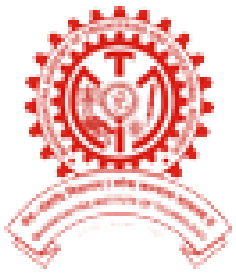
A SEMINAR REPORT ON

An Effective Biologically-motivated Neural Network Algorithm to Real-time Autonomous Robot Navigation

SUBMITTED BY

Aditya Jain
Roll No : 302029

UNDER THE GUIDANCE OF
Prof. Dr. Nitin Pise



Department Of Computer Engineering
MAEER's MAHARASHTRA INSTITUTE OF TECHNOLOGY
Kothrud, Pune 411 038

2016-2017

MAHARASHTRA ACADEMY OF ENGINEERING AND EDUCATIONAL RESEARCH'S

MAHARASHTRA INSTITUTE OF TECHNOLOGY PUNE

DEPARTMENT OF COMPUTER ENGINEERING

C E R T I F I C A T E

This is to certify that

Aditya Jain (302029)

of T. E. Computer successfully completed seminar in

An Effective Biologically-motivated Neural Network Algorithm to Real-time Autonomous Robot Navigation

to my satisfaction and submitted the same during the academic year 2016-2017 towards the partial fulfillment of degree of Bachelor of Engineering in Computer Engineering of Savitribai Phule Pune University under the Department of Computer Engineering , Maharashtra Institute of Technology, Pune.

Prof. Dr. Nitin Pise
(Seminar Guide)

Prof.(Dr.)V.Y.Kulkarni
(Head of Computer Engineering Department)

Place: Pune

Date:

ACKNOWLEDGEMENT

I take this opportunity to express my sincere appreciation for the cooperation given by Prof. Mrs. V. Y. Kulkarni, HOD (Department of Computer Engineering) and need a special mention for all the motivation and support.

I am deeply indebted to my guide Prof. Dr. Nitin Pise for completion of this seminar for which she has guided and helped me going out of the way.

For all efforts behind the Seminar report, I would also like to express my sincere appreciation to staff of department of Computer Engineering, Maharashtra Institute of Technology Pune, for their extended help and suggestions at every stage.

Aditya Jain
(Roll No.:302029)

Contents

List of Figures

List of Tables

Abstract

Majority of previous attempts to autonomous driving were based on use of pre-calculated data such as exact distance to other cars or actual position of cars with respect of center of track. As humans, we drive on knowledge locally available information that is gathered by or senses, mainly sight. This works explores and evaluates the development of autonomous steering using visual-only input. Convolutional Neural Networks have been proven to excel in categorical image classification, and can be used to deciding steering values. Different pavement detection have also been discussed here, which can be used to detect pavement mark on an image of road in front of car, which can help car in making right decisions. Object recognition and pedestrian detection are of crucial importance to autonomous driving applications. Bag of visual Words (BOW) approach by using SURF, HOG and k-means have been discussed for object and pedestrian detection. Self-driving cars require robust and fas t path planning algorithms to operate in dynamic environment. A model based path planning algorithm for dynamic environment have been discussed. This approach generates candidate trajectories online, then evaluates and selects the most appropriate one according to real-time environment information.

1 Introduction

1.1 Into the topic

Previous attempts to implement autonomous driving include use of different parameters gathered from different sensors installed on car, the computer perform different calculations on these gathered parameters and make steering decision. But this is now how brain operate, brain take decisions mainly based on visual data. Neural network is an algorithm which simulates brain environment, neural networks learn by taking different examples. Convolutional neural networks can take visual data as an input to make decisions. Neural networks requires too much computation, which takes too much time to process on normal processors. So neural networks have been not been implemented to make real time decisions like for autonomous driving. Since development of GPGPU have provided us with parrallel computation at many nodes simultaneously, which can process neural networks very efficiently and make real time decisions. Thus, now convolutional neural networks can be implements for autonomous driving thanks to GPGPU. Different other algorithms have also been declared which helps self driving cars to make real time decision like pavement detection algorithm, object and padestrian detection. Different types of neural networks work together to make car running on correct path. So its open different areas of study to design different algorithm which can help automonous driving more efficient and error free.

1.2 Neural Networks

1.2.1 About Neural Networks

Information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing

elements (neurones) working in unison to solve specific problems. Neural Networks like people, learn by example.

Neural Networks is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurones.

A typical neural network has anything from a few dozen to hundreds, thousands, or even millions of artificial neurons called units arranged in a series of layers, each of which connects to the layers on either side. Some of them, known as input units, are designed to receive various forms of information from the outside world that the network will attempt to learn about, recognize, or otherwise process. Other units sit on the opposite side of the network and signal how it responds to the information it's learned; those are known as output units. In between the input units and output units are one or more layers of hidden units, which, together, form the majority of the artificial brain. Most neural networks are fully connected, which means each hidden unit and each output unit is connected to every unit in the layers either side. The connections between one unit and another are represented by a number called a weight, which can be either positive (if one unit excites another) or negative (if one unit suppresses or inhibits another). The higher the weight, the more influence one unit has on another.

1.2.2 Types of Neurons

Linear Neurons : These are simple but computationally limited. If we can make them learn we may get insight into more complicated neurons.

$$y = b + \sum xiwi$$

Binary threshold neurons : There are two equivalent ways to write the equations for a binary threshold neuron:

$$z = b + \sum xiwi$$

$$y = 1; z \geq 0$$

Sigmoid Neurons : These give a real-valued output that is a smooth and bounded function of their total input. Typically they use the logistic function.

$$z = b + \sum xiwi$$

$$y = 1/1 + ez$$

1.2.3 How does a Neural Network learn things?

Information flows through a neural network in two ways. When it's learning (being trained) or operating normally (after being trained), patterns of information are fed into the network via the input units, which trigger the layers of hidden units, and these in turn arrive at the output units. This common design is called a feedforward network. Not all units "fire" all the time. Each unit receives inputs from the units to its left, and the inputs are multiplied by the weights of the connections they travel along. Every unit adds up all the inputs it receives in this way and (in the simplest type of network) if the sum is more than a certain threshold value, the unit "fires" and triggers the units it's connected to (those on its right).

Neural networks learn things by a feedback process called backpropagation (sometimes abbreviated as "backprop"). This involves comparing the output a network produces with the output it was meant to produce, and using the difference between them to modify the weights of the connections between the units in the network, working from the output units through

the hidden units to the input units—going backward, in other words. In time, backpropagation causes the network to learn, reducing the difference between actual and intended output to the point where the two exactly coincide, so the network figures things out exactly as it should.

1.2.4 How does it work in practice?

Once the network has been trained with enough learning examples, it reaches a point where you can present it with an entirely new set of inputs it's never seen before and see how it responds. For example, suppose you've been teaching a network by showing it lots of pictures of chairs and tables, represented in some appropriate way it can understand, and telling it whether each one is a chair or a table. After showing it, let's say, 25 different chairs and 25 different tables, you feed it a picture of some new design it's not encountered before let's say a chaise longue and see what happens. Depending on how you've trained it, it'll attempt to categorize the new example as either a chair or a table, generalizing on the basis of its past experience—just like a human.

That doesn't mean to say a neural network can just "look" at pieces of furniture and instantly respond to them in meaningful ways; it's not behaving like a person. Consider the example we've just given: the network is not actually looking at pieces of furniture. The inputs to a network are essentially binary numbers: each input unit is either switched on or switched off. So if you had five input units, you could feed in information about five different characteristics of different chairs using binary (yes/no) answers. The questions might be 1) Does it have a back? 2) Does it have a top? 3) Does it have soft upholstery? 4) Can you sit on it comfortably for long periods of time? 5) Can you put lots of things on top of it? A typical chair would then present as Yes, No, Yes, Yes, No or 10110 in binary, while a typical table might be No, Yes, No, No, Yes or 01001. So, during the learning phase, the network is simply looking at lots of numbers like 10110 and

01001 and learning that some mean chair (which might be an output of 1) while others mean table (an output of 0).

1.2.5 Neural Network Architecture

Feed-Forward Neural Network

These are the commonest type of neural network in practical applications. The first layer is the input and the last layer is the output. If there is more than one hidden layer, we call them "deep" neural networks. They compute a series of transformations that change the similarities between cases. The activities of the neurons in each layer are a non-linear function of the activities in the layer below.

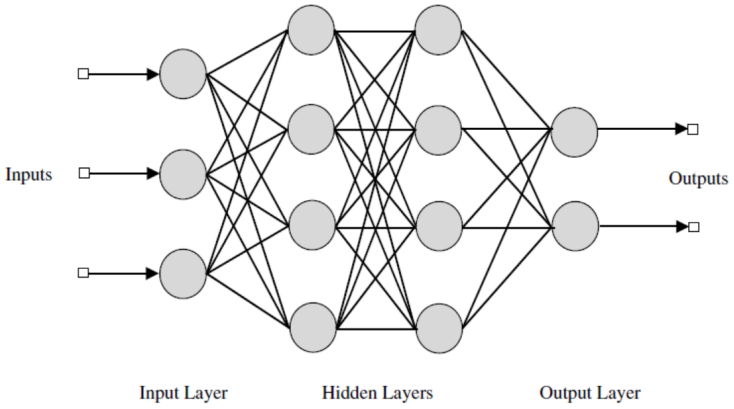


Figure 1: Feed Forward Neural Network

Recurrent Neural Network These have directed cycles in their connection graph. That means you can sometimes get back to where you started by following the arrows. They can have complicated dynamics and this can make them very difficult to train. There is a lot of interest at present in finding efficient ways of training recurrent nets. They are more biologically realistic.

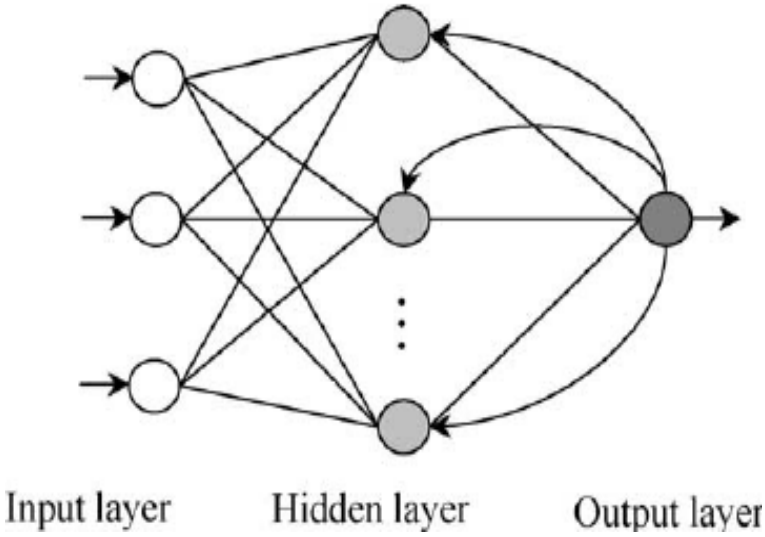


Figure 2: Recurrent Neural Networks

2 Social Relevance

2.1 Why Autonomous driving?

Among the anticipated benefits of automated cars is the potential reduction in traffic collisions (and resulting deaths and injuries and costs), caused by human-driver errors, such as delayed reaction time, tailgating, rubbernecking, and other forms of distracted or aggressive driving.

If a human driver isn't required, automated cars could also reduce labor costs; relieve travelers from driving and navigation chores, thereby replacing behind-the-wheel commuting hours with more time for leisure or work; and also would lift constraints on occupant ability to drive, distracted and texting while driving, intoxicated, prone to seizures, or otherwise impaired.

Additional advantages could include higher speed limits, smoother rides; and increased roadway capacity; and minimized traffic congestion, due to decreased need for safety gaps and higher speeds.

There would also be an improved ability to manage traffic flow, combined with less need for trafic police, vehicle insurance; or even road signage since automated cars could receive necessary communication electronically (although roadway signage may still be needed for any human drivers on the road). Reduced traffic congestion and the improvements in traffic flow due to widespread use of autonomous cars will also translate into better fuel efficiency.

Widespread adoption of autonomous cars could reduce the needs of road and parking space in urban areas, freeing scarce land for other uses such as parks, public spaces, retail outlets, housing, and other social uses. Some academics think it could also contribute, along with automated mass transit, to make dense cities much more efficient and livable.

The vehicles' increased awareness could reduce car theft, while the removal of the steering wheel—along with the remaining driver interface and the requirement for any occupant to assume a forward-facing posi-

tion—would give the interior of the cabin greater ergonomic flexibility. Large vehicles, such as trucks, would attain appreciably enhanced ease of use.

When used for car sharing, the total number of cars is reduced. Furthermore, new business models (such as mobility as a service) can develop, which aim to be cheaper than car ownership by removing the cost of the driver. Finally, the robotic car could drive unoccupied to wherever it is required, such as to pick up passengers or to go in for maintenance (eliminating redundant passengers).

3 Literature Survey

3.1 Algorithms

3.1.1 Towards Neuroimaging real-time driving using Convolutional Neural Networks

Humans use locally available information that is gathered by our senses, mainly sight. This information is not computed. This gives idea of development of autonomous driving using visual data as in real time. Convulational Neural Networks (CNNs)have been proven to excel in image classification, but little has been performed in continuous spaces such as deciding steering values.

Stategies that use visual data has been applied to General Game Play as the first implementation to use visual information as input to neural network. This showed how a recurrent neural network layout could be constructed using evolutionary algorithms to navigate a simple track. Those were imposed in small parts due to high computation 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 predicting steer commands only, all other decision such as gear change, acceleration, brake are made by hardcoded policy.

System uses TORCS rendered image from top of car as input which crops it to discard borders, applies to more 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.

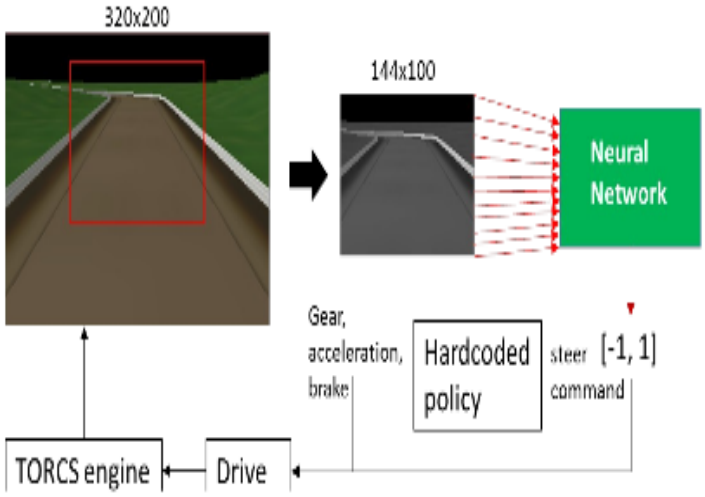


Figure 3: Steer CNN Diagram: Image to render driving decision

Compared to handwriting recognition. CNN is based on work of LeNet-5 (achieve very good accuracy rates on handwriting recognition). 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 differences 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 overfitting. The batch size used is 32 and training sessions were run for 50000 iterations.

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

1. Mean Squared Error - Average square distance between expected and actual network output.
2. Time Stuck - Stuck when speed is less than 5km/hr or out of bounds
3. Damage - Result in collision
4. 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 final driving performance. If the image is pre-processed to abstract certain key elements, generalization may get improved.

2. Scaling up Output If two different 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 final 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 classification 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 modification of that process, where a new training set is generated in a combined effort, 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.

3.1.2 Progress in Neural-based Vision for Autonomous Robot Driving

Recent improvements for neural network based autonomous driving have made neural networks make decisions in rare scenerios. The faster a network is trained, less exposure it receives to novel or in frequent scenerios. For instance during a typical 4 min run, the network sees few if any examples of passing cars. When a rare situation like this occurs during testing, its lack of coverage may result in erratic driving. By modelling appearance of infrequent scenerios we can teach network to generalize to situations not explicitly represented in live data. Performance of neural network is highly influenced by quality of its trainig sets. If important situations are missing during trainig it is likely to perform poorly. Shown a connectionist architecture, where network learn to drive by watching a person drive. The network recieves input from camera on vehicle. The network is trained by back-propagation to activate the output unit representing the drivers current steering direction. Individual networks have been trained to drive in a wide variety of sitatuins includeing single and multilane roads with and without lane markings.

Transitory Feature Problem- The problem caused by transitory image features results from insufficient diversity in the training examples. The network does not encounter these transitory features frequently enough during the short training period to learn to ignore them. One useful aspect of the problem, exploited in the solution below, is that these transitory features do not radically alter the overall appearance of the image. Due to the redundancy of features in the image, if the network could learn to ignore spurious features, it should be capable of driving accurately.

Training with Gaussian Noise- A commonly employed technique for improving generalization from a limited amount of training data is to add uncorrelated gaussian noise to the training patterns. The idea isthat adding noise to the input prevents the network from relying on idiosyncrasies in the training patterns to perform the task. There is dramatic improvement in generalization when noise is added to training patters. Gaussian noise is a poor model of the important "noise" that occurs in the input while driving.

Training with Structured Noise- Networks trained by adding structured noise to the input,generalized better on the test set as a whole than networks trained without noise or with gaussian noise.Structured noise characteristics are used during training to determine the appearance of the noise to be added to the input patterns. Instead of a gaussian noise to each pixel, the following technique is employed to add or remove coherent two-dimensional features from the training patterns. Once the decision is made to add noise to a pattern, the next question is where to put it. The location for this single noise feature is selected randomly with a bias towards the periphery of the scene, corresponding to the upper comers of the image. Adding structured noise to the training patterns using a model describing the charaaeristics of irrelevant features improves driving performance.

3.1.3 Vision-based Detection and Classification of Pavement Mark using Neural Network for Autonomous Driving System

The performance of Convolutional Neural Networks can be improved greatly, if we are able to detect pavements on the road in front of the car. Algorithm have been designed for an autonomous driving system which detects a pavement mark in an image of the road in front of a vehicle and identifies the mark. The algorithm uses edge pairing to find a pavement mark then identifies the type using a neural network which uses the horizontal and vertical projection of the founded mark as input.

Lane departure warning, blind spot detection, and in-vehicle navigation help the driver to increase car safety and more generally road safety. In in-vehicle navigation systems, position information is important. In-vehicle navigation system needs information about the vehicle's position for route planning. A positioning system provides that information; such systems are of two types: global positioning systems (GPS), and local positioning systems (LPS). GPS has a limited accuracy , so it needs an error-correction step using local information. LPSs produce the relative positions of objects, including traffic signs and buildings, around a vehicle. Pavement marks can be a good reference for an LPS.

Pavement marks are paintings on pavement. They can provide the position of the vehicle within road.The marks provide guidance and information to drivers. Lanes divide roads, and arrows indicate a direction which a vehicle can go. Pavement marks have limitations. They may be covered by snow, may not be clearly visible when wet, and may not be durable under heavy traffic. But pavement marks have a unique and important advantage: they convey information to drivers without diverting their attention from the road.

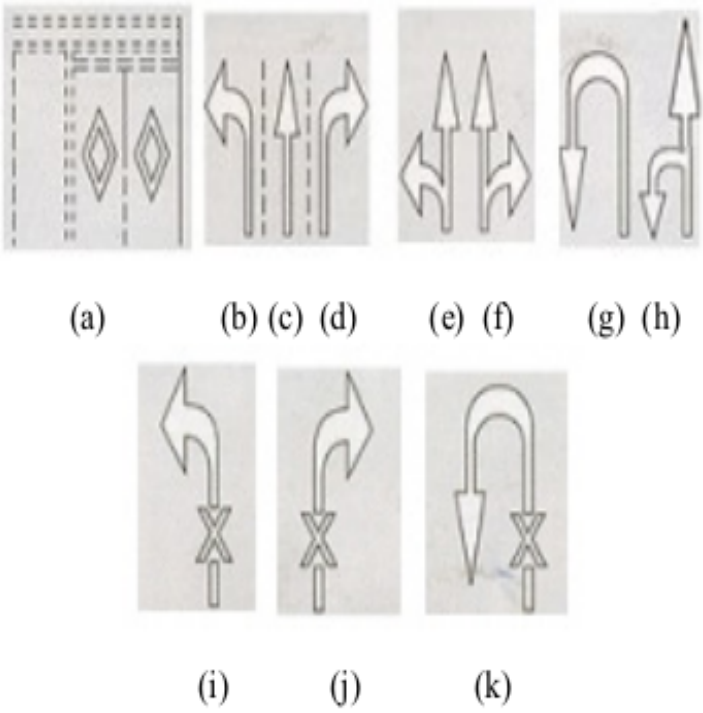


Figure 4: Target pavement marks. (a) Intersection, (b) left-turn only, (c) no turn allowed, (d) right-turn only, (e) left turn or straight, (f) right turn or straight, (g) u-turn lane, (h) u-turn or straight, (i) no left turn, (j) no right turn, (k) no u-turn.

We mounted camera on car pointed to front. We use multi-layer neural network with 60 input and 11 output(1 for each type of payment mark). The input of the neural network is a 60-dimensional feature vector. The network has two hidden layers, one with 40 nodes and the other with 30 nodes. We preprocess each input using the mapminmax function in MATLAB, the linear mapping minimum and maximum values of training data to -1 and 1. After calculating an output from a input through the neural network, we find maximum value in the output. If the maximum value is smaller than -0.2, then we decide that the network failed to identify the mark. Otherwise, we identify the mark as the index of the maximum value.

The algorithm uses edges to find the region in the image that includes the pavement marking then makes a binary image of the region. Then the algorithm generates a vector that contains the horizontal sum and the vertical sum of the image. A neural network, which uses the vector as the

inputs, identifies the type of pavement mark. With the result of pavement mark recognition, we can provide more accurate positioning information to a driver than the information from GPS.

3.1.4 Moving Towards in Object Recognition with Deep Learning for Autonomous Driving Applications

Object Recognition and pedestrian detection are of crucial importance to autonomous driving applications. Deep learning methods exhibit very large improvements in accuracy and fast decision. 2 CNN approaches are CNN in AlexNet architecture and Bag of visual Words approach by using SURF, HOG and k-means.

Recently application of of object recognition to vehicles in real life have rapidly grown. Some examples are pedestrian detection warning system using infrared images and detection of vehicles on road ahead using laser and single lens camera system. Object recognition is one of the main challenges in the field of computer vision. In the recent years, deep learning methods have emerged as a powerful machine learning method for object detection and recognition. Deep learning methods are different from all traditional approaches. They automatically learn features from raw pixels directly and in a fast way more complex models comparing to shallow ones using the manually designed features.

In deep learning methods, local receptive fields grow in a layer by layer manner. The low-level layers extract fine features such as line, border, and corner while higher-level layers exhibit higher features such as object portions, like pedestrian parts, or the whole object, like car, traffic signs. It was presented that CNNs outperform recognition performances of conventional feature extraction methods.

Deep Convolutional Neural Network- CNN is kind of feed-forward neural networks consisting of multilayers.The layer building blocks of CNN are briefly explained below as:

Convolutional Layer- The layer performs the main workhouse operation of CNN. The input image is convoluted by using set of learnable filters or kernels. Each produces one feature map in output image. The feature are feed as input of second layer.

Rectified Linear Units- The layer includes units employing the rectifier. Given a neuron R input, x, the rectifier is defined as f(x)=max(0,x) in the neural networks literature. The rectifier function increases nonlinearity of the decision function.

Fully Connected Layer- The neurons of these layers are fully connected to all activations in the previous layer. The layer is accepted as final pooling layer feeding the feature to classifier. Matrix multiplication and a bias addition are used to calculate their activations.

Loss Layer- The trade-off between predicted and real class values is determined the defined L function in this layer. The layer is usually a final layer of the network. There are different loss functions for different applications.

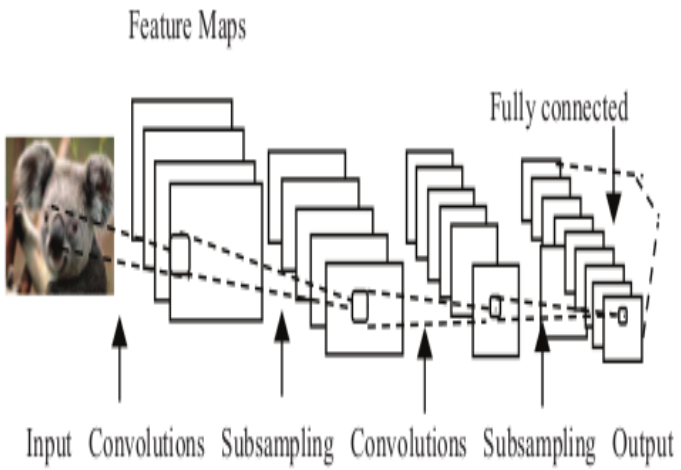


Figure 5: A typical Conventional Neural Networks architecture

Bag of Visual Words In object recognition applications, the BoW approach has provided several advantages such as fast run time, high recognition accuracy, and less computational load. In the BoW approach, the

image is considered as a text. The image features are detected, and feature keypoints are extracted. we used SURF and HOG to obtain feature keypoints and feature descriptors, respectively.

IMAGE -> Local Features (SURF and HOG) -> Feature Pool -> K-Means Clustering ->

Image Signature -> Classifiers-SVM -> Evaluation of Category Labels

The important steps of this algorithm are to divide into nine patches the image and to extract features relating to each patch. Feature extraction is carried out by using both different CNN architectures and BOW approach. Pre-trained AlexNet with nine layers and a large CNN architecture with ten layers are proposed for recognition and detection algorithm. In BOW, feature detectors and descriptors are obtained by SURF and HOG. The visual codes areobtained by using K-means. In the algorithm, the reduced features are fed to SVM classifiers. The outputs of SVM classifiers are fused by using majority voting rule.

3.1.5 A Model based Path Planning Algorithm for Self-driving Cars in Dynamic Environment

Self-driving cars require robust and fast path planning algorithms to operate in dynamic environment. In the last years, model based path planning has emerged as an effective solution to real-time path planning taking into account the kinematics of the vehicle. However, this approach only validates in simple environment such as static or single moving obstacle. a model based path planning algorithm can be used for dynamic environment. The approach generates candidate trajectories online, then evaluates and selects the most appropriate one according to real-time environment information. Thus, the planning algorithm is anytime and dynamic, and also effective in complex multi-obstacle environment.

A significant function required for self-driving cars performing missions is to plan reasonable trajectories offline or to gen erate trajectories

dynamically based on real-time environment information. Many different algorithms such as heuristic searching, artificial intelligence and model based algorithms have been developed to cope with path planning problems of self-driving cars. Artificial intelligence based algorithms including fuzzy logic algorithm, genetic algorithm, and neural networks have been successfully implemented in path planning, however, the optimal results or algorithm perfor mance are not necessarily guaranteed. Model based algorithms take vehicle kinematic constraints into consideration to guarantee smoothness and feasibility, but it mostly deals with static obstacles or single moving obstacle.

The algorithm consists of two parts: online trajectory generation and selection. The trajectory generator produces candidate trajectories based on current vehicle state, terminal vehicle state and vehicle kine matic constraints. Trajectory selector focuses on evaluating candidate trajectories and selecting the most appropriate one based on threat assessment of environment. This algorithm not only guarantees a safe, short and feasible trajectory, but also performs well in dynamic environment.

The proposed algorithm takes vehicle kinematics into account to guarantee smooth and feasible trajectories. Online trajectory generation and evalu ation methods are combined to obtain an optimal trajectory considering environment information. During the navigation on road, if any collision is detected, the self-driving car starts replanning to make sure the collision is avoided. The simulation results in both straight road and curve scenarios have shown the feasibility and high computational efficiency of the algorithm. Therefore, the algorithm is both optimal and practical, and improves the performance of path planning in dynamic environment.

3.2 Analysis

3.2.1 A survey on different types of Neural networks based algorithms

Table 1: Analysis of Different Algorithms

Paper	Technology	Algorithms	Advantages	Disadvantages	Applications
Name: Towards Neuroimaging real-time driving using Convolutional Neural Networks Year: 2016 Conference: 8th Computer Science and Electronic Engineering Conference (CEEC)	Covolutional Neural Networks for autonomous steering	Categorical image classification.	Deep neural network models have been proved ideal to process complex visual data, particularly in image classification	Processing or working with neural networks requires to much computation quickly requires use of GPGPU.	CNNs are indeed capable of performing well on visual data,and can be trained from visual example.
Name: Progress,in Neural Network-based Vision for Autonomous Robot Driving Publisher: IEEE Year: 2002	Convolutional Neural Networks	Training with structured noise and Gaussian Noise	Training with different types of noise lets CNN handle rare situations without any damage.		Training with different noises helps cars to avoid rare situation in which algorithm may take wrong decisions.

Name: Vision-based Detection and Classification of Pavement Mark using Neural Network for Autonomous Driving System Publisher: IEEE Year: 2011	Edge pairing to find a pavement mark	Edge pairing algorithm to classify different type of pavements using Neural Networks.	1.Pavements, detection helps cars to improve its decision process. 2. Different turning which cant be seen during run because of obstacle can be detected by pavements.	Ovehead increase during run of car, car has to do extra processing may affect its real time decisions	1. Pavemet detection helps to take right decisions by the car. 2. Cars during signals may stop at wrong place, pavement marks can make them stop at right place.
Name: Name: Moving Towards in Object Recognition with Deep Learning for Autonomous Driving Applications Publisher: IEEE Year: 2016	Convolutional Networks architectures, SVM classifiers	Bag of visual words algorithm.	Object detection and pedestrian detection help cars to avoid collisions with them.	Overhead increase due to extra processing a car might have to do to detect objects in its path.	With the detection of objects in its path the car might take appropriate decisions to avoid any damage and also avoiding collision paths.
Name: A Model based Path Planning Algorithm for Self-driving Cars in Dynamic Environment Publisher: IEEE Year: 2015	Model Based Path Planning	Online trajectory generation and Online trajectory selection.	1. Best path is selected to among many paths. 2. Collisions is avoided if any paths leads to collisions.	Sometimes undesired long path may be selected to detection of possible collisions in shorter paths but later it has gone.	1. Collision detection in path helps cars to take right decision and avoid such collisions. 2. Optimal path selection saves time of run.

4 Algorithms

4.1 Operation of feed forward multilayered operation

Artificial neural network is a massively parallel distributed processor made up of simple processing units, which has a natural propensity for storing experiential knowledge and making it available for use. Artificial neural Network refers to the modelling of the brain in two aspects which are functions and structure. Function in terms of the ability of a system to be able to perform the operations of human brain, such as prediction, optimization, controlling, association, thinking, reasoning and so on, structure is in terms of the layers of the neural. The models are composed of many computing elements, usually denoted by neurons; each neuron has a number of input and output. It has a matrix called memory or synaptic weights which connect the input neuron to the hidden neurons and output neurons. A linear combiner is used to produce a single value from all the inputs. The input to the neuron is obtained as the sum of the synaptic weight of the input. The input value is compared to the threshold value associated with the neuron to determine the activation signal of the neuron. This activation signal is then passed through an activation function to produce the output of the neuron. Sigmoid activation function was chosen for this research work due to its soft switching operation and also it helps artificial neural network to represent a more complex problem. A Back-propagation neural network is a multilayer feed-forward neural network, it considered as one of the simplest and most general method used for supervised training of multilayer neural network. It works in a way that it approximates the non- linear relationship that is between the input and the output. It approximates by adjusting the weight internally. It may also generalize for the input that is not present in the training pattern. The operation of back-propagation neural network is in two ways which are Feed-forward and back-propagation(feedback).

Feedforward Operation Feeding a specified training pattern into the input layer, the input weighted sum to the yth node in the hidden layer is given as :

$$T.Py = \sum WxyXy + \theta y, Eq1$$

The formula is used to calculate the summation of the input to the neuron. The ■y is the weighted value of bias node. The output value of the bias node is 1. This neuron is a false input to each of the neuron present in the hidden layer and output layer. The bias neuron is used to overcome the problem associated with situations where the values of an input pattern are zero because if any input pattern has zero value, the neural network will not be able to train without the bias node. In order to decide whether a neuron should be fire, the total potential(T.P) is passed on to an appropriate activation function. The result from this activation function determines the neurons output and becomes the input value for the neuron in the next layer connected to it. The typical activation used for this work is sigmoid function due to its soft switching operation.

$$Oy = Xz = 1/1 + e - T.P, Eq2$$

Equation 1,2determines the output value for the node z in the output layer

Backpropogation Operation Assuming that Oz is the actual activation value of the output neuron z and the desired output of neuron z is dz, the difference between the actual and desired output is given as :

$$\Delta z = dz - Oz$$

The error signal of the network z can be calculated in the output layer as :

$$\delta z = \Delta z Oz(1 - Oz)$$

$$\delta z = (dz - Oz)Oz(1 - Oz)$$

Oz (1- Oz) is the derivation of a sigmoid function. The back propagation looks for the minimum value of error function in the weight space using technique known as delta rule. The change in the synaptic weights that connect the input neurons y and output neurons z, using delta, is proportional to the error at neuron z multiplied by the activation neuron y The equation below is used to modify the synaptic weight Wy,z between the output neuron z and neuron y :

$$\Delta Wy,z = \eta \delta z Xz$$

$$Wy,z = Wy,z + \Delta Wy,z$$

4.2 Algorithm for vision based pavement marks detection and classification

The algorithm has three assumptions.

1. The first assumption is that we already know equations of the lanes of a driving road in the road image. We model the lanes as two linear equations. The algorithm finds a candidate region of pavement marks between the lanes in the road image.

2. The second assumption is that we ignore marks which extend beyond the top or bottom search region of the image, so we only find entire marks in the search region.

3. The third assumption is that the mark is white and the pavement is black.

We first convert the input color images of road to gray images. For each row between the top and bottom search region, we calculate the horizontal edge value of each pixel between two lanes using a mask, i.e. Then we filter each edge value between the threshold values, 300 and -300, and set the edge value to 0. The threshold values were obtained manually. We make groups of non-zero values in the filtered edge values using zero values as boundaries. Then we store the maximum value in the group of positive values as a local maximum value, and the minimum value in the group of negative values as a local minimum value. By the third assumption, the edge value of the pixel on the left side of the mark is a large positive value, and the edge value of the pixel in right side of mark is large negative value. From left lane to right lane, we find a local maximum value and a local minimum value which is on the right side of the local maximum value in the filtered edge values. More than one maximum-minimum pair can exist in one row. We store the pixels between the maximum and minimum as a candidate mark (Fig. d). The algorithm applies this process on each row, collects results, and makes a binary image indicating a candidate mark.

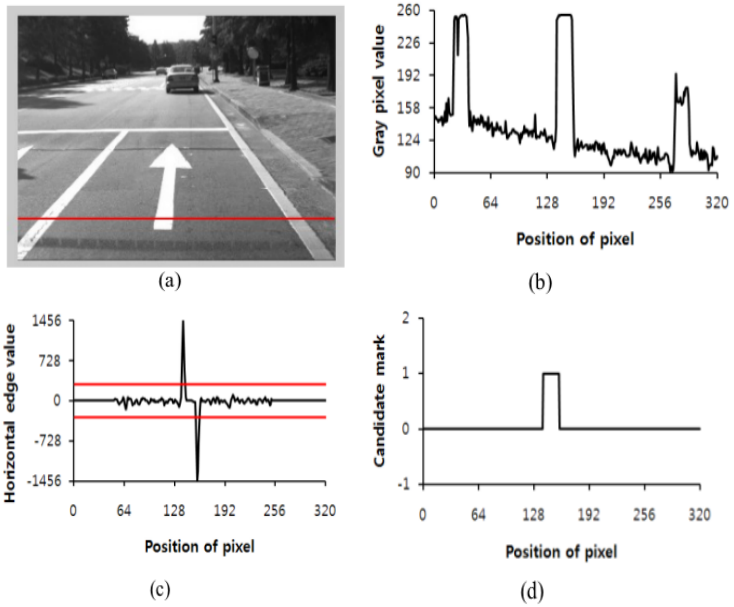


Figure 6: (a) Gray image. Red line indicates the example row (b) Gray image pixel value of the example row (c) Horizontal edge value of pixels in the example row. Two red lines indicate the threshold values, 300 and -300. (d) Candidate mark obtained from (c).

After making a binary image of the candidate mark, the algorithm calculates the horizontal sum of the image to add the rows of the image simply. Then the algorithm finds the top and bottom range of mark by finding the largest non-zero group in the horizontal sum. After getting top and bottom range of mark, the algorithm calculates vertical sum of the image in the top and bottom range. However, to extract an identical feature regardless of a rotation of the marking, the algorithm makes the vertical histogram to add the direction parallel to lanes in the real world. Then the algorithm finds the left and right range of the mark by finding the largest non-zero group in the vertical sum. The algorithm recalculates the horizontal sum of the image in the left and right range of mark. The algorithm calculates the horizontal sum twice to minimize the effect of image components which are not marks.

We divide each sum in the range of the mark into 30 zones, and extract 30-dimensional vectors from the maximum value of each zone. Using this procedure, the feature vectors can contain better the narrow characteristics of the sum in comparison to down sampling. The algorithm makes scaled vectors by scaling the size of the maximum of the 30-dimensional vectors to 30. The algorithm makes a 60-dimensional feature vector by concatenating these 30-dimensional vectors from horizontal sum and vertical sum. The algorithm uses the 60-dimensional feature vectors as input to the neural network that identifies marks.

4.3 Bag of Visual words (BoW) approach for Object Recognition

In object recognition applications, the BoW approach has provided several advantages such as fast run time, high recognition accuracy, and less computational load. In the BoW approach, the image is considered as a text. Hence 'words' defining the text are determined. The steps of approach are performed as follows. The image features are detected, and feature keypoints are extracted. In this step, a feature detector can be used

or can be defined a grid to extract feature descriptors. Then, it is obtained a visual word dictionary called as codebook by reducing the number of features through quantization of feature space using a clustering algorithm. Fig. 7 gives the block schema of BOW approach. There are a few descriptor methods. In this paper, we used SURF and HOG to obtain feature keypoints and feature descriptors, respectively. We used multiclass linear SVM classifier to recognize the encoded features from each image category.

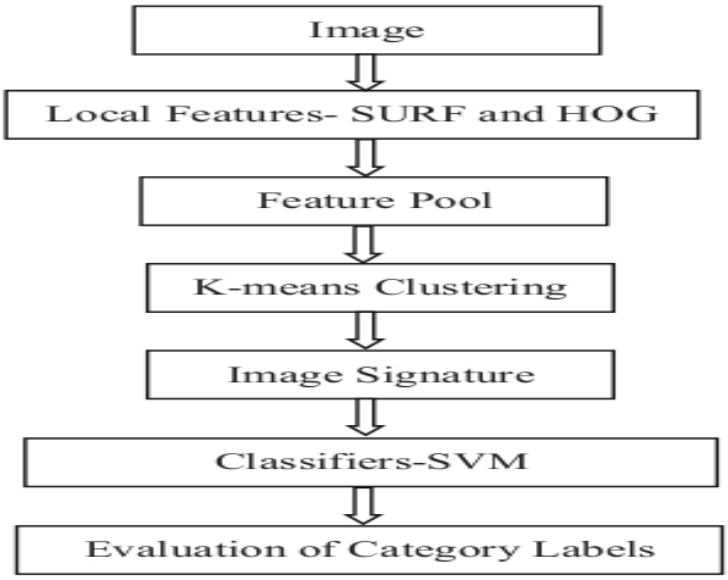


Figure 7: Schematic of Bag of Word algorithm

The block schema of the proposed CNN detection and recognition algorithm is given in Fig. 8. The algorithm is applied at eight folds:

(1) Divide to nine patches all images.,

(2) Up/down resize to 64x64x3 each patch and convert to grey image,

(3) Construct a CNN with ten layers consisting of input, convolutional, max pooling, convolutional, average pooling, convolutional, convolutional, and softmax regression,

(4) Apply stochastic gradient descent to the proposed CNN for applying to each image patch.

(5) Extract the features from full convolution layer,

(6) Determine distinctive feature sets by applying Principal Component Analysis (PCA)

(7) Use linear multi class SVM classifier for training the features,

(8) Perform the decision fusion to the outputs of nine SVM classifiers.

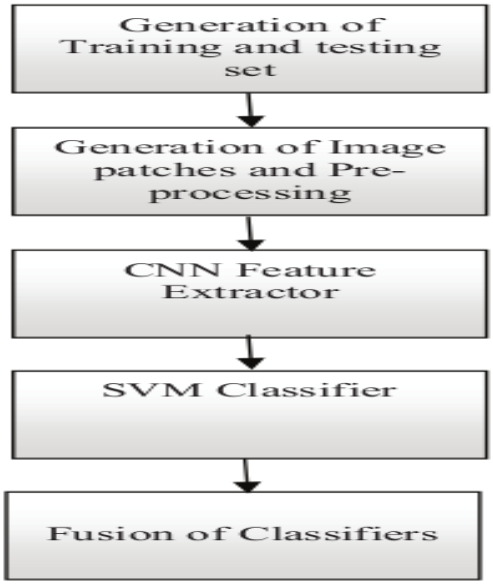


Figure 8: The block schema of the proposed algorithm.

5 Conclusion

Thus, Convolutional neural networks is the best method which can used to let a car drive autonomously using visual input. Development of GPGPUs have made it possible to use them in current scenerios. CNN learn by taking examples, different techniques such as training with gaussian noise or training with structured noise can be used to let cars handle very rare situations. Adding structured noise to the training patterns using a model describing the charaaeristics of irrelevant features siecantly improves driving performance. Pavement detection methods, can be used in parrallel to make cars improve its decisions after recognizing the pavements symbols made on the roads. It also help cars to stop at right position on signals. Object detection and padestrain detection plays a crucial role in detection of obstacles. Bag of Visual Words is such kind of algorithm which divides the image in different patches and analyze them with different operations to finally classify object. Model based path planning can also be used to avoid, autonomous cars to take long and undesired paths, during their run. During the navigation on road, if any collision is detected, the self-driving car starts replanning to make sure the collision is avoided.

6 Bibliography

References

[1] Carlos Fernandez Musoles, “Towards Neuroimaging real-time driving using Convolutional Neural Networks”, IEEE 8th Computer Science and Electronic Engineering Conference (CEEC), 2016

[2] Dean A. Pomerleau, “Progress in Neural Network-based Vision for Autonomous Robot Driving”, IEEE Intelligent Vehicles ’92 Symposium, 1992

[3] Yu-Bin Yoon and Se-Young Oh, “Vision-based detection and classification of pavement mark using neural network for autonomous driving system”, IEEE 8th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), 2011

[4] Ayşegï¿œl Uï¿œar, Yakup Demir and Cï¿œneyt Gï¿œzeliş, “Moving towards in object recognition with deep learning for autonomous driving applications”, IEEE International Symposium on innvations in intelligent system and applications (INISTA), 2016

[5] Chaocheng Li, Jun Wang and Xiaonian Wang, “A model based path planning algorithm for self-driving cars in dynamic environment”, IEEE Chinese Automation Congress (CAC), 2015

[6] Tzuu-Hseng S. Li, Chih-Yang Chen and Kai-Chuin Lim, “Combination of fuzzy logic control and back propagation neural networks for the autonomous driving control of car-like mobile robot systems”, IEEE SICE Annual Conference, 2010.

[7] Zejia Zheng, Juyang Weng, “Mobile Device Based Outdoor Navigation with On-Line Learning Neural Network: A Comparison with Convolutional Neural Network”, IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2016

[8] Chaomin Luo, Simon X. Yanga and Mohan Krishnan, ”An effective vector-driven biologically-motivated neural network algorithm to real-time autonomous robot navigation”, IEEE International Conference on Robotics and Automation (ICRA), 2014