MACHINE LEARNING ENGINEER NANODEGREE
CAPSTONE PROJECT

ZIAD AMR ELALAILY 31/03/2019

DOMAIN BACKGROUND

Since the day the car was made and there has been an increasing demand for car purchase across the globe. Cars took over the roads and started becoming the main transportation method to many people in the modern times.

With the increasing technological advancements in driving, controlling and producing the car, comes a need to track and analyze the diverse sea of car brands that exist today. Tracking and analyzing car brand has been very difficult and not cost-effective [1].

Vehicle Make and Model Recognition (VMMR) is a very effective way to recognize car brands in many aspects of life. A brand seller can use data mining to check which color is favored for a certain city or neighborhood [2] or check the main location where his customers go shop and push services related to their car models.

In another situation VMMR complements Automatic Number Plate Recognition (ANPR) [3] in many situations such as police chase, traffic surveillance and parking lot car finder [4], in another case VMMR strongly supports ANPR in the case of plate forgery due to theft or damaged plate number or removing the plate number all together. Even if the car has been painted, its model shape can never change that gives a huge advantage over ANPR methods.

PROBLEM STATEMENT

The aim of this project is to train a machine learning algorithm to help recognize cars from pictures and footages of surveillance cameras spread across the streets. The model should be able recognize the car model and make to a good extend with a good accuracy to the year of manufacture.

Due to the huge number of classes existing within this problem (e.g. Nissan, Audi … etc.) and inter-class distance (e.g. Nissan Altima, Nissan Sunny… etc.) this problem is classified as a hard classification problem [4].

I will be measuring the accuracy of my algorithms by comparing it to the labelled data in the dataset and reach an ideal and close to perfect results. I shall try to implement the algorithm on many devices (if available) and see the speed of recognition of several cars and their accuracy.

DATASETS AND INPUTS

I will use a dataset available publicly from https://ai.stanford.edu/~jkrause/cars/car_dataset.html [5]. This dataset consists of 16,185 images, divided upon 196 classes of cars models, the data is already split between 8041 images for testing and 8144 images for training, in which each class was divided almost equally.

The dataset contains different backgrounds and different angles of many car models, as well as random noise (watermarks) appearing as well.

SOLUTION STATEMENT

My project will focus on deep learning algorithms mainly, it's because Deep learning has proved to be great in determining the main features in an image and is usually used in image classification algorithms.

I will use my own CNN architecture, as Convolution neural network is very good at finding patterns in images by filtering and finding grouped pixels who are related to each other. As a start I shall implement CNN from scratch and try to reach a good accuracy.

However, if my CNN yielded low accuracy results, I may try to use transfer learning based on VGG-16 or Xception to try and reach better accuracy.

BENCHMARK MODEL

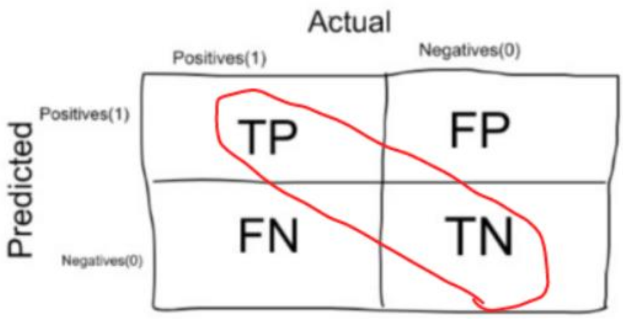
My bench mark model will be comparing my accuracy results with results found in Yaqoob, Hashir, Shaharyar Bhatti, and Rana Raees Ahmed Khan. "Car Make and Model Recognition using Image Processing and Machine Learning."

The authors followed two approaches; the first approach was to use Bag-Of-words with SURF supported by Multi-class SVM. In the second approach it is seen that they use CNN for the extraction of features in training, they also used transfer learning on pre-trained AlexaNet network. They also used accuracy as their metric evaluation method in their paper for model evaluation.

EVALUATION METRICS

To compare my results with the previous paper, I shall implement accuracy as well. And since my classes are balanced, the equation shall be:

$$\text{Accuracy} = \frac{\text{Correctly classified points}}{\text{All points}}$$



$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

The higher the accuracy the better the results

PROJECT DESIGN

DATASET

As described before I shall use the dataset from Stanford which is openly available to the public. I will make sure that the classes have equal samples (for accuracy calculations). There is no need to augment the data as it has cars from different angles.

My dataset is split between 8041 images for testing and 8144 images for training and shall be one-hot encoded.

PRE-PROCESSING

I shall try to approaches here, first I will use a car detector algorithm provided by OpenCV with CascadeClassifier and use a cropped version of the car's image. This approach saves training time as part of the image is used which contains the important data

And in my second approach I shall leave my CNN to train on the whole image without cropping in an attempt to have a more adequate algorithm to be used on the road. However, there will be a cost of training time (if the time taken was too long for each epoch, I may drop this approach and head on with cropped images).

TRAINING (CNN LAYERS)

As mentioned before Deep learning is my go-to algorithm for this problem and utilizing convolutional neural networks would be suitable for the image classification problem.

I will start with 4 Convolutional Layers, and since I am a fan with ascending numbers (plus showed good results on previous projects) I shall build my 2D layers as follows:

1. Start with a Conv2D with 32 filters and have an input of the same size as the cropped image (or in case of full image this will change to the original image size) and a relu activation function followed by a max pooling layer with pool size of 2 and stride of 4.
2. Conv2D with 64 filters and a relu activation function followed by a max pooling layer with pool size of 2 and stride of 4.
3. Conv2D with 128 filters and a relu activation function followed by a max pooling layer with pool size of 2 and stride of 4.
4. Conv2D with 256 filters and a relu activation function followed by a max pooling layer with pool size of 2 and stride of 4 and global average pooling layer.
5. Dense Layer with softmax activation function.

Dropout layers may be added if needed if I find problems with a certain set of images.

The hyperparameters will be tuned till an adequate performance and results are reached, and the model will be trained for multiple epochs till convergence is seen.

I will be using a Multi-Class Classification loss function as my evaluation metrics, due to its suitability for the 192 classes available. And since my dataset will be hot-encoded, I shall use cross entropy as my loss function. The average difference between the actual and predicted probability distributions for all classes in the problem will be summarized through a score that is calculated by cross-entropy. The lower the score the better and a perfect score for the cross-entropy would be 0. [6].

Some refinements that may be used:

- As mentioned before, I can utilize transfer learning to achieve better results on pretrained models like VGG16.
- Grid-search can be used to fine tune the hyperparameters

POST-PROCESSING

Since I will be using accuracy, accuracy of my model will be tested against my test image dataset of 8041 images available in the dataset and as mentioned before in the Benchmark section I will compare my accuracy score against the paper specified there.

I would like to see if a certain class is having difficulty in predicting, so I will implement a confusion matrix in my code and observe the deficiency (if any) and try to better my model in general.

REFERENCES

[1] Ullah, Ihsan. (2017). Vehicle Make and Model Recognition System based on Constitutional Neural Network.

[2] Yaqoob, Hashir, Shaharyar Bhatti, and Rana Raees Ahmed Khan. "Car Make and Model Recognition using Image Processing and Machine Learning."

[3] Emami, Hajar, Mahmood Fathi, and Kaamran Raahemifar. "Real time vehicle make and model recognition based on hierarchical classification." International Journal of Machine Learning and Computing 4.2 (2014): 142.

[4] Biglari, Mohsen & Soleimani, Ali & Hassanpour, H. (2017). Vehicle Make and Model Recognition using Auto Extracted Parts.

[5] Krause, Jonathan, et al. "3d object representations for fine-grained categorization." Proceedings of the IEEE International Conference on Computer Vision Workshops. 2013.

[6] https://machinelearningmastery.com/how-to-choose-loss-functions-when-training-deep-learning-neural-networks/