IB Computer Science Paper 3 Case Study 2018 Preparation

Compiled by Lucas Gurney

These notes are based on the 2018 Paper 3 Case Study for the IB Computer Science course on autonomous driving. They were compiled using various (unfortunately) uncredited sources—I wasn't worrying about citing my sources when I did this! I believe that these notes are sufficient preparation for the exam, however I am not a teacher, and it is hard to determine exactly what the IB is looking for. However, I did obtain a Level 7 is the May 2018 season using these notes exclusively for Paper 3, so I should hope they are at least somewhat suitable.

Please be aware that as these were my personal notes, there may be mistakes as no one else has checked them. I strongly encourage you to carry out your own research, using this resource only to aid you in that process. You must understand all of the key concepts, and the only way you'll do that is by exploring them on your own. I would in fact advise you to produce your own document, perhaps using the same format as this one, but with your own words and diagrams. Ultimately though, revision is a personal matter and all I can do is help you by providing you with this tool, it is up to you how you use it.

Good luck with your exams!

Contents

Autonomous cars – An introduction

What are autonomous cars?

An autonomous vehicle is able to navigate and manoeuvre itself without any need for human control or intervention.

State of current research

The Society of Automotive Engineers is a profession association that helps develops standards for engineering professionals. One such standartd that they have created is the 'levels of autonomy'.

Levels of autonomy classify to what extent a vehicle is truly autonomous and how much it can accomplish on its own without human intervention. It is measured from level 0 (no automation) through to level 5 (full automation)

Level	Description	Control of vehicle	Monitoring of environment	Fallback control	Conditions
Level 0	No automation	Human	Human	Human	N/A
Level 1	Drive assistance	Human and System	Human	Human	Some
Level 2	Partial automation	System	Human	Human	Some
Level 3	Conditional automation	System	System	Human	Some
Level 4	High automation	System	System	System	Many
Level 5	Full automation	System	System	System	All

In the *Levangerstadt Project*, the vehicles would aim to be at level 5. The key distinguishing factor of this level is that the system must be able to drive under any conditions a human would be able to

Right now, different cars are at different levels, but increasingly modern cars are moving higher and higher up the levels.

Some notes about the current state of self-driving car development are as follows:

- Tesla cars are now all shipped with all of the hardware needed for fully autonomous driving
- Waymo (Google) autonomous cars have covered over 4 million miles in self-driving, with 25,000 autonomous miles covered a week, mostly on complex city streets. In addition, they drove 1 billion simulated miles just in 2016
- Waymo will star the first public trial of their cars in Phoenix, Arizona
- The Roborace program aims to be the first driverless electric racing car
- The Uber business model relies on autonomous cars to become profitable. They have started piloting a program where in certain areas you can get matched with a self-driving Uber. They aim to have these in general circulation within 18 months
- Nissan and DeNA have just launched their own self-driving taxi service
- NVIDIA DRIVE is an AI platform aiming to give manufacturers the ability to integrate self-driving into their vehicles
- Most current projects involve a tech company providing the architecture whilst an auto company provides the vehicle

- The Baidu Apollo self-driving system has been offered to anyone who wants to use it- an attempt to speed up the development of its system (similar to how Android worked)
- A lot of cars are tested in California, where it is possible to get a license to test self-driving cars
- Drive.ai is a startup which through software innovation with its neural nets has been able to establish itself as a major player in the industry

Advantages of autonomous cars

Autonomous vehicles have a number of potential advantages that could be realised, including:

- Increased **safety**
  - Reduction in traffic collisions (and hence a reduction in the consequences of these, including injuries, related costs, and insurance costs)
- Increased **mobility**
  - Improved traffic flow reducing journey times
  - Gives more mobility to those who would not otherwise be able to drive, such as children and the elderly
- Reduced **costs of mobility and infrastructure**
  - Potential for car sharing to bring down costs and pollution impact
  - Reduced need for parking spaces
  - More efficient travel as autonomous vehicles are more 'perfect' at driving
- Reduced **crime**
  - All traffic laws would be obeyed by the autonomous vehicles, making roads safer for everyone
- Increased **customer satisfaction**
  - Would relieve drivers of the burden/chore of driving
  - 'Drivers' would be able to focus on something else whilst travelling, which may be more enjoyable or productive

Current limitations of autonomous cars

- **Technological challenges**- the technology simply isn't there yet
- Disputes concerning who is **liable** for the actions of an autonomous vehicle
- The time and cost required to **replace** the existing stock of vehicles on the road
- Concerns with the **safety and reliability** of an autonomous car system
- Lack of **legal frameworks** on how to address cases that concern autonomous vehicles
- Concerns about the potential **loss of jobs** and how governments will address this
- **Security** of the car's systems and how susceptible they would be to hacking
- **Difficulty of city driving**, when compared to the highway-based driving that already exists

Keys to full autonomy

Knowing the car's exact location

Global Positioning System (GPS)

GPS = a network of satellites that allow people on the Earth to pinpoint their location from anywhere on the planet

- GPS is a network of about 30 satellites developed and owned by the US military, but now available for anyone with a GPS device to use
- Each GPS satellite transmits information about its position and the current time at regular intervals. A GPS receiver can then be used to calculate how far away each satellite is based on the time taken for the message to arrive.
- Trilateration can be used once there are at least three satellite signals being received. The more signals you receive though, the more accurate the process will be. Four satellites allows you to get vertical height. The orbits have been set up so you should always be in range of at least six satellites
- The GPS satellites use atomic clocks which are highly accurate- but they still need to have allowances made within them for relativistic effects.

Wide Area Augmentation System(WAAS)

WAAS = an aid used to augment the GPS by using ground stations to provide ground-based reference stations to measure small variations in the GPS signals, and then send corrections to WAAS satellites for broadcasting

High definition mapping

HD mapping = high precision maps at centimetre-level accuracy

- Mapping is a crucial tool for helping AVs make their decisions, so it is important that they are as accurate as possible
- HD maps are built for self-driving purposes and so are made to be as accurate as possible, containing all the information possible about the road that the car is on: including lanes, road boundaries, curb size, etc.
- They are made by sensors on vehicles mapping the road out. This also means that AVs can work on and improve the maps as they are driving
- The maps need to be kept under constant maintenance to maintain integrity, and they are therefore available over cloud infrastructure

Perception of the car's immediate environment

Sensor Fusion

Sensor fusion = taking the inputs of different sensors and sensor types in order to be able to perceive the environment more accurately

- In order to give the vehicle the information it needs in order to be autonomous, a huge amount of data is needed from sensors
- Different sensors have different shortcomings with them. These shortcomings cannot be overcome by simply adding more sensors of the same type (for example, adding more radar

units will not overcome the fact it doesn't have very high resolution). In order to make up for these shortcomings, data from multiple sensors must be combined.
- Allows the car to make better and safer decisions
- Can add redundancy to situations where one type of sensor may not be of use (for example, cameras will be no good in low-visibility environments)
- Sensor fusion systems can be centralised (where all processing and decision making is done in a single location) or distributed (where sensors carry out their own processing and local decision making but the final decision is done by a central system.

LIDAR

LIDAR = using pulsed laser light in order to measure distances and generate 3D representations of environments.

- LIDAR enables cars to have continual 360-degree visibility as well as accurate depth information up to ± 2cm
- LIDAR systems continually fire off beams of laser light and then measure how long it takes for this light to be returned to the sensor. By repeating this process millions of times a second, the sensor can build up a 3D image of the surrounding environment
- The biggest use of LIDAR is 3D mapping in order to aid navigation. Recent developments have made LIDAR more accurate so that it can now also be used for object detection and tracking
- The future of LIDAR depends on further research being able to bring the cost of it down significantly (solid state LIDAR has potential for this) and resolution and range improvements

Point clouds

Point clouds = a set of data points in a coordinate system used to represent the external surface of an object

- Point clouds are often obtained from LIDAR information
- They can be used to build up the image of the local surroundings

Bounding boxes

Bounding box = an imaginary box that surrounds an object that is being checked for collision, in order to represent the space which it occupies

- Bounding boxes can be constructed from the LIDAR data
- The box must fully enclose an object in order to be as accurate as possible
- The box can also help simplify the processing of complex objects. By using a simple shape such as a cuboid to surround a complex shape, like a person, the whole thing does not need to be modelled

Making correct driving decisions

Machine learning = getting machines to learn how to accomplish certain tasks without being explicitly programmed to do so

> Supervised learning = giving the learning system labelled data so it is able to check to see if its predictions are right or wrong. It focuses of mapping a specific input to a specific output based on example input-output pairs

Unsupervised learning = using unlabelled data and hence attempting to infer some kind of hidden structure in the input data

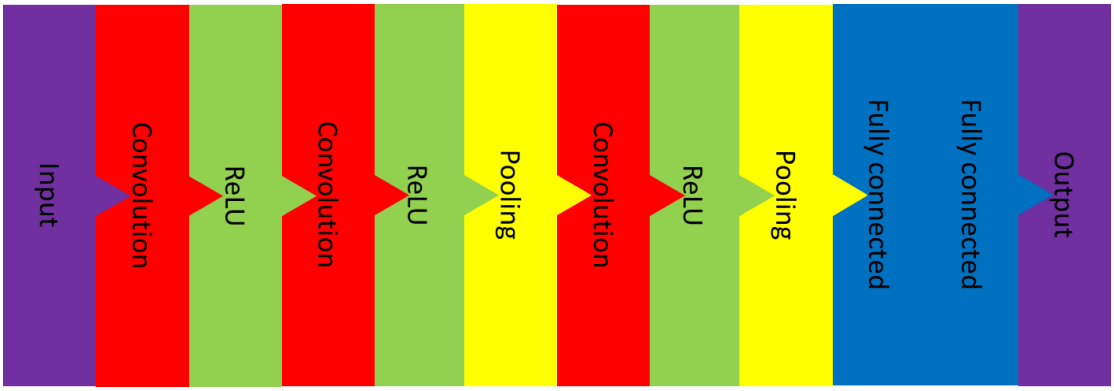
Deep learning = a technique for implementing machine learning by using feature learning-- getting systems to automatically abstract inputs into smaller features, creating a nested hierarchy of increasingly abstract concepts

Convolutional Neural Networks

CNN = a type of deep, feed-forward artificial neural network that has be applied in particular to analysing visual imagery. They can be used to allow machines to learn to recognise patterns in images.

CNNs are made up of four types of interconnected layer: convolution, pooling, rectified linear unit, and fully connected layers. The final network is made by layering these on top of each other to create some final topology.

Hyperparameter = a parameter set before any of the automated learning takes place, so that the workings and performance of the algorithm can be tweaked as required



The layers can be linked together because they all share similar inputs and outputs

Feedforward = the process of moving from some given input to an output by moving the data through all of the layers.

Training = the process of a machine improving its own performance at a specific task. In a neural network, it does this by using gradient descent to minimise a cost function.

Cost function = a function that gives the amount of error associated with an output from a neural network. The process of training seeks to minimise this value by using an optimisation algorithm

Gradient descent = an algorithm to find the local minimum of a function. It is one such algorithm that can be used to minimise a cost function. It aims to calculate the gradient at a certain point and then move downwards

Backpropagation = an algorithm used by gradient descent in order to calculate the gradient of the cost function with respect to the model's parameters. Error can be propagated backwards through the network and weights adjusted

This is a two-phase process. Forward propagation will produces an output, which is compared to the correct output using a cost function. This gives the amount of error in the final answer. The error is then propagated backwards through the network until each neuron has an associated error that reflects its contribution to the original output.

The individual errors as used to calculate the gradient of the cost function. A ratio (corresponding to the learning rate) of the weight's gradient is then subtracted from the weight, in an attempt to minimise the cost function next time. This process is repeated until a desirable level of error is reached

Model = a means of fitting a pattern to a set of data so that the data is easily summarised

Overfitting = where a model matches a particular set of data too closely, and so fails to generalise to new data well

Underfitting = where a model doesn't really match or describe a dataset at all, and so doesn't really do a good job at representing it

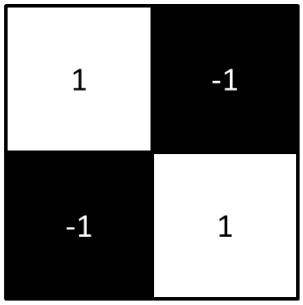
Convolution

Purpose of this layer: To find and locate the presence of smaller patterns in a larger image

Most of the heavy computation of the network is done in this layer

Convolution = the process of searching for patterns within a larger image

Hyperparameters include the number of features and the dimensions of each feature, as well as things like the filter stride

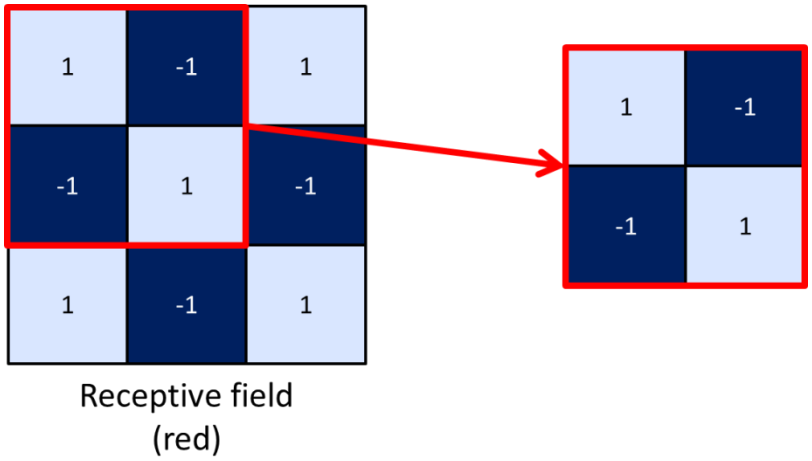


An example of a feature, a diagonal line similar to a forwards slash. The numbers are used for filtering

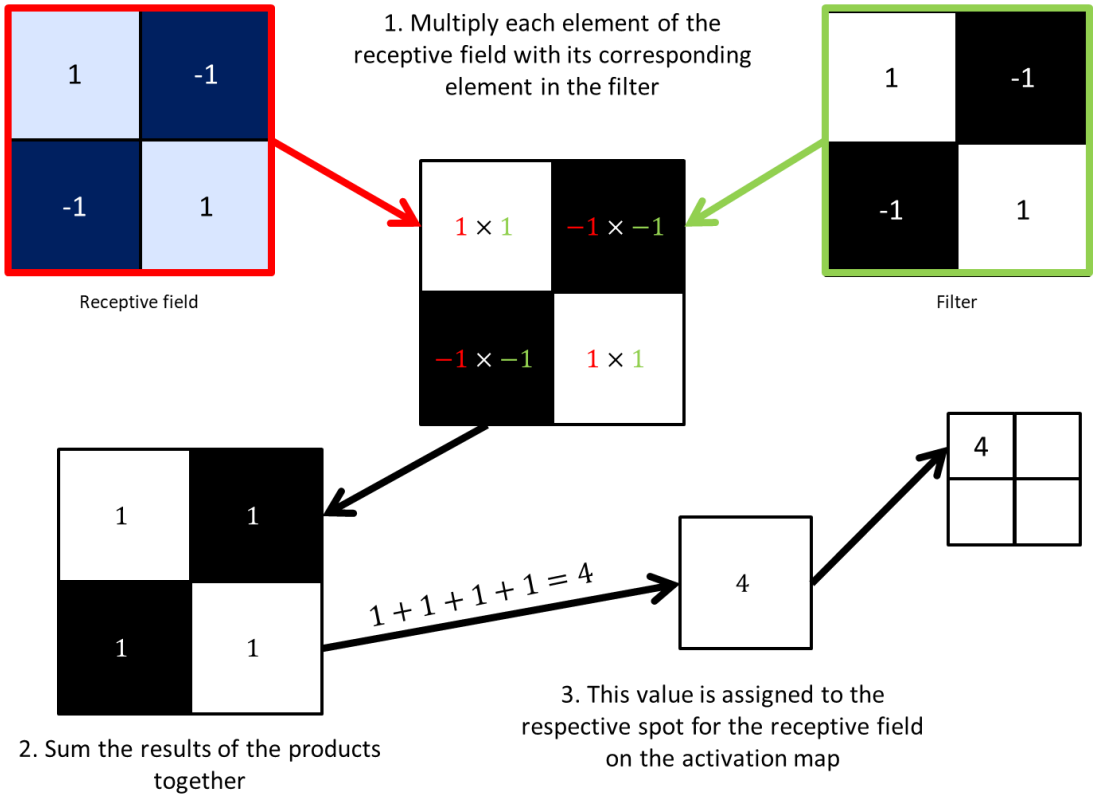
Feature = a specific piece, or smaller pattern, within a larger image that a CNN searches for in convolution.

Filter = a map of weights that are used to describe what a feature looks like

In convolution, the filter is slid around the input image. Convolution is repeated for as many types of filter as needed, with a different feature map being created for each one



Receptive field = the region of the larger image currently being analysed by the filter

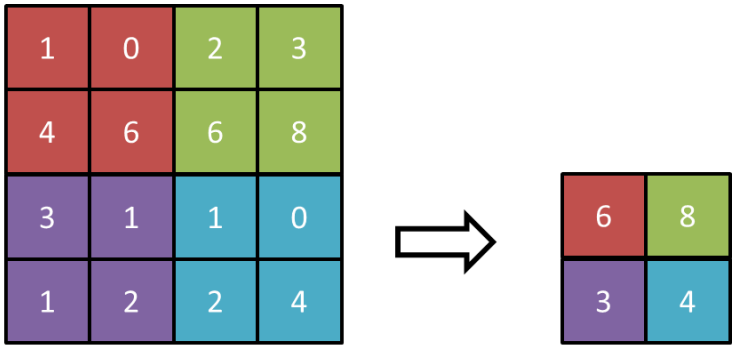


Activation map = a map that is formed from the result of each iteration of the convolution process as an output. It corresponds to the sums of each of the element wise product of each receptive field test (sort of a dot product)

Pooling

Purpose of this layer: To reduce the size of the input image without losing information, allowing for faster computation of the neural network

Pooling = reducing the size of an input whilst preserving the most important information about it. For a raw image, this is done by shrinking large images down to reduce the size of the image being processed without losing any information.



Input image (divided into windows)

Window = a subset of a larger image, comprised of multiple values which are combined into one single value in the process of pooling

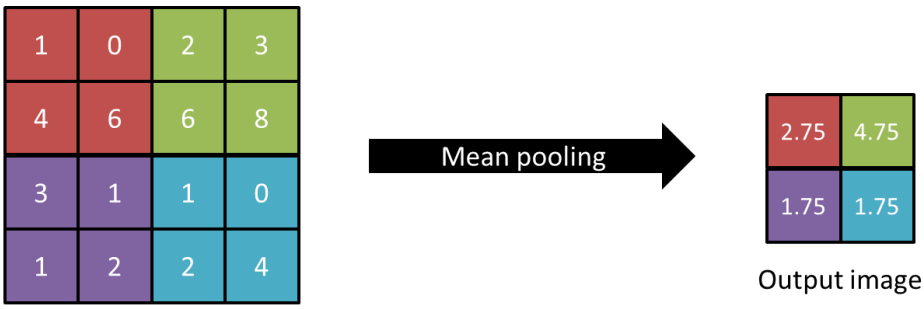
Stride distance = the spacing between each window on the input space

The window size and stride distance are two hyperparameters that can be set. Generally, these are made to be non-overlapping.

Input image

Output image

Max-pooling = a method of pooling where the resultant value is the maximum of the initial values inside the window



Input image

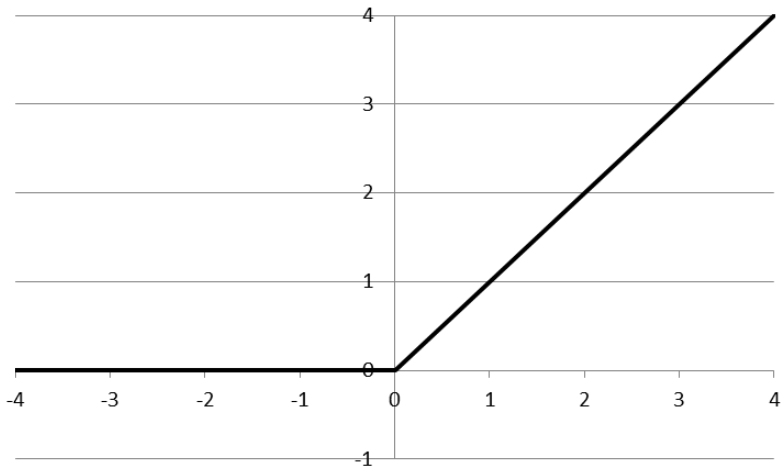
Output image

Mean-pooling = a method of pooling where the resultant value is the mean of the initial values inside the window. This method is no longer really used as max pooling has been shown to be better in practice

Rectified Linear Units

Purpose of this layer: To map an input signal to an output one by compressing the input into a range, removing unnecessary information and adding non-linearity

Activation function = a function that constrains its input to a certain range in some way. They are used for determining if the neurons in the CNN should fire (i.e. be activated) or not and to add non-linearity into models produced



$$a(x) = \max(0, x)$$

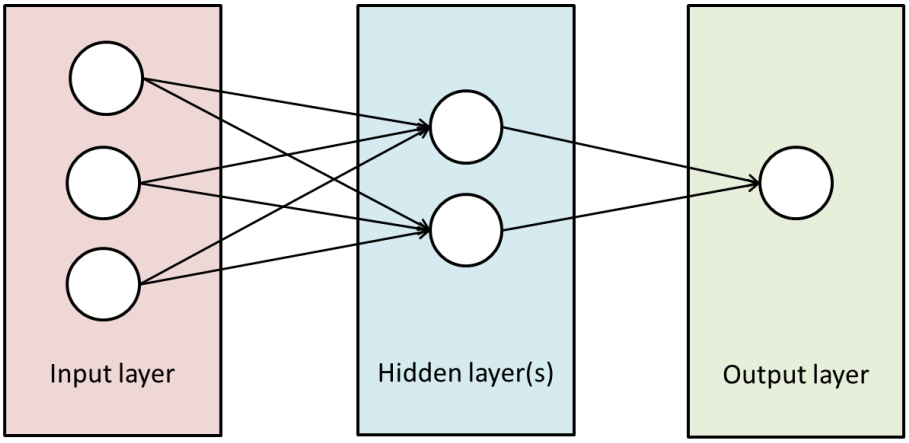
ReLU = A type of activation function that removes negative values by replacing them with zeroes. The activation function is applied element-wise to each part of the image in the CNN.

> Other activation functions are available (for example the sigmoidal logistic function), however ReLU is much faster to compute and so with the large scale of CNNs, they can be very useful in reducing computation time. Another major advantage is that they do not suffer from the vanishing gradient problem of, say, the sigmoid function. In that function, gradient approaches zero as x approaches infinity, but this does not happen with ReLU.
>
> A problem with ReLU is that it can lead to negative inputs dying as they are just set to zero. To deal with this, leaky ReLUs are sometimes used where a slight gradient is also applied to the negative values in order to keep them alive

Fully connected

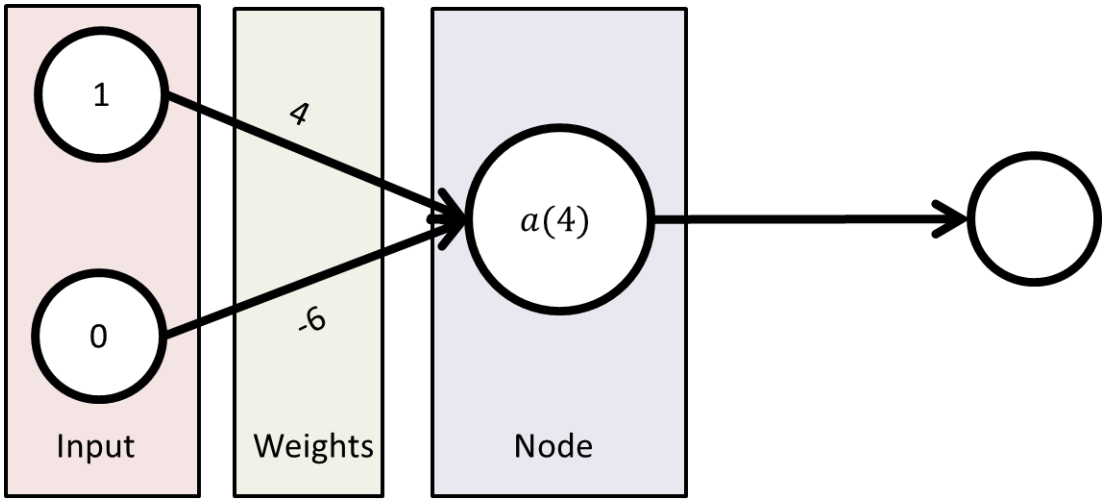
Purpose of this layer: enable the CNN to make high-level reasoning and produce a final judgment by using a multilayer perceptron based model

Multilayer perceptron = a 'normal' neural network where a set of inputs are taken in before being propagated through a series of nodes and hidden layers in order to produce a final output activation. It gets its fully connected name from the fact that every neuron in one layer connects to every neuron in another layer.



Each circle is a **node** and they are connected together by **weighted synapses**

Hyperparameters for this layer generally revolve around the choice of network topology used, and there are many ways in which that can be customised (number of layers, number of nodes, etc.)



1. Multiply each input value by its corresponding weight

$$1 \times 4 = 4$$
$$0 \times -6 = 0$$

2. Sum these values together

$$4 + 0 = 4$$

3. Apply activation function to the summation to get output

$$\text{output} = a(4)$$

Each perceptron takes a number of input values from nodes in the previous layer. These input values are then multiplied by their respective weights that have been learnt in the learning process. Then, the weighted inputs are summed together. An activation function is then applied to this summation in order to get an output

In fully connected layers, generally a logistic function is used in order to offer a probability of the input meeting some given criteria.

Navigation algorithms

Shortest path problem

Shortest path problem = the problem of finding the shortest path between two nodes on a graph. This is analogous to finding the shortest path between two points on the map

There are a number of different algorithms that exist for solving this problem. Some of the most popular ones include: Dijkstra's algorithm and the A* search algorithm

Dijkstra's algorithm

Dijkstra's algorithm = an algorithm for finding the shortest paths between nodes of a graph

Dijkstra's algorithm is being actively considered for use in Levangerstadt in the taxi navigation system.

Dijkstra's original 1959 algorithm is $O(n^2)$, where n is the number of nodes being investigated. This means that the algorithm runs in deterministic polynomial time (i.e. is in complexity class P).

The algorithm is as follows:

1. Start with a graph and identify the start and end nodes on the graph. Also, establish a priority queue to store 'tentative distances' to each of the nodes, where the nodes are sorted by this distance. Set the tentative distance to the start node to be 0, and the distance to every other node infinity (as it is presently unknown how long it will take to get there). The priority queue also needs to store the previous node visited to get to that node
2. From the start node, look at a the connecting node and calculate the sum of the distance to arrive at the current node, plus the weight of the connecting node. If this is shorter than the connecting node's tentative distance, replace the tentative distance and previous node of the connecting node.
3. Repeat the above step for all other connecting nodes.
4. Repeat the above two steps for the next shortest tentative distance node, until the destination node has been reached
5. Work backwards from the destination node, going through each visited node until the path that was travelled is identified.

Advantages of Dijkstra's algorithm

- Prioritises checking short paths
- It can reveal the shortest path to all nodes on the graph, not just the target one
- Time complexity is actually relatively low so can be used on larger problems

Disadvantages of Dijkstra's algorithm

- Is a form of blind search, so wastes a lot of resources
- Fails with negative weights, although with this application this shouldn't be a problem

Travelling salesman problem

Travelling salesman problem = a problem that asks for the shortest possible path connecting a number of nodes together so that each one is visited exactly once

> Will be used in the bus system for finding the shortest possible route to visit all of the required locations in the shortest time possible.

Complexity of problem

TSP is NP-hard. This means that a solution is not guaranteed to be found within polynomial time, although a solution can be verified in this time. Because of this, the options for solving the problem include devising a fairly inefficient exact algorithm or devising heuristic algorithms (ones that work well enough)

Because there are a fairly low number of seats on the bus, an exact algorithm can probably be used successfully.

Possible solutions

- An exact algorithm that can be used is a **brute force search** where every permutation is calculated and then the shortest one is identified. This executes in factorial time.
- The problem has been solved in exponential time using the Held-Karp algorithm. This was one of the first applications of dynamic programming (where a larger problem is broken down into several smaller ones)
- The nearest neighbour algorithm is a heuristic approach that chooses the nearest unvisited node as the next move. It is a greedy algorithm that quickly yields an effective route, however this is not always the case and it depends on the city layout.

Nearest neighbour algorithm

Nearest neighbour algorithm = a greedy and heuristic solution to TSP that uses the closest unvisited node as the next node.

The algorithm is as follows:

1. Start on the current node
2. Find the next closest unvisited node
3. Move to that next node
4. Mark the new node as visited
5. If all nodes have been visited, terminate the search
6. Go back to step 2

Advantages of the nearest neighbour algorithm

- Is very easy to implement
- Executes quite quickly

Disadvantages of the nearest neighbour algorithm

- Due to greedy nature, it can miss some shorter routes, including ones that a human would spot easily
  - This can be assessed by checking if the last moves are of comparable length to the first ones. If not, then it is likely a more optimal solution exists.

Social and ethical challenges to the project

Impact on jobs

- In the US, there are about 4 million people who drive as the main part of their work. These people are most at risk of losing their jobs as their whole job revolves around driving from A to B
- There is a further 11.7 million where driving makes a substantial part of their job, or is a necessary part of it (from US Department of Commerce). For these on-the-job drivers, they could benefit from better productivity and working conditions. This could result in a rise in demand for these jobs. Even if these people lose their jobs, they will likely be able to find other ones in related fields, since their core skills are not actually driving
- Would create a number of jobs too: specialised AV mechanics, car data scientists, and IT technicians working on the infrastructure (from Wired)

The 'trolley problem'

- Idea of the problem: train is going to hit a group of five people, but can be redirected to hit just one person. Given this situation, what would a person do?
- In self driving car situations the problem typically comes down to a situation where the death of either a passenger or pedestrian is inevitable, and it must decide which one lives and which one dies
- In a study by the Toulouse School of Economics, they found that 75% of people thought the car should always swerve to kill a passenger, even to save just one pedestrian.
- Utilitarianism reasoning means the car should take whichever decision would save the most people and result in the most happiness. However, a different perspective comes from Helen Frowe of Stockholm university who believes cars should be programmed to protect innocent bystanders
- Must also consider situations where a person jumps out in front of a car, knowing it will swerve and kill the passenger.

The understandability of neural network models

- From MIT Technology Review
- The decision making part of a self-driving car is very much a black box- we don't really know how some of the most complex algorithms work
- It may, for example, be difficult to find out why a car made a decision that it did. If a car crashes into a tree unexpectedly, we may not be able to work out why and hence be unable to do anything to prevent it in the future.
- It makes it hard to predict when ML algorithms will have failures, and hence makes it hard to guarantee that they will be safe.
- A future EU law could require companies to give users an explanation for how their algorithms make decisions- this would be impossible.

Testing on public roads

- What happens if someone is killed? Recall recent situation where Uber self-driving car killed a woman crossing a dark road. Whilst many agree that the crash was inevitable, many also argue that the AV should have detected the person with LIDAR etc.

- Does it becomes the responsibility of the person sitting in the driver's seat to ensure that nobody comes to harm due to the actions of the autonomous vehicle?
- Number of deaths is supposed to go down as the technology matures—but what happens in the mean time? Are people's lives being put at risk?
- In areas like San Francisco and Arizona companies are testing self-driving cars on public roads, but the public haven't signed anything giving permission for this, they are not given the choice.
- Companies are currently requested to hand in a voluntary safety self-assessment, but very few companies have. Companies are however required to publically report all crashes and report on 'disengagements'.
- There is currently no regulation on a 'driving test' that AVs would have to pass, but this is something that could happen in the future.

Who pays for insurance

- Laws need to be updated to clarify who is to blame for accidents
- Many people believe that the manufacturer should pay. However, this increases the likelihood of a lawsuit and may deter the companies from making self-driving cars in the first place
- Waymo have started a partnership with insurance firm Trov to insure any passengers who use their vehicles. The passengers don't even know this coverage is there, it is something that Waymo takes care of for them. This is in advance of its first commercial ride-hailing service
- Questions would need to be asked about the quality of the model of car and its self-driving capability, rather than driver competency
- Costs may have to go up due to the increased value of technology within the car and the increased likelihood that one of these components could fail.
- If autonomous cars make accidents less frequent, insurers will make more money as there will be fewer claims. However this could make people less likely to pay for insurance, resulting in more uninsured drivers on the road.

Technology to be incorporated throughout town

The wireless technology which has the most traction for both V2I and V2V protocols is DSRC – dedicated short range communication. This standard was designed specifically for automotive use

V2I

Vehicle to infrastructure protocols include technologies that directly link road vehicles to their physical surroundings, primarily to improve road safety

- V2I is similar to V2V (see next) in that it uses short range communication protocols to send data from infrastructure equipment to vehicles in an ad hoc network
- According to Techtarget V2I sensors can give cars real time advisories about local conditions, including: road conditions, traffic congestion, accidents, construction zones, and parking availability
- There is likely to be a significant cost in implementing V2I- it could see a shift away from the public sector maintaining infrastructure and more public-private partnerships, where the private companies can benefit from the various data accumulated

According to the Hungarian MOGI department, V2I has a number of potential applications:

- Safety
  - Drivers can be provided with information about other cars that may otherwise not be obtained
  - Emergency vehicles can be prioritised through the system
  - Cars can be ensured they are operating within legal limits (such as speed)
- Efficiency
  - Traffic can be monitored as well as managed
  - Can detect congestion before it happens
  - Dynamic traffic light and traffic control
  - Connected navigation systems
- Payment and information
  - Could automatically pay for parking and other tolls
  - Can provide drivers with any relevant information they need

Audi have already deployed a system in Las Vegas allowing drivers to see how long until a traffic light turns green. BMW have released a mobile app that does a similar thing.

Other V2I technologies could include:

- Advanced road markings that are visible to humans and machines in any road condition, helping them to stay in lane. They would use lines outside the vision-based spectrum so that cars can better detect lanes
- Smart signs that can be detected in any condition for similar reasons, so that cars always know what they have to do. These can also be detected in any condition and can be fully compatible with traditional signage
- Wireless communication between different devices on the roads so that everyone is informed of the latest information and conditions.
- Alerts to pedestrians if they are distracted by their phones and a car is heading their way

V2V

Vehicle to vehicle protocols would facilitate the communication of different vehicles on the road at the same time

- V2V networks are a form of VANET (vehicular ad hoc network). These are classes of networks that are comprised of decentralised vehicles
- They make use of a part of the radio spectrum that has already been marked as for this purpose
- V2V networks can help cars to overcome blind spots and avoid accidents by sharing information for the benefit of other drivers
- Possible issues could arise if systems are tricked into providing erroneous information and hence cause accidents. Also an issue if one car isn't broadcasting, other cars could be placed into a false sense of security

According to MIT Technology Review:

- Research by General Motors has been able to come up with systems where cars broadcast their position, speed, steering wheel position, brake status, and other data to other vehicles in the local area.
- This form of communication gets around the issues of other sensors not being able to see around obstructions
- There is a prediction that the technology could help prevent more than half a million US accidents annually