# Neural Computing Semester 2

## Introduction

For this second semester I have been asked to produce three different neural networks the first of these neural networks was a Probabilistic Neural Network, PNN, the second was a self-driving car based on the fuzzy logic neural network and the third was a genetic algorithm.

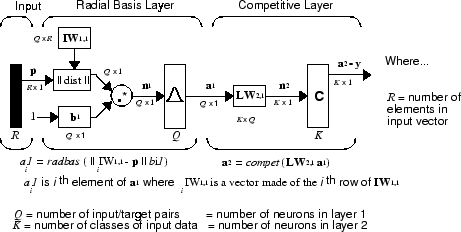
## Literature Review

This is a brief literature review of the findings that I made whilst doing some research on parts A, B and C. this review will be broken down into 3 sections as the later stages of this report have been.

Part A

Probabilistic neural networks can be used for classification problems. When an input is presented, the first layer computes distances from the input vector to the training input vectors and produces a vector whose elements indicate how close the input is to a training input. The second layer sums these contributions for each class of inputs to produce as its net output a vector of probabilities. Finally, a compete transfer function on the output of the second layer picks the maximum of these probabilities and produces a 1 for that class and a 0 for the other classes. The architecture for this system is shown below.

### Network Architecture



It is assumed that there are Q input vector/target vector pairs. Each target vector has K elements. One of these elements is 1 and the rest are 0. Thus, each input vector is associated with one of K classes.

The first-layer input weights, IW1,1 (net.IW{1,1}), are set to the transpose of the matrix formed from the Q training pairs, **P**'. When an input is presented, the || dist || box produces a vector whose elements indicate how close the input is to the vectors of the training set. These elements are multiplied, element by element, by the bias and sent to the radbas transfer function. An input vector close to a training vector is represented by a number close to 1 in the output vector **a**1. If an input is close to several training vectors of a single class, it is represented by several elements of **a**1 that are close to 1.

The second-layer weights, LW1,2 (net.LW{2,1}), are set to the matrix **T** of target vectors. Each vector has a 1 only in the row associated with that particular class of input, and 0s elsewhere. The multiplication **Ta**1 sums the elements of **a**1 due to each of the K input classes. Finally, the second-layer transfer function, compet, produces a 1 corresponding to the largest element of **n**2, and 0s elsewhere. Thus, the network classifies the input vector into a specific K class because that class has the maximum probability of being correct.

Part B

In order to start building a neural network that can handle autonomous driving I need to define what autonomous driving is and as in an article be BMW AG (2019) The Path to Autonomous Driving they stated that there are 5 levels to autonomous driving the first three are mostly driver assistance and this is not full autonomy however level four and five. Level four of the article states that at Level 4 they considered this to be fully autonomous driving, although a human driver can still request control, and the car still has a cockpit. In level 4, the car can handle the majority of driving situations independently. The technology in level 4 is developed to the point that a car can handle highly complex urban driving situations, such as the sudden appearance of construction sites, without any driver intervention.

The driver, however, must remain fit to drive and capable of taking over control if needed, yet the driver would be able to sleep temporarily. If the driver ignores a warning alarm, the car has the authority to move into safe conditions, for example by pulling over. While level 4 still requires the presence of a driver, cars won’t need drivers at all in the next, final level of autonomous driving.

These statements are well above the threshold for automated driving and implementing something like this in MATLAB would be possible with fuzzy logic.

Level five of this article states they believe that unlike levels 3 and 4, the “Full Automation” of level 5 is where true autonomous driving becomes a reality: Drivers don’t need to be fit to drive and don’t even need to have a license. The car performs any and all driving tasks – there isn’t even a cockpit. Therefore, every person in the car becomes a passenger, opening up new mobility possibilities for people with disabilities, for example.

Cars at this level will clearly need to meet stringent safety demands and will only drive at relatively low speeds within populated areas. They are also able to drive on highways but initially, they will only be used in defined areas of city centres.

While this is full automation this sort of fuzzy logic network would take an extensive amount of time to implement and would be well out of the scope of such a small section of a semesters worth of assignment.

Part C

## Methods

### Part A

The PNN Network is used to automate the classification of data from a large data set the data that this network will be classifying will have many different data types in it from integer-based data through to text. The first step in creating this neural network was to look at the data and prepare it for input into the network as this would have a large impact on the initial building steps.

### Part B

The main aim of this section of work was to make a self-driving car in Matlab that could take the user anywhere they wanted to go with and address input on the GPS. During the design of this network the first thing that I did was to set some limits in order to keep this section within the scope of the assignment. These parameters have helped to stop this section getting over complicated. The first of these was the cars ability to speed up or slow down based upon the speed limit. Initially this was a difficult idea to process into something the computer could handle. However given that a self-driving car of this type would have a satellite navigation system this car would have access to the current speed limit to the road that it was driving on therefore the rules that needed to be imposed upon the car would be that if the car was not that the current speed limit then the car would have to accelerate to that speed limit if however the car was at the speed limit already then it would not have to accelerate just hold its current speed and if the car was faster than the current speed limit then it would have to slow down as it would be speeding. The next step for this section was to design the way that the car would know how to stay on the road and stay on the correct side of the road. Due to roads having a solid white line on the edges and different types of lines in the centre the program would have to know what type of central line it is and where that line is in relation to the sides of the car and by staying in these lines the car will be able to turn corners with the aid of the GPS data that it has. Following these basic driving rules there are some override rules that take place in specific events that require special attention to other road users such as when an emergency stop needs to be performed and when a lane change on a dual carriageway or motorway needs to be performed. Furthermore if the conditions of the road are such that it is unsafe to drive altogether or that there is a higher risk than usual then the car will either refuse to drive if the conditions are such that the risk is too high or it will drive slower than it normally would in order to reduce the risk to the passengers. Finally if there are other road users then the car will maintain a safe distance from them to avoid crashes.

### Part C

The aim of part C was to from the minimum value of f for the value of x between 0 and 30 and to do this where f is defined as f= -(10+(-x^2/10) +3\*x) +50 +5\*sin(2\*x). The first thing to do was to learn how genetic algorithms were implemented in MATLAB and following this implement the algorithm that I needed.

The first piece of code that I wrote was the simple fitness function that is described above

## Conclusions

Throughout this assignment I found this coursework challenging and was continually pushed as Matlab isn’t one of the pieces of software that I am most comfortable with or have a good knowledge base in.

## References

BMW AG (2019). The Path to autonomous driving. <https://www.bmw.com/en/automotive-life/autonomous-driving.html> Accessed 21/04/18

MathWorks (2019). *Probabilistic Neural Networks* <https://uk.mathworks.com/help/deeplearning/ug/probabilistic-neural-networks.html;jsessionid=733f7f7a2cac8fe26a94c4e7363e> Accessed 08/05/19