Assignment 3

Aim :

Extraction of base paper

Objective :

To study algorithms/tools/technologies used and analyse the topic.

Introduction

Humans use locally available information that is gathered by our senses, mainly sight. This information is not computed. This work explores and evaluates the development of autonomous steering using visual-only input in real time driving. Convolutional Neural Networks (CNNs) have been proven to excel in categorical image classification, but little has been done on how they could perform in continuous spaces such as deciding steering values.

Stategies that use visual data has been applied to General Game Play as the rst implementation to use visual information as input to neural network. This showed how a recurrent neural network lay- out could be constructed using evolutionary algorithms to navigate a simple track. Those were imposed in small parts due to high com- putation and variation of visual input but now GPGPU has made possible to implement more complex model in genetic algorithms and most neural networks. CNNs are one such model, they consists of series of partially-connected layers with local connectivity and shared weights to allow detecting patterns irrespective of position of image. The System implemented Steer-CNN is a TORCS bot that uses neural network for controlling steering. Steer-CNN focuses on pre- dicting steer commands only, all other decision such as gear change, acceleration, brake are made by hardcoded policy.

Algorithm Overview

The system takes as input the rendered image from a third person perspective from the top of the car facing forwards- and passes it to a pre-processing module, which crops it to discard the borders and applies one or more filters, on different filters. The final image is a greyscale 1-byte depth 144x100 structure that acts

as the input for the CNN. Each input unit value is scaled ,so values from 0 to 255 are transformed to 0 to 1. The CNN outputs a single value that is used as the steering command, which is combined with the remaining driving decisions from the hardcoded policy and returned to the engine.

The Main step of this algorithm is -

1. Image is given as an input to system taken from top of the car.
2. Image is converted to 144x100 greyscale for processing.
3. Each input value to neural network is scaled to 0-1 from 0-255
4. CNN gives single value output which is used as a steering command.

Convolutional Network Architecture

CNN is based on work of LeNet-5 (achieve very good accuracy rates on handwriting recognition). Feature that makes it a good candidate for our problem is its ability to detect patterns irrespective of their position on the image, a common trait of CNNs due to the use of shared weights. The original model has been scaled up number and size of feature maps to deal with higher resolution images:

- Convolutional layer (CN) 1: 144x100 inputs, 30 output channels, 10x10 kernels.
- Subsampling layers: down sampling x2, stride 2.
- CN 2: 67x45 inputs, 50 channels, 6x6 kernels.
- Multilayer perceptron (MLP): 2 layers (31000x500) with 100 or 200 output units.

LeNet-5 gives categorial output in range 0 to 9 but in CNN for TORCS we need output in continuous range from -1 to +1. Hence to use this model, steering space is discretedinto number of steps. There is little impact on drive provided no of steps is high.100 or more divide the 2 units interval into steps of less than 1 degree per step, which is more acceptable solution. CNN is trained based on high variety of of race situations and dierences in track width to provide heterogeneous dataset to CNN. The CNN is trained using Stochastic Gradient Descent, a batched algorithm that uses a very small subset of the training data on each iteration to modify the weights and bias of the network. It has been shown to be resilient to overtting. The batch size used is 32 and training sessions were run for 50000 iterations.

Learning Framework

Data gathering

The CNN is trained from a hardcoded line-following driving policy. Whilst driving, 10 data samples are gathered per second, recording:

1) Input image: the image is stored as an array of pixel values, where each is a 1-byte greyscale value (0- black, 255-white);

2) Expected output: discretized steering value determined by the driving policy.

The tracks were chosen based on their high variety of race situations and differences in track width, to provide a heterogeneous dataset to the CNN.

Network training

The CNN is trained using Stochastic Gradient Descent, a batched algorithm that uses a very small subset of the training data on each iteration to modify the weights and bias of the network. It has been shown to be resilient to overfitting. The batch size used is 32 and training sessions were run for 50000 iterations.

Prior to training, the data is normalised so all pixel information is provided within the range 0 to 1.

Performace of CNN

Improving Performance of CNN :

The performance of CNN is measured on dierent parameters, and there parameters decide whether the CNN is improving or not. Performance is calculated on basis of:

- Mean Squared Error - Average square distance between ex- pected and actual network output.
- Time Stuck - Stuck when speed is less than 5km/hr or out of bounds
- Damage - Result in collision
- Lap time

Different techniques have been used to improve working of CNN, some of them is given below:

1. Pre-processing image prior to feeding CNN:

Pre-processing the image has an impact on the nal driving performance. If the image is pre-processed to abstract certain key elements, generalization may get improved.

2. Scaling up Output:

If two dierent CNN are used, both with identical topology but varying in the number of output units on the last layer:

1. CNN_100: 100 output units
2. CNN_200: 200 output units

An increased steering resolution from 100 to 200- leads to an enhancement of driving performance (reduced lap time, stuck time and damage taken) and prediction accuracy.

3. Optimal output selection policy:

On a LeNet-5 network, the nal output is chosen by a winner takes all policy, this is not clear whether it is the best strategy for continuous space such as steering. a winner- takes-all selection criterion would reduce the network's ability to incorporate this. we compared the classication results of a winner-takes-all selection strategy with a weighted average that uses the values from all output units. The results indicate that the network achieves better accuracy using the weighted average, with a lower average error.

4. Diversity of Training data:

So far all CNNs have been trained using data from a single track. It includes a variety of race situations that make the CNN generalize to driving on other track. Thus training with a more diverse dataset improves performance of CNN.

5. Learn from Steer-CNN driver:

Naturally the performance of the network on some situations is poor and in fact it accounts for most of the situations where Steer-CNN gets stuck.training data, which is then used to train the next generation. Here we use a modication of that process, where a new training set is generated in a combined eort, using an acceptable but imperfect driver trained from a hardcoded policy to navigate around the track, but recording the expected output from the perfect driver using the line following policy.

Conclusion

Steer-CNN results have demonstrated that a visual-only driving policy using CNNs is feasible for real-time decision making in the TORCS environment. In addition, this work has laid out several optimization elements that lead to quantitative and qualitative improvements on driving performance and network accuracy:

- Network generalization is improved when pre-processing the image to abstract relevant features.
- Using weighted average to select CNN output units instead of a winner-takes-all method yields lower prediction error and produces more accurate driving.
- A varied training data set tends to improve performance both in known and unknown tracks.
- Using a suboptimal driver to gather training data in conjunction with a hardcoded policy significantly improves driving performance, allowing the driver to complete difficult tracks that were not possible before.

Even though the results are promising, further work must be carried out to fully realize an autonomous neurovisual driver. The following are areas of potential improvement:

- Incorporate other sensory data to the policy as well as previous steering history and generate CNNs to control other driving decisions –accelerate and brake
- CNN architecture: experiment with other CNN architectures and explore the use of automation tools to optimize CNN hyperparameters and layers.