Convolutional Neural Network Impletation of Vehicle Re-identification

Yuanyuchen Li

Research School of Computer Science, Australian National University
u7102698@anu.edu.au

Abstract. Vehicle re-identification is a controllable and lightweight application problem of studying the appearance hierarchy between synthetic data and real data. Due to the uncertainty of lighting, shooting distance and angle, achieving accurate vehicle recognition in reality is still a challenge. This article uses ResNet-based convolutional neural networks to solve the classification problem of vehicle types. It is expected that a deep learning model that can distinguish 1362 vehicle types can be trained by using a data set containing more than 70,000 vehicle pictures. In addition, We explore the advantages of deep learning models over traditional classification methods. The test accuracy of the deep learning framework reached 79.88%, the test accuracy of the decision tree reached 0.44%, and the test accuracy of the maximum likelihood classification reached 5.81%. The results show that the deep learning framework has great potential and far exceeds the traditional model.

Keywords: Convolutional neural network· Vehicle re-identification· ResNet

1 Introduction

The research of computer vision has always been a very hot topic in machine learning. In the data set processing of computer vision, the use of synthetic data provides convenience for problem research. However, there are many domain differences between synthetic data and real data. This difference is divided into two parts, namely the content level and the appearance level. In subsequent research, in order to focus on the research of appearance level attributes such as lighting and viewpoint, the synthetic vehicle images generated by VehicleX were used for analysis. As using synthetic pictures and real-world pictures can bring the same effect [2], researchers can use the former to reduce research costs.

In order to analyze these synthetic pictures and meet the Re-idefication requirements, I used these pictures to train a deep learning neural network. After preprocessing that satisfies the input conditions, I used the ResNet18 convolutional neural network to process the pixel information of the picture to achieve the classification effect. The data volume of training set, validation set and test set are 45437, 15141 and 14935 respectively. Each data item is a 256x256 color picture with a label range of 0 to 1361, representing different types of vehicles.

On the basis of using CNN, some optimization methods were discussed and applied to neural networks to observe their effects. At the same time, the feature value set obtained from the image set is used to train traditional models, such as decision trees and maximum likelihood classification. Compare the deep learing method with traditional classification methods and discuss the advantages and disadvantages of neural network methods.

2 Method

2.1 Database

The main data set used in this paper is an image data set, which is generated by a large-scale synthetic data set generator named VehicleX. VehicleX has a variety of different realistic backbone models and textures, allowing it to adapt to differences in real data sets. It has 272 backbone bones made with Unity3D. The backbone includes cars, SUVs, trucks, hatchbacks, and so on. Each backbone network represents a vehicle model of the real world. From these backbones, we obtained 1362 variances (identity) by adding textures or attachments of various colors. Research has shown that the use of these composite pictures is no different from the use of pictures collected from the real world. This dataset will be used in the neural network.

Fig. 1. Sample of the VehicleX dataset

After purining the image data set with ResNet and preprocessing with ImageNet, I got a digital feature value set. The dimensionality of these data has been reduced and purified, and can be used as features in classification tasks. The data volume of training set, validation set and test set are also 45437, 15141 and 14935 respectively. Each item of data has 2048 dimensions, and the label range is 0 to 1361, representing different types of vehicles. I will use this data set in my research on decision trees and maximum likelihood classification.

0. 3294625	0. 3691084	0. 6205224	0. 4750478	0. 5833543	0. 3347161	0. 3133748	0. 3687446	0. 3757017	0. 5627502	0. 4673636
0. 5991998	0. 5505929	0. 8346471	0. 435633	0. 4541471	0. 1706953	0. 2855304	0. 5905021	0. 6851563	0. 6147128	0. 421284
0. 1570985	0. 4201204	0. 509716	0. 2243121	0. 3911553	0. 3435935	0. 6647816	0. 7799565	0. 3358556	0. 2484883	0. 3387815
0. 0401992	0. 3552457	0. 2596585	0. 5383728	0. 2238927	0. 1116581	0. 2193225	0. 8083723	0. 7170904	0. 1127235	0. 3819076
0. 6840283	0. 461172	0. 2578148	0. 668069	0. 3743299	0. 3438612	0. 5000879	0. 5754417	0. 9283251	0. 6701136	0. 0937451
0. 8319767	0. 407053	0. 3621393	0. 640606	0. 4566042	0. 1541253	0. 4859119	0. 697477	0. 6946763	0. 1790653	0. 6528742
0. 1682032	0. 508343	0. 7209216	0. 4821593	0. 2717872	0. 2103308	0. 6009159	0. 3318323	0. 5777389	0. 3987654	0. 7287201
0. 4911138	0. 49641	0. 5179074	0. 2911561	0. 3854872	0. 1999943	0. 2617468	0. 3917596	0. 3480878	0. 3260016	0. 4612859
0. 2191746	0. 4323136	0. 7600753	0. 293371	0. 6083024	0. 2787011	0. 3290166	0. 7111849	0. 6333715	0. 4761908	0. 5024156
0. 2623044	0. 4773476	0. 2290809	0. 2669389	0. 4812857	0. 3252694	0. 3618866	0. 6100514	0. 5780823	0. 5269315	0. 3615378
0. 5227714	0. 4244595	0. 7475728	0. 3103485	0. 5278064	0. 1921196	0. 3261023	0. 6933016	0. 4714759	0. 3989732	0. 5865555
0. 564481	0. 3670584	0. 8453998	0. 4857497	0. 4580786	0. 4475068	0. 1362297	0. 3987131	0. 3108255	0. 2943007	0. 3656059
0. 2235329	0. 6014188	0. 6622422	0. 2331756	0. 571642	0. 3109052	0. 3278198	0. 2432441	0. 2446064	0. 2824582	0. 3808669
0. 7606444	0. 6216917	0. 7286307	0. 4736606	0. 3705908	0. 2996659	0. 3955407	0. 6268233	0. 7701343	0. 3840159	0. 3940026
0. 6172516	0. 2217561	0. 5430483	0. 2527024	0. 5082309	0. 4325431	0. 5606291	0. 5374238	0. 3789797	0. 135304	0. 6127789
0. 5641649	0. 6592751	0. 3866805	0. 2703236	0. 2881483	0. 1105583	0. 5788145	0. 3325372	0. 7714909	0. 4223748	0. 1743487
0. 6325347	0. 403735	0. 4455574	0. 480201	0. 4751798	0. 3600277	0. 3126373	0. 4123287	0. 7264388	0. 1719665	0. 3994827
0. 5102139	0. 3282335	0. 3016165	0. 39149	0. 3317146	0. 465124	0. 3865576	0. 2816098	0. 4532855	0. 4542966	0. 4302926
0. 2737458	0. 5407661	0. 3038607	0. 5378915	0. 5686678	0. 0355791	0. 3251799	0. 5899655	0. 6833562	0. 4341244	0. 4799656
0. 6774045	0. 6414974	0. 1576767	0. 3604913	0. 5722045	0. 1885345	0. 4758395	0. 5869303	0. 4758053	0. 2203906	0. 6083592

Fig. 2. Sample of the feature value data set.

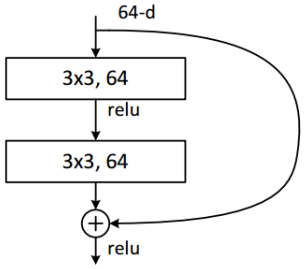
2.2 Framework

In order to use deep learning methods to classify and analyze image data sets, I built a convolutional neural network and used ResNet18 as the network architecture. And other components were discussed to achieve the best simulation results.

ResNet18 In our overall impression, the deeper the deep learning (complex, more parameters), the stronger the expression ability. Relying on this basic standard, the number of layers of the CNN classification network continues to increase, but later it was discovered that the deep CNN network reached a certain depth, and then blindly increasing the number of layers did not bring further improvement in classification performance, but a degradation problem.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

(a) Architecture diagram given in the ResNet paper



(b) Basic-block used in ResNet18.

Fig. 3. Authoritative illustration

The ResNet network [3] uses multiple parameterized layers to learn the residual representation between input and output, instead of directly trying to learn the mapping between input and output. Experiments show that the former has a faster convergence rate and can achieve higher classification accuracy by using more layers. At present, ResNet has become the basic feature extraction network in general computer vision problems.

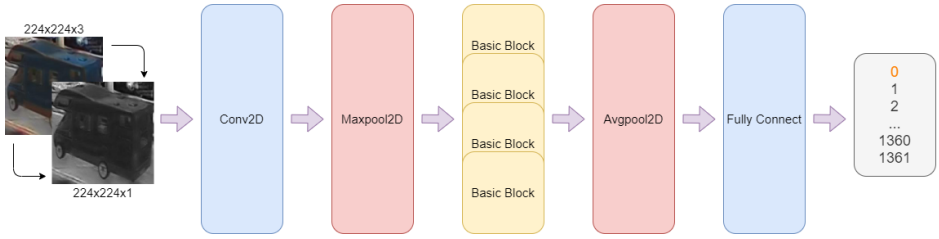


Fig. 4. ResNet18

The difference between ResNet networks is mainly due to the difference in the block parameters and number of the intermediate convolution part, and ResNet18 is the smallest among them. The CNN would first downsample the input a convolution layer and then a max pooling layer, whose output would then be sent into the ResBlocks. We build thefour ResBlocks according to Fig. 3(b). At last, we adoptaverage pooling and then flatten the output to a vector of length 512. The bidirectional layer would finally yield the prediction.After each convolution and before activation, we adopt batch normalization (BN).

C4.5 Decision Tree In machine learning, a decision tree is a predictive model, which represents a mapping relationship between object attributes and object values. C4.5 program [4] was used to derive a set of rules for classifying the feature value, which is a program based on the ID3 algorithm for generating a knowledge base from data set. It selects the splitting attribute through the information gain rate.

$$SplitInfo_A(S) = -\sum_{j=1}^{m} \frac{|S_j|}{|S|} log_2 \frac{|S_j|}{|S|}$$

Among them, the training data set S is divided into m sub-data sets by the attribute value of the attribute A, $|S_j|$ represents the number of samples in the j-th sub-data set, and $|S|$ represents the total number of samples in the data set before the division. The information gain rate of the sample set after splitting by attribute A:

$$InfoGainRation(S, A) = \frac{E(S) - E_A(S)}{SplitInfo_A(S)}$$

When the decision tree is constructed by the C4.5 algorithm, the attribute with the largest information gain rate is the split attribute of the current node.

Maximum likelihood classificatione Maximum likelihood classification is an image classification method, which assumes that various distribution functions are normal distributions, and is used to calculate the attribution probability of each sample area to be classified, also known as Bayesian classification. According to the Bayesian formula:

$$P_{(Y_i|X)} = \frac{P_{(X,Y_i)}}{P_{(X)}} = \frac{P_{(X,Y_i)}P_{Y_i}}{\sum_i^n P_{(X|Y_i)}P_{Y_i}}$$

Finally launched [5] the discriminant function $g_i(x)$

$$g_i(x) = -ln|\sum_i| - (x - m_i)^2 \sum_i$$

where m_i is mean of class i ,and $\sum_i$ is covariance matrix for class i.

2.3 Optimization

It can be seen from the above data that there is overfitting in the training of the neural network. In response to this shortcoming, I tried to use a variety of methods to optimize.

Preprocess I use Resnet18 implemented by pytorch [8]. Since training with 3 channels is very slow, I used grayscale images for training. Due to the limitation of input requirements, the picture was converted from 256x256 to 224x224 format. For the pixel information of each picture for each picture, I used the histogram to equalize, and then perform the normalization operation. It is hoped that such processing can reduce the difference caused by the data size and make the model converge in a more accurate direction.

For feature value data set,I normalized the data and shuffled it to reduce the uneven weight caused by the numerical value.

Optimizer setting The loss function of the neural network uses the cross-entropy cost function. When choosing the optimization method, I compared the two main methods, SGD and Adam. After testing, I chose Adam as the optimization method.

Optimizer	Epoch	Learning Rate	Train Accuracy	Val Accuracy
Adam	35	0.001	100.00%	78.24%
SGD	35	0.001	98.74%	67.38%

Table 1. Optimizer training

About the best choice There are many attributes that can be used to judge the training results, such as training loss, training accuracy, and validation accuracy. It can be seen from the training accuracy rate and the validation accuracy rate that the structure has obvious overfitting problems, so most judgments use the validation accuracy rate as the criterion.

Explore the parameters required by the CNN optimizer, including learning rate, weight decay rate and corresponding settings.

For the learning rate, I set a drop every 15 epochs, down to 50% of the original. I measured the choice of the learning rate and weight decay rate, and the results are as follows

Learning rate	Epoch	Weight decay	Train Accuracy	Val Accuracy
0.01	25	0	96.35%	57.85%
0.005	35	0	99.65%	65.05%
0.001	35	0	100.00%	79.14%
0.0005	35	0	100.00%	74.71%
0.0001	35	0	97.97%	57.77%
0.001	35	0.001	96.66%	56.98%
0.001	35	0.00001	100.00%	78.24%
0.001	35	0.0000001	100.00%	79.64%

Table 2. Learning rate and Weight decay training. As the number of learning rate drops, the accuracy of the validation set increases. After 0.001, the accuracy of the validation set decreases for jumping into the local optimal solution. I choose 0.001 as learning rate and 0.0000001 as weight decay rate.

Batch size and training epoch Generally speaking, the larger the batch size of CNN training, the better. Due to the limitation of GPU memory, I set it to 512. In order to balance over-fitting and under-fitting, the training period of model training was adjusted and the loss was recorded.

Epoch	Learning rate	Val Loss	Train Accuracy	Val Accuracy
10	0.001	82.5961	91.9671%	39.4282%
15	0.001	32.0422	99.9098%	71.6993%
20	0.001	24.0060	100.0000%	78.9100%
25	0.001	24.0609	100.0000%	79.1979%
30	0.001	24.1029	100.0000%	79.2783%

Table 3. Epoch training. As the number of training increases, the accuracy of the validation set will also increase. After 20 iterations, the loss no longer decreases, and the accuracy of the verification set stabilizes. For stability, the most suitable training epoch is selected as 25.

3 Results and Discussion

Since the threshold method cited in the paper [1] is only suitable for binary classification, not for this data set and target, the other two tranditional prediction models mentioned by it are used. After many optimizations and experiments, I have obtained relatively good neural network parameters. The parameters and test results in the validation set and test set are shown below. I also use decision trees and maximum likelihood classification to analyze the data set, and analyze the different effects of these two "medieval" classification methods.

3.1 Final Result

TestLoss	Train Accuracy	Val Accuracy	Test Accuracy
0.6716	100.00%	79.58%	79.88%

Table 4. Final Result

3.2 Comparison with other traditional classification algorithms

I built decision tree [6] and Bayesian classification function [7] by calling the Sklearn package. The final accuracy rate obtained using C4.5 is 0.44%, and the final accuracy rate obtained using maximum likelihood is 5.81%, which is far lower than the 79.88% using neural network.

In fact, using decision trees for image analysis tasks is inherently impossible, as using feature values that contain non-interpretable information to deal with the interrelationships between pixels extensively. At the same time, using decision trees to explore such large data sets is relatively complex and time-consuming. The final training model is also very large, and the depth and complexity of the decision tree is terrifying. But for the 1362 classification target, an accuracy of 0.44% means that it at least made some effort.

Maximum likelihood also hard to be used for image recognition. And compared with decision trees, Bayesian classification is much better in processing large data sets, and the processing speed is also very fast. The accuracy

rate of 5.81% shows that it is more suitable for feature value data sets than decision tree. However, compared with the neural network and decision tree, it cannot process the changed results based on the feature combination.

It can be said that due to the large-scale and content nature of the image data set, it is impossible for traditional neural networks to complete this task. At the same time, the defects of neural networks are also obvious. The first is the "black box" nature. Decision trees can clearly express the classification process and principles, which is impossible for neural networks. Moreover, neural networks are more computationally expensive than traditional algorithms, and the amount of training calculations and training duration are much higher than traditional algorithms.The scale of the database used in this study is fairly appropriate. If the amount of data is small, the neural network may not be successfully constructed.

4 Conclusion and Future Work

4.1 Conclusion

In this article, I use deep learning and traditional models to re-identify 1362 individual vehicles. I built and optimized the ResNet18 convolutional neural network and used it for data classification. The accuracy obtained on the verification set is 79.58%, and the accuracy obtained on the test set is 79.88%. Using traditional models such as decision trees and maximum likelihood classification, accuracy rates of 0.44% and 5.81% were obtained. The results prove that the deep learning model has great potential and is far superior to traditional methods.

4.2 Future Work

The deep learning architecture still has room for optimization. Using deeper and more complex architectures may be able to obtain higher training results, such as changing the input from grayscale images to multi-channel images, using more advanced ResNet or other architectures.

After obtaining a judgment model that treats synthetic pictures and real-world pictures equally, we can apply more synthetic pictures to vehicle re-recognition, and then to other target recognition tasks. This will greatly accelerate the progress of computer vision in real-world applications.

References

1. Milne L K, Gedeon T D, Skidmore A K. Classifying Dry Sclerophyll Forest from Augmented Satellite Data: Comparing Neural Network, Decision Tree & Maximum Likelihood[J]. training, 1995, 109(81): 0.
2. Yao Y, Zheng L, Yang X, et al. Simulating content consistent vehicle datasets with attribute descent[J]. arXiv preprint arXiv:1912.08855, 2019.
3. He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.
4. Salzberg, Steven L. "C4. 5: Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993." (1994): 235-240.
5. Richards, John Alan, and J. A. Richards. Remote sensing digital image analysis. Vol. 3. Berlin: Springer, 1999.
6. Sklearn Decision Trees, https://scikit-learn.org/stable/modules/tree.html. Last accessed 25 Apr 2021
7. Sklearn Naive Bayes, https://scikit-learn.org/stable/modules/naive_bayes.html. Last accessed 25 Apr 2021
8. pytorch implement resnet18https://www.jianshu.com/p/7fc2f206ceaa