EL net: Ensemble Learning in end-to-end learning

Yizhen Peng

Digital Technology, Dalian Institute of Science and Technology, Dalian, Liaoning, LN 411, China

e-mail: pyzyzeason@163.com.

Abstract. Kednell. et al. [1] propose a robust and real-time monocular six degree of freedom (6-DOF) relocalization system which uses the end-to-end method to regress the 6-DOF pose in the RGB image. However, when the scenes change, PoseNet has huge variance. Ensemble learning can deal with this problem. We propose to use a new end to end camera relocalization method which combines the ensemble learning with PoseNet. In this model we modify the structure of PoseNet and get several different weak models. After that we train these different models and then use ensemble learning to do them together. Experiments show the effectiveness of ensemble learning, compared to previous PoseNet, using ensemble learning can get better result.

1. Introduction

In recent years, with the advancement of technology, the application of robotics becomes popular. Compared to the rising cost of labor workers, robots are preferred to be used in the industry. Many logistics companies now use robots to carry goods, in complex warehouses. The traditional 3-dimensional pose estimation method has many limitations; GPS cannot be used for indoor positioning and it has low positioning accuracy. How to achieve accurate positioning is one of the challenges faced by robots.

Visual-SLAM (Simultaneous Localization and Map- ping) is a visual-based robot positioning technology. In 2004 David Nister published a paper [4] about a SLAM scheme designed from 3D Vision. Although the design of the whole scheme is not complicated, its algorithms and ideas are used by many later researchers. A well-known example in 2003, Mars robots used this technology to detect the surface of Mars [3].

In 2015, A new ideal [1] is provided, using the GoogLeNet [2] to learn the High-dimensional features and later these features will be used for position regression. The main contribution of this paper is to propose a robust and real-time 6-DOF repositioning method. Although its main test site is outdoors, it can also be used in an indoor environment. After that based on GoogLeNet Ken et al. [9] proposed a method that uses Bayesian convolutional neural network to get the 6-DOF camera pose and uses the uncertainty metric to estimate the positioning error and detect the presence or absence of the input image scene.

In Posenet2: Geometric Loss Functions for Camera Pose Regression with Deep Learning [6], they explore a number of novel loss functions for learning camera pose which are based on geometry and scene reprojection error. By leveraging geometry, this paper significantly improves PoseNet’s performance across datasets ranging from indoor rooms to a small city.

Compared with traditional positioning methods, Xu et al. [7] proposed a multi-sensor indoor global positioning system based on the combination of visual positioning and probabilistic positioning. They
use a closed-loop positioning mechanism to take advantage of the correlation of continuous images to improve the accuracy and speed of positioning.

Since Ensemble Learning is not a new concept, it actually has a history of more than ten years, and it has a wide range of applications in machine learning. Ensemble learning is a way to get a better model by combining multiple relatively weak classifiers or predictors. Such as in 2018 Researchers used Ensemble learning with deep learning to classify images [5]; In the medical field Sujit et al. [10] propose to use the overall deep learning model based on deep convolutional neural network to automatically evaluate the quality of multi-center structure brain MRI images.

As in this paper, we focus on end-to-end learning method for camera localization, the stacking is used. Specifically, we have train three PoseNets with different structure and then we use ensemble learning to put them together which can get better results than previous methods.

The rest of the paper is organized as follows: Sections II is the detail about GoogLeNet, Ensemble Learning. Sections III introduces Experimental information. Section IV is the concludes.

2. Methodology

2.1. Overview

It is worth mentioning that the result of PoseNet has huge variance, since each scene is different. This will cause model to have different results. In this section we introduce structure of ensemble learning and how to improve the result of PoseNet by using ensemble leaning. By modifying the GoogLeNet, we will get three different PoseNets, which will be combined together to improve the performance of original PoseNets.

2.2. Structure of GoogLeNet convnet

GoogLeNet proposed a new deep Convolutional Neural Networks (CNN) architecture—Inception, without a fully connected layer, which can save computing time and reduce number of parameters. The GoogLeNet team proposed the Inception network structure, which uses a “basic neuron” structure to build a sparse and high computing performance network structure. Inception network is to add an additional 1 × 1 convolutional layer and to use ReLu as the activation function. Its main function is dimension reduction, without sacrificing the performance of the model.

Especially, the basic convolution block in GoogLeNet is called the Inception block (see Figure 1), we can see that there are four parallel lines in the Inception. The first three lines use convolutional layers with sizes of 1 × 1, 3 × 3, and 5 × 5 to extract information under different spatial sizes. The middle 2 lines will first perform 1 × 1 convolution on the input to reduce the input Number of channels to reduce model complexity. The fourth line uses a 3 × 3 maximum pooling layers, followed by a 1 × 1 convolutional layer to change the number of channels. All four lines use appropriate padding to make the input and output consistent in height and width. Finally, the output of each line is connected on the channel dimension and input into the next network layer.

![Figure 1. Structure of Inception.](image-url)
2.3. Stacking Method
We use stacking method to combine the structure of different PoseNets. As show in Equation (1). We use the same image dataset to train three different structures, so that we will get three different models, and these three models can be combined together to generate a new model.

\[
\text{pose} = E(\text{PoseNet1}(\text{Image}), \text{PoseNet2}(\text{Image}), \text{PoseNet3}(\text{Image})).
\]  

The Ensemble learning is mainly divided into bagging, boosting and stacking methods. In this article we use stacking method to get a better model. In general, it can be divided into the following steps:

1. Divide training data into two groups.
2. Select m weak learners and use them to fit the first set of data.
3. Make each of the m learners predict the second set of data.
4. Use the predictions made by the weak learner as input to fit the meta-model on the second set of data.

PoseNet's model is a slightly adjusted version of GoogLeNet, the SoftMax layer set by GoogLeNet for classification is removed, and a fully connected layer is added at the end to reduce the dimension of high-dimensional features, the input of the model is a 3-channel color image with a resolution of 244 × 244. The middle network layer removes the two output branch structures compared to GoogLeNet. Figure 2 show the basic structure of the PoseNet.

![Figure 2. PoseNet Structure.](image)

The image dataset passes through three network structures in sequence then PoseNet get the final result. Our method is modified based on the basis of PoseNet. We retain most of the PoseNet structure, just modify it in the three output sections. For model 1 we retain only one output part, and Model 2 retains two output layers. Model 3 retains all output layers. Then the stacking method is used to combine three different models in order to get better result. As shown in Figure 3:

![Figure 3. Ensemble Learning Structure.](image)
Reducing the output part is actually reducing the number of network layers. The first model only retains the first output part, the second model include the first and second network layers. In model three it has all the network layers.

2.4. Structure of Model 4
Model 4 base on the previous three models, it has similar network structure. We use same data loader to get the information of image dataset. In the training part, we put the same training picture data into three different models, but we take the last output of each model to generate the predicted value of model 4. In the test part, in order to observe the error of the prediction result more clearly, we use the fully connected layer to convert the original prediction result into a two-dimensional tensor that meets the calculation conditions. In the fully connected layer, we use the linear transformation formula to convert the latitude.

\[ y = xA^T + b \]  

In this formula, A is weight the learnable weights of the module of shape; b is bias the learnable bias of the module of shape.

2.5. Loss Function
The loss function of the overall network structure is shown in Equation (3):

\[ \text{loss}(I) = \| \hat{x} - x \|_2 + \beta \| \hat{q} - \frac{q_i}{\|q\|} \|_2 \]  

Among them, \((\hat{x}, x)\) respectively represent predicted value and real value at the three-dimensional coordinate position, \((\hat{q}, q)\) represent the predicted value and actual value of the camera pose angle. \(\beta\) is the hyperparameter to balance the translation error and the rotation error. We have several different models; each model has a different loss function. In the Model 1 the loss is

\[ \text{loss} (\text{model 1}) = \Sigma_{i=1}^{3} \| \hat{x}_i - x_i \|_2 + \beta_i \| \hat{q}_i - \frac{q_i}{\|q_i\|} \|_2 \]  

Since Model 2 retains the two output parts of PoseNet. In this model the loss is

\[ \text{loss} (\text{model 2}) = \Sigma_{i=1}^{2} \| \hat{x}_i - x_i \|_2 + \beta_i \| \hat{q}_i - \frac{q_i}{\|q_i\|} \|_2 \]  

As for Model 3 it has one output part the loss is

\[ \text{loss} (\text{model 3}) = \| \hat{x}_3 - x_3 \|_2 + \beta_3 \| \hat{q}_3 - \frac{q_3}{\|q_3\|} \|_2 \]  

Model 4 combines the previous three models and without \(\beta\) the loss is

\[ \text{loss} (\text{model 4}) = \| \hat{x}_4 - x_4 \|_2 + \| \hat{q}_4 - \frac{q_4}{\|q_4\|} \|_2 \]  

We can see that the basic format of the loss function of these four models is similar. The models have more output layers, the more loss function items they have.

3. Experiments
3.1. Preparation for the experiment
To run this model, we use one GPU card (RTX2080 Ti) base on Ubuntu 16.04. The Pytorch version is 1.4. In this experiment training Batch size is 35 and test Batch size is 1, crop size is 256, learning rate is 0.001 and random seed is 2020. For comparison, the training set and test set of each dataset are the same.
3.2. Datasets
We selected 3 of the indoor datasets (see Figure 4) from the public dataset 7−Scenes [8] which is a collection of tracked RGB-D camera frames and including Chess, Fire, Heads. Each of them includes several sequences, and split into distinct training and testing sequence sets. Each sequence consists of three files: RGB images, depth images and pose.txt which including $4 \times 4$ matrix in homogeneous coordinates and each image has $640 \times 480$ resolution.

![Figure 4. Three of 7-Scenes dataset: Chess, Fire, Heads.](image)

3.3. Experimental results
This paper records in real time the change of the loss value with the number of iterations during the training process, and compares it with the change of model 1’s loss function. Taking the experimental results of the Chess dataset as an example, Figure 5 shows the network structure of model 4 and the loss value of model 1. the loss value of the network structure in model 4 is significantly lower than PoseNet. Similar situations have been achieved on different datasets.

![Figure 5. Difference in loss.](image)

In order to verify the effect of integrated learning, we compare the results of the operation with PoseNet. Table 1 shows the error comparison of the two models. Although from the experimental results, our model is not better than PoseNet, our model is to prove that ensemble learning is effective, rather than simply comparing the results. Model 1 has a complete PoseNet network structure, so the model 1 result is regarded as the PoseNet result in our experiment.

| Dataset | PoseNet | Ensemble Learning |
|---------|---------|-------------------|
| Fire    | 0.51m   | 1.02m             |
| Heads   | 0.06m   | 0.75m             |
| Chess   | 0.35m   | 1.07m             |

Table 1. Error comparison
Table 2 shows the prediction results of this model for three public datasets and the prediction results of PoseNet. We can clearly see that Model 3 is better than Models 1 and 2, because the structure of Model 3 is more complicated than Models 1 and 2. And the result of model 4 is better than the other three models. Because Model 4 uses the stacking method to combine the previous three models.

| Dataset | Model 1 | Model 2 | Model 3 | Model 4 |
|---------|---------|---------|---------|---------|
| Fire    | 1.47m   | 1.50m   | 1.53m   | 1.02m   |
| Heads   | 1.38m   | 1.41m   | 1.42m   | 0.75m   |
| Chess   | 1.44m   | 1.45m   | 1.46    | 1.07m   |

In the three datasets, the heads dataset is the least, and the training set has only 1000 pictures, so stacking performs better in the heads than the other two datasets. The chess dataset has the largest number of pictures in the three datasets, and there are many dynamic blurs in the pictures. This may be the reason for the poor performance.

In the three datasets, the heads dataset is the least, and the training set has only 1000 pictures, so stacking performs better in the heads than the other two datasets. The chess dataset has the largest number of pictures in the three datasets, and there are many dynamic blurs in the pictures. This may be the reason for the poor performance.

The Chess dataset has many pictures covering almost the entire space. The picture contains rich checkerboard texture features, and continuous images rotate slowly. Choose 3000 pictures as training set, and 3 000 pictures as test set. Experiments show that there is a certain improvement over PoseNet.

The Fire dataset is similar to the chess dataset, but there are fewer pictures than chess. Continuous pictures move slowly, and most of the pictures are clear without motion blur. On this data set, 2000 pictures are selected as the training set, and another 2000 pictures are used as the test set. Experiments show that there is a certain improvement over PoseNets.

4. Conclusion
In this work, we propose an end-to-end model which uses the method of ensemble learning stack to combine three different structural models and each model is modified based on the PoseNet. Compared with PoseNet, it obtains faster convergence speed, and higher accuracy. Future work will focus on the pose estimation of high-speed mobile cameras in order to improve the accuracy of the estimated position at high speed.

References
[1] A. Kendall, M. Grimes and R. Cipolla, "PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, 2015, pp. 2938-2946, doi: 10.1109/ICCV.2015.336.
[2] C. Szegedy et al., "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 1-9, doi: 10.1109/CVPR.2015.7298594.
[3] Yang Cheng, Mark Maimone and Larry Matthies, "Visual odometry on the Mars Exploration Rovers," 2005 IEEE International Conference on Systems, Man and Cybernetics, Waikoloa, HI, 2005, pp. 903-910 Vol. 1, doi: 10.1109/ICSMC.2005.1571261.
[4] Nister, D., Naroditsky, O., Bergen, J., (June-July 2004) "Visual odometry," Computer Vision and Pattern Recognition, 2004. Proceedings of the 2004 IEEE Computer Society Conference on , vol.1, no., pp.1-652, I-659 Vol.1, 27.
[5] TaSci, Erdal & Ugur, Aybars. (2018). Image classification using ensemble algorithms with deep learning and hand-crafted features. 1-4. 10.1109/SIU.2018.8404179.
[6] A. Kendall and R. Cipolla, "Geometric Loss Functions for Camera Pose Regression with Deep Learning," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 6555-6564, doi: 10.1109/CVPR.2017.694.

[7] Xu S, Chou W, Dong H. A Robust Indoor Localization System Integrating Visual Localization Aided by CNN-Based Image Retrieval with Monte Carlo Localization. Sensors (Basel). 2019;19(2):249. Published 2019 Jan 10. doi:10.3390/s19020249.

[8] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, pages 2930–2937. IEEE, 2013.

[9] Kendall A Cipolla R. Modelling uncertainty in deep learning for camera relocalization[J]. Designing Engineering and Analyzing Reliable and Efficient Software 2015 31: 4762-4769.

[10] Sujit SI, Coronado I, Kamali A, Narayana PA, Gabr RE. Automated image quality evaluation of structural brain MRI using an ensemble of deep learning networks. J Magn Reson Imaging. 2019;50(4):1260-1267. doi:10.1002/jmri.26693.