A End-to-End Footprint Recognition Approach Based on Convolutional Neural Network

by Hanfei Sun

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
hanfeis@andrew.cmu.edu

Abstract

The author proposes an end-to-end footprint recognition method based on convolutional neural network. Though this approach doesn't need to do explicit feature extraction and complex image preprocessing, it is robust to rotations and noises. The author gives an evaluation for different data augmentation approaches and network strucutres. On the test set, the author achieves 93.85% accuracy for Left/Right Footprint Detection task and 33.64% accuracy for Footprint Matching task. The output of the hidden layer and softmax result can be used as a convenient respresentation of the data, and the author uses this information for clustering to finish the Pattern Discovery task.

1 Introduction

Footwear impressions, or footprint for short, is an important evidence at crime scenes. The goal of footprint recognition is to assign the crime scene footprint to a reference footprint in the databases. As the crime scene impression is usually noisy and may not have the same scale and orientation[1]. Traditional methods are consist of three steps: (1) **pre-processing**, such as center point determination, rotation correction, denoising and binarization (2) **feature extraction**, including but not restricted to recoginizing edges, structures or periodicity pattern of the footprints. (3) **post-processing** or **modeling**, with either supervised approaches (Bayesian models, SVM) or unsupervised ones (kNN classifiers). Usually all these three steps depend on the expertise of computer vision, machine learning, signal processing as well as domain knowledges of footwear impressions.

Convolutional neural network has been used for pattern recognition and image processing[2]. In this paper, the author tries to use an end-to-end approach for footprint recognition task with the help of convolutional neural network. With little image pre-processing and feature extraction, the model can achieve 93.85% accuracy for Left/Right binary classifier, and 33.64% accuracy for 1000-class footprint matching classifier with only one training example per class. The system could still be further optimized with the addition of better feature extraction and represetation. Moreover, the author illustrates that the intermediate output of the network layers could be used for feature extraction and combines with other machine learning approaches such as hierachical clustering for pattern discovery and data mining.

2 Data Preprocessing

2.1 Rotation Correction

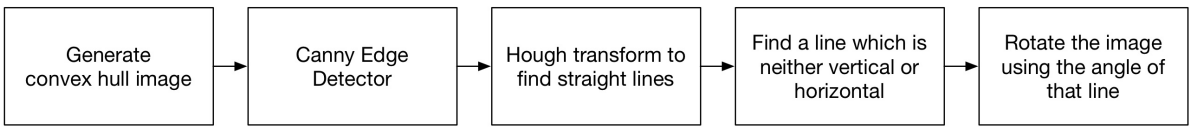

Figure 1. The workflow of rotation correction for images

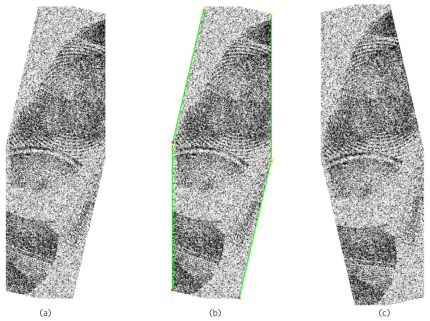

Figure 2. (a) original image (b) straight lines detected by Hough transform (c) rotated image

The author use a naive but effective way to do rotation correction for images. The workflow and effect can be seen in Figure 2 and 3. However, in the experiment, rotation correction preprocessing doesn't improve the accuracy significantly for task1 and task2. The reason may be that the author also do the data augmentation by rotating origial dataset, so rotation correction doesn't provide much help. Therefore, the rotation correction is not used for all three tasks.

2.2 Data Denoising

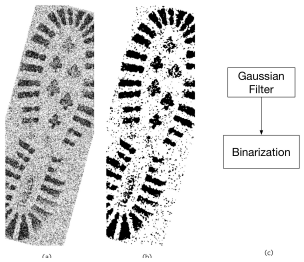

Figure 3. (a) original image (b) denoised image (c)denoising workflow

Figure 3 (b) shows the denoised result for original images. The images goes through a gaussian filter first, with 0.8 as standard deviation of gaussian kernel. Then I binarize the intermediate result using 0.6 as threshold. I find data denoising for task1 and task3 doesn't provide significant improvement, so I only use that in task2.

2.3 Data Augmentation

For task1, I augment the training set by horizontal and vertical mirroring. If an instance belongs to left foot, then its horizontal or vertical mirroring should belong to right foot. In this way, I triple the size of training set.

For task2, as a 1000-category classifier is used and there is only one instance per category. I augment the data more aggreesively. The training set is augmented by adding random rotation angle and zooming level.

3 Left/Right Footprint Detection

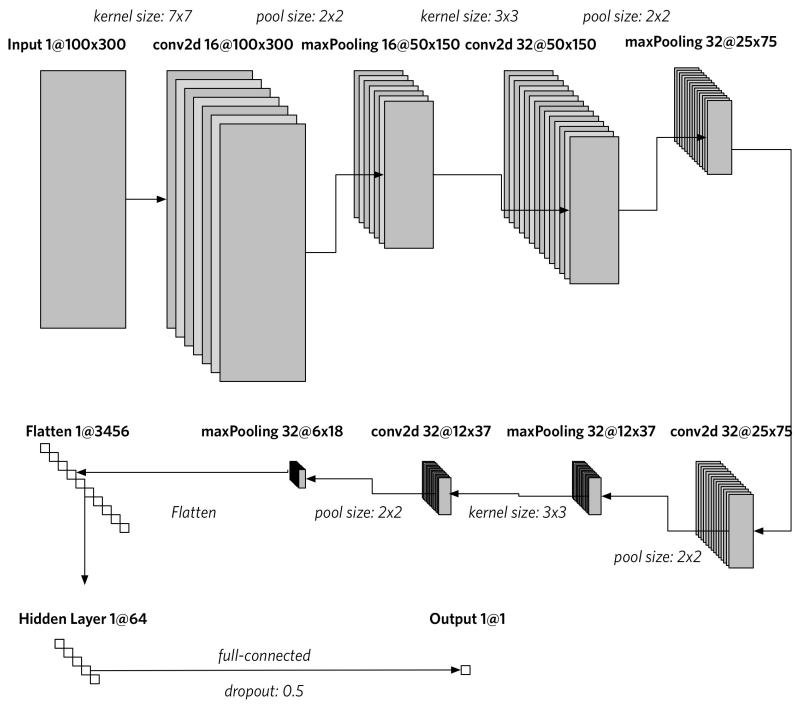

Figure 4. The Architecture of CNN for Left/Right Footprint Detection

For left/right footprint detection, the convolutional neural netwrok architecture can be seen as Figure 4. In the original dataset, the aspect ratio of images is approximately 1:3 and have grayscale color mapping. First, the original images' sizes are scaled to the resolution of 100×300 and the grayscale values ($0 \sim 255$) are scaled to $0 \sim 1$. Then the images go through a series of convolutional layer as well as max pooling layer. The stride for convolutional layers is 1 and the stride for max pooling layers is 2. After that, the feature map of the last max pooling layer is flatten and fully connected with a hidden layer. Then it goes through a dropout process with dropout ratio of 0.5 and connect to a single output layer. The workflow get the sigmoid result of the output layer to do binary classification. All the other activation functions is rectified linear unit (ReLU).

For data partition, I split the 10000 original training set into 9000 training set and 1000 validation set.

Next I'll discussion about how the author chooses these parameters and hyper-parameters and some empirical results during the parameter tuning. In the experiment, the critical parameters are: (1) depth of the neural network (2) kernal size of convolutional layers (3) batch size (4) dropout rate (5) early stopping

Deepen the neural network improves the accuracy on training set but makes it slower to converge. An architecuture which uses four pair of convolutional/maxPooling layers can achieve higher validation accuracy then the one with three pairs, but five pairs of convolutional/maxPooling layers are not significantly better than the one with four pairs. Deepen the fully-connected layer doesn't provide much improvement in the experiment either.

Increasing the kernal size increases the validation accuracy significantly. 5×5 kernal size is better than 3×3 kernal size, while 7×7 is even better than 5×5. Increasing kernal size on the first layer provides the largest improvement.

Increasing batch size makes the network converge with fewer epoches. In the experiment, I use 256 as the batch size. If I use batch size smaller than 100, the converging process will be much slower, or even not converge at all.

Dropout rate will make the netwrok spend more time before converging, but it can avoid overfitting and improve the validation accuracy finally.

I train the model for more than 600 epoches, and the model converges at around 100 epoches, which can be viewed in Figure 5. I used the 113th epoch with validation accuracy of 95.8%for the prediction.

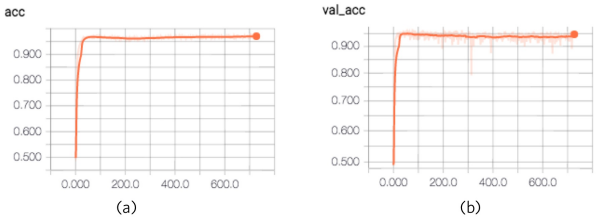

Figure 5. (a) Accuracy vs Epoch for training set (b) Accuracy vs Epoch for validation set

4 Footprint Matching

The architecture of CNN for task2 is very similar to task1. A few differences are:

1. The data augmentation is done using random rotation and random zooming in addition to horizontal/vertical flip. This is because the training samples are quite sparse and we need to generate more instances per class.
2. All the data are preprocessed by Gaussian Filter followed by binarization, which is shown in Figure 3 in Section2
3. I choose 3×3 as the kernel size of the first layers, this makes the converging process faster.
4. I use smaller batch size, which is 100, in task2. In my experience, smaller batch size can be used as a kind of regularization.
5. The size of fully-connected hidden layer is 32 instead of 64 for faster converging.
6. I remove the dropout layer for faster converging.
7. The final output layer has 1000 units (one unit per class) and softmax function is used to convert the values into probabilities.

The decisions above are mostly empirical tradeoff between training time and prediction accuracy. The first priority here is to converge as fast as possible due to the limitation of machine's hardware.

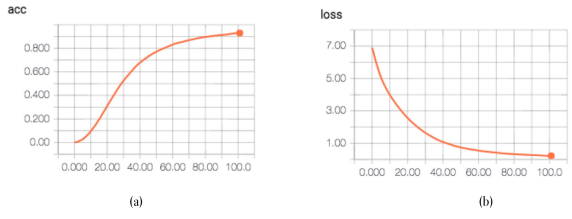

Figure 6. (a) Accuracy vs Epoch for training set
(b) Loss (categorical cross entropy) vs Epoch for training set

As can be seen in Figure 6, the model converges at around 100th epoch. Continuing the training will lead to overfitting and lower the test accuracy. I pick the result of 65th epoch as the trained model for prediction and get 32.54% accuracy for test set.

5 Pattern Discovery

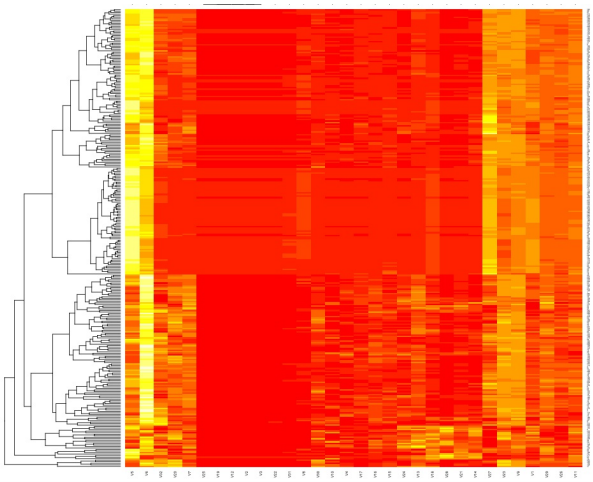

Figure 7. Heatmap (yellow means higher value) and Hierarchical Clustering for footprints using feature maps of CNN

In task3, I used the trained CNN in task2 for feature extraction. Specifically, I use the output from hidden layer who has 32 units as the feature map. Thus, each instances in task3 can be represented as a 32-dimension vector.

Then I use heatmap to visualize all these vector and hierarchical clustering to find similarities between them. In Figure 7, each column represents a feature and each row represents an instance. We can see some features are very sparse while others are dense.

After checking the cluster result, I find the instances within the same cluster usually have similar distribution of brightness, but don't show significant similarity in the patterns. The reason may be that samples in task3 are from crime scenes and is much noisier than the one in task2, so the performance of clustering is not very good if I use the feature map from task2 directly.

I also try to use other layers as the feature map of the data, but don't observe significant improvement.

PCA is also used for dimension reduction, but there is also no significant performance improvement.

6 Conclusion

In the experiment, I also try local binary pattern (LBP) and the histogram of oriented gradient (HOG) for feature extraction. But the performance is not better than using CNN directly.

For supervised learning such as task1 and task2, CNN could provide researchers better than chance performance without much expertise on computer vision and signal processing. But using these techiques for data denoising and feature extraction may still be useful and can accelerate the converging of training process. When using CNN, the main effort would be put on tuning the parameters and hyperparameters.

For unsupervised learning such as task3, the CNN can be used as an flexible way to extract features from images. Though it doesn't perform well in this project, we could try to tune the parameters, use better data denoising methods, use auto-encoder methods or other machine learning approaches to improve the performance.

In all, this project shows that CNN could be used as an alternative approach for footprint recognition.

Bibliography

[1] A. Kortylewski, T. Albrecht, and T. Vetter, "Unsupervised footwear impression analysis and retrieval from crime scene data," in *Asian Conference on Computer Vision*, (), pp. 644–658, Springer, 2014.

[2] M. Matsugu, K. Mori, Y. Mitari, and Y. Kaneda, "Subject independent facial expression recognition with robust face detection using a convolutional neural network," *Neural Networks*, vol. 16, no. 5, pp. 555–559, 2003.