Residual Neural Networks are artificial neural networks (ANNs) whose structure is known to pyramid cells in the cerebral cortex [8]. Although the deeper the network is when building convolutional networks, the richer the level of extractable features, we tend to use deeper network structures to achieve higher levels of characteristics. But when using deep network structure, we encounter two problems, gradient disappearance, gradient explosion problem and network degradation problem. By utilizing skip connections, or shortcuts to jump over some layers, ResNet solves the degradation of deep networks through residual learning, allowing us to train deeper networks.

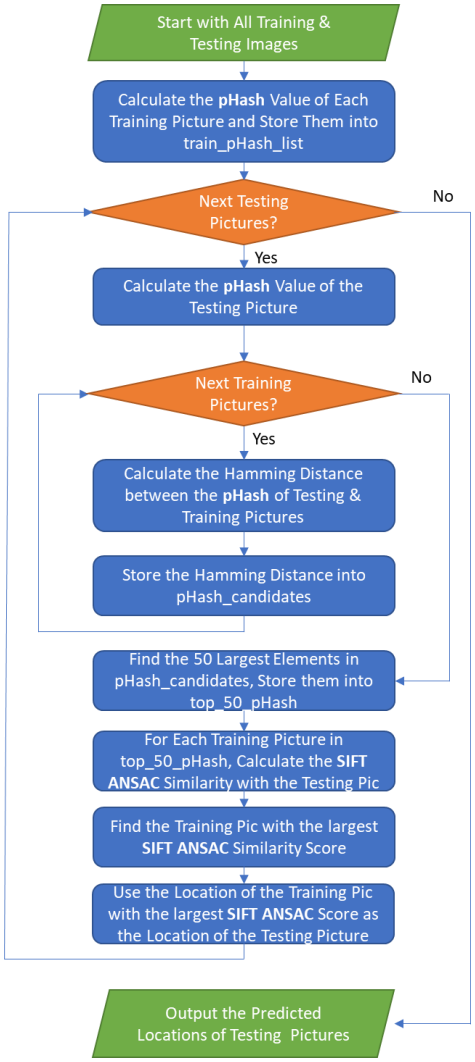

Fig. 3.2 Procedures for Models using pHash and SIFT

After retrieving the characteristic value of an image by using the three methods described above, our approach will use SIFT to further filter its approximate collection of candidates thus to select its closest image and make prediction of its geographic coordinates. Here, we will introduce the feature detection algorithm we adopted, SIFT:

A. *Scale Invariant Feature Transform (SIFT)*

Scale Invariant Feature Transform (SIFT) is a computer vision algorithm to detect and describe local features in images, which is invented by David Lowe [9]. SIFT algorithm is a kind of algorithm based on local points of interest. It's insensitive to image size and rotation, and it also has excellent resistance to the effects of lighting, noise, etc. Compared with the previous algorithms, SIFT has these qualitative improvements in the performance. However, SIFT has its disadvantage: it is mathematically complicated and computationally heavy. Therefore, some researchers came up with some improved methods, for example, SURF.

After obtaining the feature descriptors, our approach will use RANSAC to simultaneously solve the correspondence of two images. RANSAC was proposed by Fischler and Bolles [10], and it is widely used in computer vision for image's feature matching.

III. Methodologies

In order to predict the location of a testing image given the locations of training image, a straightforward way is to find the nearest training image and use the location of training image as the location of testing image. SIFT RANSAC is found to be the most suitable method to find the nearest images. However, its advantages come with high computational cost, which will be discussed in Part III.A. As a result, several combinations of model were implemented to balance the performance and cost, which will be discussed in Part III.B & C.

A. *Naïve Trial: Using SIFT RANSAC only*

SIFT RANSAC is the first option for finding the nearby images for its high accuracy. However, as shown in Fig 3.1, in the testing, SIFT similarity calculation takes more than 1 hour to loop through all the 7500 training images and find the similarity for a single test picture. As a result, all 1500 testing pictures will take more than 1500 hours in total, which is not feasible considering the time complexity.

```
traindir = './COMP90086_2021_Project_train/train/'
train_dir = os.listdir(traindir)
testdir = './COMP90086_2021_Project_test/test/'

for img in tqdm(train_dir):
    if re.match(r'(.*?jpg)', img) is not None:
        print('./COMP90086_2021_Project_test/test/IMG42

data.close()
```

```
100%|██████████| 7500/7500 [1:17:35<00:00,  1.61it/s]
```

Fig. 3.1 Executing Time of Applying SIFT RANSAC to 1 Testing Image and All 7500 Training Images

B. *pHash with SIFT RANSAC*

As shown in Fig 3.2, in the beginning, pHash was applied to all the training images to calculate their characteristics in

frequency aspect. And then for each testing image, we only apply SIFT RANSAC to the top 50 similar training image from pHush, which reduce the candidates from 7500 to only 50. As a result, the whole calculation finishes in 6 hours.

C. *ResNet/VGG with SIFT RANSAC*

As a pretrained model, ReNet with imagenet / VGG19 can export the feature matrix of the input pictures from its convolutional layer with speed and accuracy. In this model, we extracted the output from 'block5_conv4" as the feature matrix, and then use NeareastNeighbour from KNN to find the top 50 candidate to feed to the SIFT RANSAC

Similar to Part III.B, as shown in Fig 3.3, in the beginning, the pretrained model ResNet/VGG was applied to all the training images to calculate their feature matrixes. And then for each testing image, we only apply SIFT RANSAC to the top 50 similar training image from ResNet/VGG, which reduce the candidates from 7500 to only 50. As a result, the whole calculation also finishes in 6 hours.

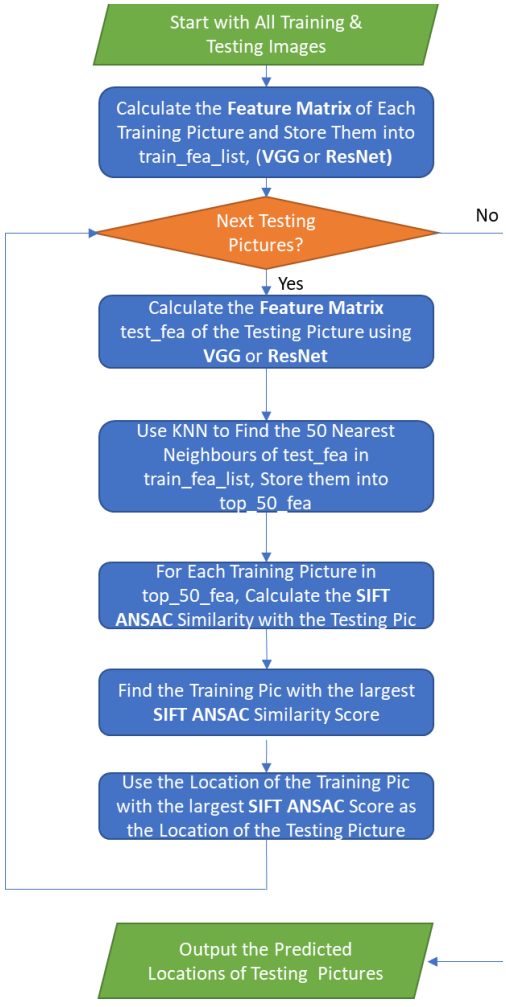

Fig. 3.3 Procedures for Models using VGG/ ResNet and SIFT

IV. Result and Conclusion

Model	Manhattan Distance (Kaggle Score)	Executing Time /hours
Pure SIFT	--	1500 (Estimated)
pHash + SIFT	44	6.2
VGG19 + SIFT	9.32	6.3
ResNet + SIFT	7.3	6.2

The result shows that all of the SIFT combined methods will roughly have the same computational time. This is because the approximate similarity computation costs relatively less time compared with feature matching by RANSAC. As for the final performance, ResNet + SIFT outperform the other two, and pHash + SIFT has the worst performance. The reason may be that pHash is only a rough feature calculation of the uniqueness of the image, which means it doesn't understand the inner meaning of the image as much as the other two CNN models.

V. Future Improvement

There are two major improvement could be done to make the prediction of our approach more accurate:

A. *Epipolar Geometry*

As far as the methodology used in this report is concerned, after sorting out the image most similar to the particular unlabeled image, we simply return the coordinate information of the image retrieved from the train database directly without calculating the new coordinate points. This simple approximation is bound to lead to irrefutable errors. Because, obviously, there aren't two images with the same content that happen to be taken at the same geographic coordinates. Therefore, in the future work, we need to calculate the feature point coordinates on the existing images, by applying the Epipolar Geometry methods [11]. The major steps could be concluded as follows:

1) Detect the keypoints for the images, which are the same as our original approach.

2) .Use RANSAC to find the Fundamental matrix for the two image, along with the inliers which will further be used to determine the camera's transform.

3) Compute the Essential matrix after the camera's movement, including translation and rotation between cameras.

4) After obtaining the Esseantial matrix, decompose it to determine the camera's transform, including translation and rotation.

B. *RANSAC Improvements*

In the actual operation process, our approach spends a lot of time in using RANSAC for image matching, while the matching results were not satisfactory. This may be because the SIFT description sub match produces too many mismatches, exceeding the mismatch tolerance limit of the RANSAC algorithm. In addition, because the maximum number of iterations is artificially specified in practice, RANSAC has a certain probability of retaining mismatches

after a limited number of iterations, however, the greater the number of iterations in theory, RANSAC will have a greater probability of getting the correct results. In the course of the experiment, based on our observations, it was found that if there was a large distortion in the two images used for comparison, it would violate the basic assumptions of RANSAC, resulting in a large error in the algorithm

REFERENCES

[1] Ham, H., Wesley, J., & Hendra, H. (2019). Computer vision based 3D reconstruction: A review. International Journal of Electrical and Computer Engineering, 9(4), 2394.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] Medina, S., Dai, Z., & Gao, Y. (2018). Where is this? Video geolocation based on neural network features. arXiv preprint arXiv:1810.09068.

[3] Chen, D. M., Baatz, G., Köser, K., Tsai, S. S., Vedantham, R., Pylvänäinen, T., ... & Grzeszczuk, R. (2011, June). City-scale landmark identification on mobile devices. In CVPR 2011 (pp. 737-744). IEEE.

[4] Niu, X. M., & Jiao, Y. H. (2008). An overview of perceptual hashing. Acta Electronica Sinica, 36(7), 1405-1411.

[5] Zakharov, V., Kirikova, A., Munerman, V., & Samoilova, T. (2019, January). Architecture of Software-Hardware Complex for Searching Images in Database. In 2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus) (pp. 1735-1739). IEEE.

[6] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[7] Khan, A., Sohail, A., Zahoora, U., & Qureshi, A. S. (2020). A survey of the recent architectures of deep convolutional neural networks. Artificial Intelligence Review, 53(8), 5455-5516.

[8] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[9] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. International journal of computer vision, 60(2), 91-110.

[10] Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM, 24(6), 381-395

[11] Zhang, Z. (1998). Determining the epipolar geometry and its uncertainty: A review. International journal of computer vision, 27(2), 161-195