ASTM : Autonomous Smart Traffic Management System Using Artificial Intelligence CNN and LSTM

Christofel Rio Goenawan
Master of Robotics and AI
Korea Advanced Institute
of Science and Technology
Daejeon, South Korea
Email: christofel.goenawan@kaist.ac.kr

Hyundai Motors
Department of
Artificial Intelligence Engineering
Hyundai Motors
Seoul, South Korea

Prof. Seung-Hyun Kong
Department of
CCS Graduate School of Mobility
Korea Advanced Institute
of Science and Technology
Daejeon, South Korea
Email: skong@@kaist.ac.kr

Abstract—**In this modern world , Artificial Intelligence (AI) development have helped improvement in a lot of area , including automation , computer vision , fraud detection and others. Artificial Intelligence can be leveraged to increase ASTM and decrease ASTM Congestion Rate. This paper presents Autonomous STM using Artificial Intelligence to increase Traffic Management congestion flow rate. Autonomous Smart Traffic Managements System used YOLO V5 Convolutional Neural Network to detects Traffic Vehicles in Smart Traffic Management Images. Then Autonomous Smart Traffic Management predicts Traffic Vehicles next 12 hours using Recurrent Neural Network Long Short Term Memories. Smart Traffic Management Cycle Length Analysis Manage Smart Traffic Management Cycle Length based on Traffic Vehicles Prediction using Artificial Intelligence. From Result of Recurrent Neural Network Long Short Term Memory to Predict Traffic Vehicles in Next 12 Hours We Can see that Recurrent Neural Network Long Short Term Memory Can predict Traffic Vehicles in Next 12 Hours with Mean Squarred Error 4.521 Vehicles and Root Mean Squared Error (RMSE) 2.232 Vehicles. After Simulation of STM in ASTM Simulation CARLA We got Traffic Management Congestion Flow Rate with ASTM (21 Vehicles / Minutes) is 50 % Higher than Traffic Management Congestion Rate without STM (Around 15 Vehicles / Minute) and Traffic Management Vehicles Pass Delay with STM (5 Seconds / Vehicles) are 70 % Less Than Traffic Management Vehicles Pass Delay without STM (Around 12 Seconds / Vehicles). Then We Can See that STM using Artificial Intelligence can increase Traffic Management Congestion Flow Rate with 50 % Traffic Management Congestion Flow Rate and decrease Traffic Management Vehicles Pass Delay with 70 % Traffic Management Vehicles Pass Delay.**

I. Introduction

A. *Artificial Intelligence*

The development of Artificial Intelligence (AI) began from 1943, where neurophysiologist Warren McCulloch and mathematician Walter Pitts published a paper introduced Artificial Neural Network (ANN) to the world [2]. Then the develpment of AI started to gain attention when British Polymath , Alan Turing , published his paper "Can Machines Think?" in 1950 where Turing suggested that machines can do the same thing as humans to use available information and reason in order to solve problems and make decisions. The development of AI officially started after Allen Newell, Cliff Shaw, and Herbert Simon published the *proof of concept* in the first AI program named *Logic Theorist* in 1955 [3].

After that until 1974, AI growth very rapidly because a lot of investment put in the field and computer could store more information and became faster, cheaper, and more accessible. But until the start of 1980 , AI entered the "dark era" because of many obstacle founded , such lack of computational power to do anything substantial and computers simply couldn't store enough information or process it fast enough that discouraged many investor and researcher to dig deeper in this field . And until end of 20th century AI development through roller coaster of success and setbacks , including famous "deep learning" techniques which allowed computers to learn using experience popilarized by John Hopfield and David Rumelhart in 1980s.

In 1997 AI regained the hype after IBM's Deep Blue, a chess playing computer program , defeated reigning world chess champion and grand master Gary Kasparov in 1997. But it not until 2012 when AlexNet CNN Architectures by Alex Krizhevsky and his team won the 2012 ImageNet annual image recognition challenge by a huge margin, where it ignited again the development in AI field. Until now numerous development have been founded in AI field , because of huge data available to train the model , tremendous increase in computing power since 1990s and many revolutionary AI Architectures have been founded with much higher performance and usability.

Nowadays , Artificial Intelligence (AI) development have helped improvement in a lot of area , from scientific research , industry , environmental area until governmental and social issues. The example is AI has been shown effective in solving a variety of practical problems such as disease detection [4], language translation [5], autonomous self-driving cars [6] and customer behavior prediction [7].

However the AI development have been hampered by difficulty of sharing ML models and difference in dependencies and machine's environment , that make one model can't always be deployed in another machines [8]. Usually ML models contain multi-stage, complex pipelines with procedures that are sequentially entangled and mixed together, such as pre-processing, feature extraction, data transformation, training, and validation [9]. Hence improving an individual component may in fact worsen the whole performance , because the part is strongly correlated with other components. Therefore, building models becomes a trial-and-error-based iterative process which demands expert level knowledge in ML concepts to create and tune the ML models manually [10]. Moreover because of each dependencies for different AI tools like Tensor Flow , Keras and PyTorch , the machine dependencies must be installed manually to provide the tools environment that usually is a time- wasting process and not always an easy case.

B. *Object Detection using Computer Vision*

Object Detection whis Computer Vision is Detecting Object around Artificial Intelligence Sensor Image Camera to detect Object using Artificial Intelligence. In computer vision, we have a convolutional neural network that is very popular for computer vision tasks like image classification, object detection, image segmentation and a lot more. Image classification is one of the most needed techniques in today's era, it is used in various domains like healthcare, business, and a lot more, so knowing and making your own state of the art computer vision model is a must if you're in a domain of AI. Most computer vision algorithms use something called a convolution neural network, or CNN. A CNN is a model used in machine learning to extract features, like texture and edges, from spatial data. Like basic feedforward neural networks, CNNs learn from inputs, adjusting their parameters (weights and biases) to make an accurate prediction. However, what makes CNNs special is their ability to extract features from images. Take an image of a car, for example. In a normal feedforward neural network, the image would be flattened into a feature vector. However, CNNs are able to treat images like matrices as they exist and extract spatial features from them, like texture, edges and depth. They do this by using convolutional layers and pooling. The Architecture of Artificial Intelligence Convolutional Neural Network using CNN Can be Seen as below.

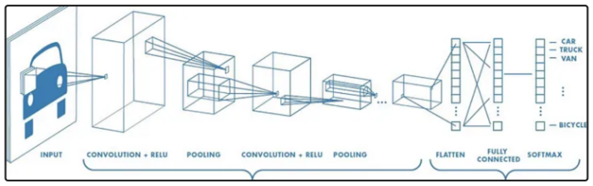


Fig. 1. Convolutional Neural Network for Image Detection in Artificial Intelligence

Take an image of a car, for example. In a normal feed-forward neural network, the image would be flattened into a feature vector. However, CNNs are able to treat images like matrices as they exist and extract spatial features from them, like texture, edges and depth. They do this by using convolutional layers and pooling. You can think of convolutional layers as a set of feature maps — the convolutional layer applies a series of image filters to an input image represented as a matrix. The resulting filter images, or feature maps, have different appearances, as they extracted different features. Above, you can see a convolution kernel in action. The blue matrix represents the input image, as the shaded square passing over it is the convolution kernel. The convolution kernel is a matrix of weights (the subscripts in the shaded square) that passes over a larger matrix.Each weight is multiplied by the larger number it is shaded over. Then, all of the numbers in the shaded region (9, in this case), are summed, resulting in the green matrix on the right. This green matrix is called a feature map.Convolutional layers usually have multiple convolution kernels with different weights, resulting in different feature maps. The number of feature maps is what gives the convolutional layer its depth.These image filters are called convolutional kernels.

C. *Traffic Congestion Prediction using Artificial Intelligence*

One of the use of Artificial Intelligence is Predict Traffic Congestion using Artificial Intelligence. By using Traffic Time , Traffic Jam condition and weather condition Artificial Intelligence can predict Traffic Congestion using Artificial Intelligence. Recurrent neural networks (RNNs) are deep learning models, typically used to solve problems with sequential input data such as time series. What are they, and how do we use them in time series forecasting? RNNs are a type of neural network that retains a memory of what it has already processed and thus can learn from previous iterations during its training. Probably you have done what most of us do when we hear any technical term for the first time. You have tried to understand what recurrent neural networks are by clicking on the top-listed non-ad Google search result. Then you will have found that Wikipedia's article exhibits a high level of abstraction. It is of limited usefulness when we try to understand what RNNs are and what they are for: "A recurrent neural network (RNN) is a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence. This allows it to exhibit temporal dynamic behavior. Derived from feedforward neural networks, RNNs can use their internal state (memory) to process variable length sequences of inputs .... Recurrent neural networks are theoretically Turing complete and can run arbitrary programs to process arbitrary sequences of inputs." Say what? Michael Phi provided an excellent, non-mathematical guide on RNNs in a previous Towards Data Science article of his: "Illustrated Guide to Recurrent Neural Networks — by Michael Phi — Towards Data Science". So did Will Koehrsen, in "Recurrent Neural Networks by Example in Python — by Will Koehrsen — Towards Data Science." Let me summarize the basics we should understand about RNNs, in non-mathematical terms (and then I'd refer you to the additional explanations and illustrations in the two articles Michael and Will wrote in 2018). A neural network — of which recurrent neural networks are one type, among other types such as convolutional networks — is composed of three elementary components: the input layer, the hidden layers, and the output layer. Each layer consists of so-called nodes (aka neurons). I've read the following analogy for the three main types of neural networks, which are said to mimic human brain functions in specific ways. The following comparisons oversimplify, so best take them with a grain of salt.

1) The temporal lobe of our brain =¿ artificial neural networks =¿ mainly for classification and regression problems =¿ one of the functions of the temporal lobe is long-term memory
2) The occipital lobe =¿ convolutional neural networks =¿ mainly for computer vision problems (though temporal convolutional networks, TCNs, can be applied to time series)
3) The frontal lobe =¿ recurrent neural networks RNN =¿ mainly for time series analysis, sequences, and lists — for instance, in language processing, which deals with sequences of characters, words, and sentences ordered by a grammar; or time series, which consist of temporal sequences of observations =¿ one of the frontal lobe's functions is short-term memory

Feed-forward neural networks (FFNNs) — such as the grandfather among neural networks, the original single-layer perceptron, developed in 1958— came before recurrent neural networks. In FFNNs, the information flows in only one direction: from the input layer, through the hidden layers, to the output layer, but never backwards in feedback loops. FFNN are often used in pattern recognition. The FFNN multiplies a matrix of weight factors with the inputs and generates the outputs from these weighted inputs. Feed-forward neural networks don't retain a memory of the inputs they have processed. They suffer from anterograde amnesia, the inability to form new memories (similar to the protagonist in Christopher Nolan's movie Memento — Wikipedia [this seemed a rare opportunity to mention anterograde amnesia and Memento in a data science article]). A recurrent neural network, by contrast, retains a memory of what it has processed in its recent previous steps (we'll come back to the "recent" qualifier in a minute). It makes recurrent connections by going through temporal feedback loops: the output of a preceding step is used as an input for the current process step. Unlike amnesiac FFNNs, this memory enables RNNs to process sequences of inputs without loosing track. The loops make it a recurrent network. Another distinguishing characteristic of recurrent networks is that they share parameters across each layer of the network. While feedforward networks have different weights across each node, recurrent neural networks share the same weight parameter within each layer of the network. That said, these weights are still adjusted in the through the processes of backpropagation and gradient descent to facilitate reinforcement learning. Recurrent neural networks leverage backpropagation through time (BPTT) algorithm to determine the gradients, which is slightly different from traditional backpropagation as it is specific to sequence data. The principles of BPTT are the same as traditional backpropagation, where the model trains itself by calculating errors from its output layer to its input layer. These calculations allow us to adjust and fit the parameters of the model appropriately. BPTT differs from the traditional approach in that BPTT sums errors at each time step whereas feedforward networks do not need to sum errors as they do not share parameters across each layer. Through this process, RNNs tend to run into two problems, known as exploding gradients and vanishing gradients. These issues are defined by the size of the gradient, which is the slope of the loss function along the error curve. When the gradient is too small, it continues to become smaller, updating the weight parameters until they become insignificant—i.e. 0. When that occurs, the algorithm is no longer learning. Exploding gradients occur when the gradient is too large, creating an unstable model. In this case, the model weights will grow too large, and they will eventually be represented as NaN. One solution to these issues is to reduce the number of hidden layers within the neural network, eliminating some of the complexity in the RNN model. The Architecture of Recurrent Neural Network Artificial Intelligence for Predicting Traffic Congestion System using Artificial Intelligence can be seen as below.

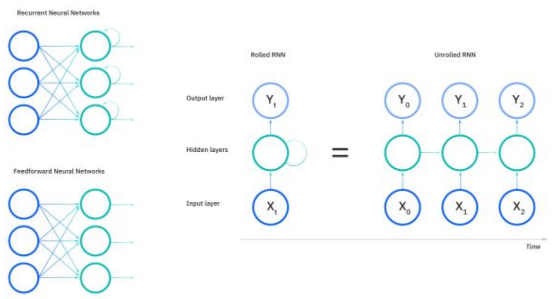


Fig. 2. Recurrent Neural Network Architecture for Traffic Congestion System Prediction using Artificial Intelligence

D. *Smart Traffic Management Systems*

One of the cornerstones in smart city design is having an integrated smart transportation solution. It can be argued that a city is not completely intelligent without a smart traffic management system. Intelligent transportation systems (ITS), or smart traffic management systems (Figure 1), provide an organized, integrated approach to minimizing congestion and improving safety on city streets through connected technology. PR Newswire expects the intelligent traffic management system market to grow to 19.91 billion by 2028 at a 10.1% CAGR. The demand and increased adoption rate of smart traffic management solutions can be attributed to the boom of smart city technology. Guidehouse Insights reports that there are more than 250 smart city projects globally. Symmetry Electronics supplier, Digi International, defines smart traffic management systems as technology solutions that municipalities can integrate into their traffic cabinets and intersections today for fast, cost-effective improvements in safety and traffic flow on their city streets. Efficient and successful smart traffic management systems utilize next-generation hardware and software to optimize traffic infrastructure (Figure 2). Architecture of Smart Traffic Management System using Artificial Intelligence can be seen as below.

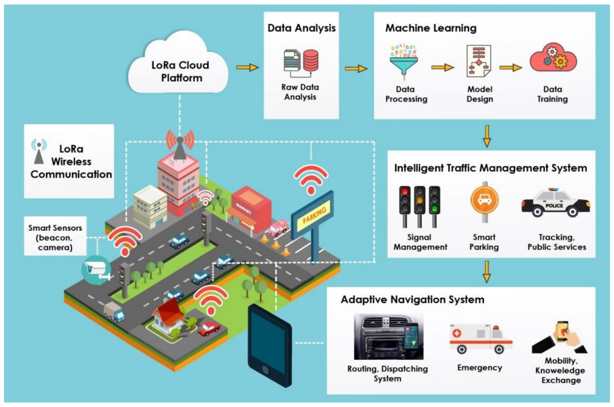

LoRa Cloud Platform
Data Analysis
Raw Data Analysis
Machine Learning
Data Processing
Model Design
Data Training
LoRa Wireless Communication
Intelligent Traffic Management System
Signal Management
Smart Parking
Tracking, Public Services
Smart Sensors (beacon, camera)
PARKING
POLICE
Adaptive Navigation System
Routing, Dispatching System
Emergency
Mobility, Knowledge Exchange

Fig. 3. Architecture of Smart Traffic Management using Artificial Intelligence

Transportation plays a crucial role in any community, connecting people to jobs, services, and opportunities. Monitoring key metrics can provide valuable insight into how a transportation system is performing, and whether it is meeting the needs of those who rely on it. This blog explores some key transportation metrics to analyze in any community, including measures of mobility, road safety, and accessibility. By understanding these metrics, community leaders and transportation professionals can make informed decisions to improve transportation systems and enhance the overall quality of life for residents in their communities. Metrics to Measure Traffic Congestion in Smart Traffic Management can be seen as below.

1) Average Daily Traffic (ADT) and Annual Average Daily Traffic (AADT) ADT and AADT both quantify how busy a stretch of road or highway is. They do this by reporting the number of vehicles passing through over the course of a day, or year, respectively. ADT and AADT have many applications within traffic engineering such as signal timing, determining where infrastructure investments should go, and much more. We wrote a whole blog post diving into AADT; read it here!
Minimum data needed to calculate this metric: traffic volumes.
2) Corridor Travel Times Corridor Travel Times help determine how long it takes for vehicles to travel along specific roadways or corridors. Many factors can impact travel times such as poor traffic signal timing, events, or car crashes. Analyzing Corridor Travel Times helps transportation professionals identify when travel times rise above the baseline and develop solutions to improve traffic flow, such as adding lanes, synchronizing traffic signals, or increasing the enforcement of traffic laws. Corridor Travel Times can also be used to evaluate the effectiveness of transportation infrastructure and policy changes. For example, if a new LRT line is built, travel times on the corridor can be tracked before and after the change to measure the impact of the new service on traffic flow.
Minimum data needed to calculate this metric: Location-based data i.e from Bluetooth devices or smartphones.
3) Vehicle Miles Traveled (VMT) VMT measures the total distance that vehicles travel on a specific road or network of roads over a certain period of time. It is a useful metric that is closely linked to important factors such as traffic congestion, environmental impact, energy consumption, and economic activity. For instance, a high VMT on a road may indicate it's over capacity and in need of expansion or improvement. As an indicator of the environmental impact of transportation, monitoring VMT can help identify areas to implement emission reduction strategies.
Minimum data needed to calculate this metric: traffic volumes, road segment lengths.
4) Crash Rates Crash Rates are a key traffic metric to evaluate the relative safety of different roads or intersections. They are based on the number of crashes, traffic volume, and road segment lengths in that location over time. Analyzing crash rates alongside data like traffic volumes or road design helps officials target and address specific safety issues.
Minimum data needed to calculate this metric: traffic volumes, crash data, road segment lengths, traffic volumes.

II. Methodology

In this Project Writer propose Novel Architecture of Smart Traffic Management System by using Artificial Intelligence to Intelligently Manage Traffic System in Smart Traffic Management System. First We use Artificial Intelligence to Detect Traffic Management Vehicles in Traffic Management System images. After We detect Traffic Management Vehicles in Traffic Management System images We Predict the number of Traffic Management Vehicles System using Artificial Intelligence to understand how many Smart Traffic Management System Vehicles in Smart Traffic Management System. Then Writer manages the Smart Traffic Management System Traffic Length Cycle based on Fuzzy Decision Making of Smart Traffic Management System Traffic Length. The System of Autonomous Smart Traffic Management System using Artificial Intelligence can be seen in figure 4.

A. *Smart Traffic Management System Vehicles Detection using Convolutional Neural Network*

First We detect Traffic Management System Vehicles in Traffic Management System images using Artificial Intelligence Convolutional Neural Network. Artificial Intelligence Convolutional Neural Network detect Traffic Management System Vehicles in Traffic Management System Images using Pixels and Colours on Traffic Management System images. In the Smart Traffic Management System Vehicles Detection

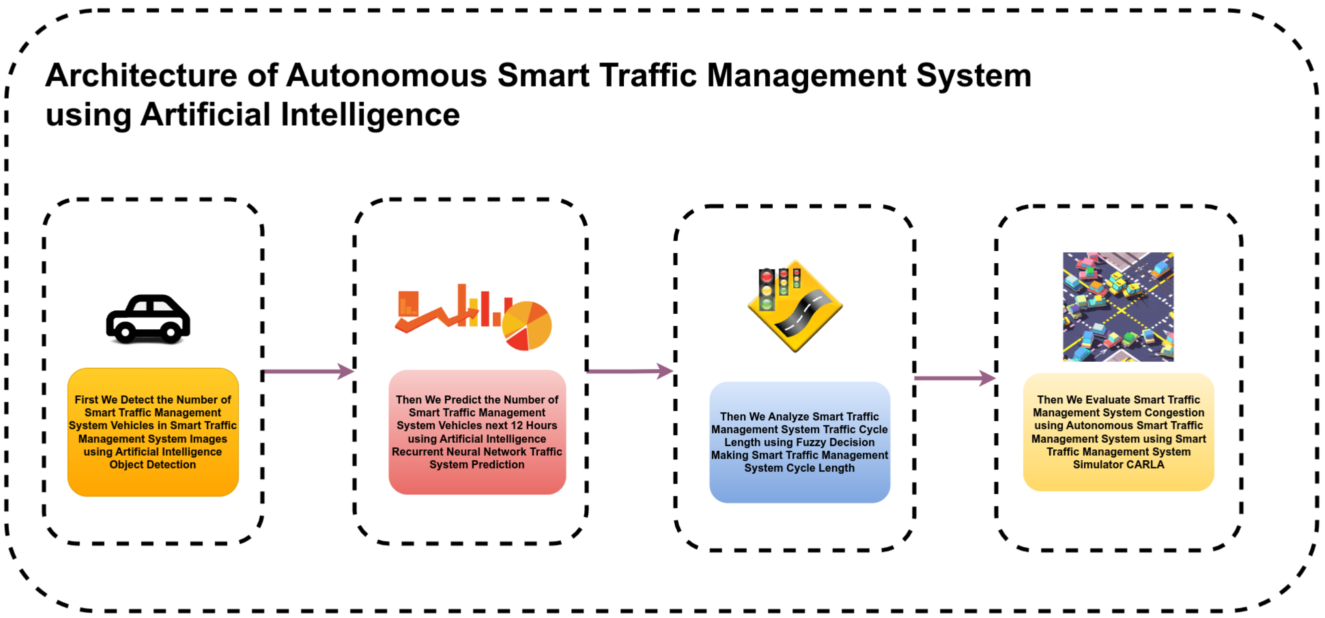

Architecture of Autonomous Smart Traffic Management System using Artificial Intelligence
First We Detect the Number of Smart Traffic Management System Vehicles in Smart Traffic Management System Images using Artificial Intelligence Object Detection
Then We Predict the Number of Smart Traffic Management System Vehicles next 12 Hours using Artificial Intelligence Recurrent Neural Network Traffic System Prediction
Then We Analyze Smart Traffic Management System Traffic Cycle Length using Fuzzy Decision Making Smart Traffic Management System Cycle Length
Then We Evaluate Smart Traffic Management System Congestion using Autonomous Smart Traffic Management System using Smart Traffic Management System Simulator CARLA

Fig. 4. Architecture of Autonomous Smart Traffic Management System using Artificial Intelligence

using Convolutional Neural Network Writer using Traffic Management System Vehicles Images in *Road Vehicle Image Dataset of Bangladesh Traffic* made by Ashfak Yeafi in 2017. This dataset contains Bangladesh road valencias images with annotation. There are two separate folders in this dataset, one contains train images and the other contains validation images. This Road Vehicle Image Dataset contains more than Million of Road Vehicle Image Dataset in Bangladesh from January 2017 until December 2017.

For Smart Traffic Management System Vehicles Detection using Convolutional Neural Network Writer using YOLO V5 Convolutional Neural Network. YOLO V5 Convolutional Neural Network is one of most used Convolutional Neural Network Python Package used for Smart Traffic Management System Vehicle Detection. YOLOv5 is a model in the You Only Look Once (YOLO) family of computer vision models. YOLOv5 is commonly used for detecting objects. YOLOv5 comes in four main versions: small (s), medium (m), large (l), and extra large (x), each offering progressively higher accuracy rates. Each variant also takes a different amount of time to train. Object detection, a use case for which YOLOv5 is designed, involves creating features from input images. These features are then fed through a prediction system to draw boxes around objects and predict their classes. The YOLO model was the first object detector to connect the procedure of predicting bounding boxes with class labels in an end to end differentiable network.

The YOLO network consists of three main pieces.

1) Backbone: A convolutional neural network that aggregates and forms image features at different granularities.
2) Neck: A series of layers to mix and combine image features to pass them forward to prediction.
3) Head: Consumes features from the neck and takes box and class prediction steps.

The Architecture of YOLO V5 Smart Traffic Management System Vehicles Detection using Convolutional Neural Network can be seen as below.

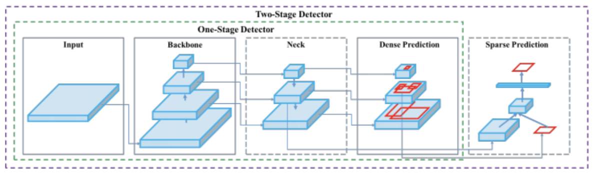


Fig. 5. Architecture of YOLO V5 Smart Traffic Management System Vehicles Detection using Convolutional Neural Network

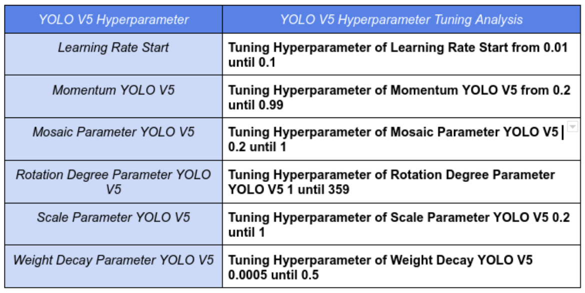
In Traffic Management System Vehicles Detection Writer Tuning Hyperparameter of YOLO V5 Smart Traffic Management System Vehicles Detection using Convolutional Neural Network. Writer tuning 5 Most important Hyperparameter of YOLO V5 Smart Traffic Management System Vehicles Detection. 5 Most Important Tuning Hyperparameter of YOLO V5 Smart Traffic Management System Vehicles Detection using Convolutional Neural Network are as below.

1) Learning-rate start (lr0): learning rate starts to determine step size at each iteration. I.e. if you configure the learning rate (0.1) before training, then its means at every iteration, your training progress will increase by 0.1.
2) Momentum: Momentum, is the tuning parameter for the gradient descent algorithm, its work is to replace the gradient with an aggregate of gradient
3) Mosaic: Mosaic, is used to increase model accuracy by creating a new image from a combination of multiple images, and then using newly created images for train-

ing. It is well known for data augmentation and finding optimal feature techniques.

4) Degree: Degree, is used to increasing model accuracy by randomly rotating images up to 360 degrees in whole training data. This parameter helps most when you need to detect objects in multiple positions/angles.
5) Scaling: Scaling, is used to resize an image either to match with grid size or for optimization of results.
6) Weight Decay: A type of regularization technique, that works by adding a penalty term to the cost function of a network, which has the effect of shrinking/compressing the weights during the backpropagation process.

The 6 Most Important Tuning Hyperparameter of YOLO V5 Smart Traffic Management System Vehicles Detection using Neural Network can be seen as below.



YOLO V5 Hyperparameter	YOLO V5 Hyperparameter Tuning Analysis
Learning Rate Start	Tuning Hyperparameter of Learning Rate Start from 0.01 until 0.1
Momentum YOLO V5	Tuning Hyperparameter of Momentum YOLO V5 from 0.2 until 0.99
Mosaic Parameter YOLO V5	Tuning Hyperparameter of Mosaic Parameter YOLO V5 | 0.2 until 1
Rotation Degree Parameter YOLO V5	Tuning Hyperparameter of Rotation Degree Parameter YOLO V5 1 until 359
Scale Parameter YOLO V5	Tuning Hyperparameter of Scale Parameter YOLO V5 0.2 until 1
Weight Decay Parameter YOLO V5	Tuning Hyperparameter of Weight Decay YOLO V5 0.0005 until 0.5

B. *Traffic Vehicles Prediction using Recurrent Neural Network*

After understand how many Smart Traffic Management System Vehicles in Smart Traffic Management System images Writer Predict Next 12 Hours of Smart Traffic Management System Vehicles in Smart Traffic Management System using Artificial Intelligence. Writer uses Artificial Intelligence to predict next 12 Hours of Smart Traffic Management System Vehicles in Smart Traffic Management System. Writer uses Artificial Intelligence Recurrent Neural Network to predict next 12 hours of Smart Traffic Management System Vehicles in Smart Traffic Management System. Long Short Term Memory is a kind of recurrent neural network. In RNN output from the last step is fed as input in the current step. LSTM was designed by Hochreiter Schmidhuber. It tackled the problem of long-term dependences of RNN in which the RNN cannot predict the word stored in the long-term memory but can give more accurate predictions from the recent information. As the gap length increases RNN does not give an efficient performance. LSTM can by default retain the information for a long period of time. It is used for processing, predicting, and classifying on the basis of time-series data. Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network (RNN) that is specifically designed to handle sequential data, such as time series, speech, and text. LSTM networks are capable of learning long-term dependencies in sequential data, which makes them well suited for tasks such as language translation, speech recognition, and time series forecasting. A traditional RNN has a single hidden state that is passed through time, which can make it difficult for the network to learn long-term dependencies. LSTMs address this problem by introducing a memory cell, which is a container that can hold information for an extended period of time. The memory cell is controlled by three gates: the input gate, the forget gate, and the output gate. These gates decide what information to add to, remove from, and output from the memory cell. The input gate controls what information is added to the memory cell. The forget gate controls what information is removed from the memory cell. And the output gate controls what information is output from the memory cell. This allows LSTM networks to selectively retain. The Architecture of Recurrent Neural Network Long Short Term Memory for Predicting Traffic Vehicles in Smart Traffic Management System can be seen as below.

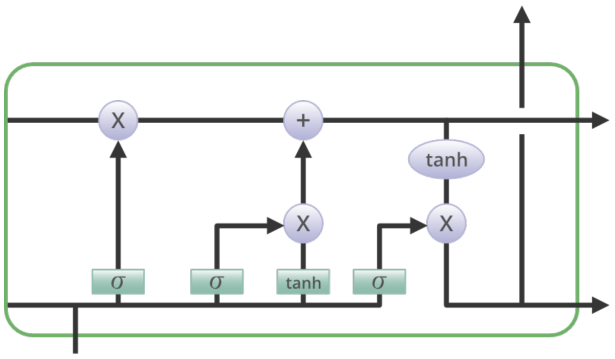
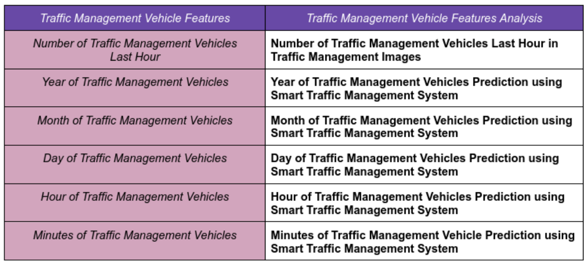


Fig. 6. Architecture of Recurrent Neural Network Long Short Term Memory for Predicting Traffic Vehicles in Smart Traffic Management System

In Recurrent Neural Network Long Short Term Memory for Predicting Traffic Management System Vehicles Writer using some Features for predicting Traffic Management System Vehicles next 12 Hours. Features of Traffic Management System Vehicles for Predicting Traffic Management System Vehicles next 12 Hours can be seen as below.



Traffic Management Vehicle Features	Traffic Management Vehicle Features Analysis
Number of Traffic Management Vehicles Last Hour	Number of Traffic Management Vehicles Last Hour in Traffic Management Images
Year of Traffic Management Vehicles	Year of Traffic Management Vehicles Prediction using Smart Traffic Management System
Month of Traffic Management Vehicles	Month of Traffic Management Vehicles Prediction using Smart Traffic Management System
Day of Traffic Management Vehicles	Day of Traffic Management Vehicles Prediction using Smart Traffic Management System
Hour of Traffic Management Vehicles	Hour of Traffic Management Vehicles Prediction using Smart Traffic Management System
Minutes of Traffic Management Vehicles	Minutes of Traffic Management Vehicle Prediction using Smart Traffic Management System

Writer uses Recurrent Neural Network Long Short Term Memory for predicting Traffic Management System Vehicles next 12 Hours. Architecture of Recurrent Neural Network Long Short Term Memory for predicting Traffic Management System Vehicles next 12 Hours can be seen in Figure 7.

C. *Smart Traffic Management Traffic Cycle Length Analysis using Fuzzy Decicion Making*

Smart Traffic Management Cycle Length Analysis Manage Smart Traffic Management Cycle Length based on Smart

Traffic Management Vehicles Prediction using Artificial Intelligence. Cycle length is composed of the total signal time to serve all of the signal phases including the green time plus any change interval. Longer cycles will accommodate more vehicles per hour but that will also produce higher average delays.

The best way is to use the shortest practical cycle length that will serve the traffic demand. Vehicles at a signal installation do not instantaneously enter the intersection. Early studies by Greenshields found that the first vehicle had a starting delay of 3.7 seconds to enter the intersection with subsequent vehicles requiring an average of 2.1 seconds each. Generally, vehicles will pass over an approach detector with a headway of 2 to 2.5 seconds. For general calculation purposes, an average time of 2.5 seconds per vehicle to enter the intersection is a conservative value. This value can be used to estimate signal timing for planning purposes. The cycle length includes the green time plus the vehicle signal change interval for each phase totaled to include all signal phases. A number of methods have been used to determine cycle lengths as outlined in the Highway Capacity Manual, ITE Manual on Traffic Signal Design, and ITE Transportation and Traffic Engineering Handbook. Webster provided the basic empirical formula that would minimize intersection delay as follows:

$$C = (1.5 * L + 5)/(1.0 - SY_i))$$

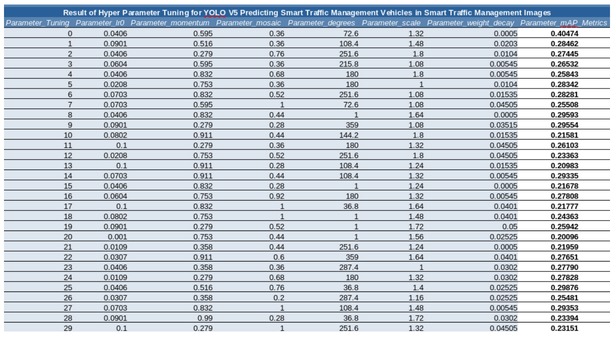
Where: C = optimum cycle length in seconds adjusted usually to the next highest 5 second interval. Cycle lengths in the range of 0.75C to 1.5C do not significantly increase delay. L = Unusable time per cycle in seconds usually taken as a sum of the vehicle signal change intervals. SY_i = critical lane volume each phase/saturation flow

The saturation flow will be between 1500 and 1800 vehicles per hour. Refer to Highway Capacity Manual. The ”Y” value should be computed for each phase and totaled to arrive at SYi for all phases. Note: The traffic volumes used should be the predicted volumes at time of signal turn-on. The volumes should also be the peak hour or peak fifteen-minute period for the cycle determination. When the cycle length has been determined the vehicle signal changes are deducted giving the total cycle green time which can be proportioned to each signal phase on the basis of critical lane volumes. The individual signal phase times are then the proportioned time plus the vehicle change interval on each phase.To ensure that critical lane volumes are adequately served, a capacity check should be computed for each green interval.

D. Evaluate Smart Traffic Management System using Artificial Intelligence using Traffic Management Simulator CARLA

After Decide Smart Traffic Management System Traffic Cycle Length using Smart Traffic Cycle Length Fuzzy Decision Making User will make simulation in Smart Traffic Management Simulator CARLA to simulate Smart Traffic Management System using Artificial Intelligence. CARLA is an open-source autonomous driving simulator. It was built from scratch to serve as a modular and flexible API to address a range of tasks involved in the problem of autonomous driving. One of the main goals of CARLA is to help democratize autonomous driving RD, serving as a tool that can be easily accessed and customized by users. To do so, the simulator has to meet the requirements of different use cases within the general problem of driving (e.g. learning driving policies, training perception algorithms, etc.). CARLA is grounded on Unreal Engine to run the simulation and uses the OpenDRIVE standard (1.4 as today) to define roads and urban settings. Control over the simulation is granted through an API handled in Python and C++ that is constantly growing as the project does. In order to smooth the process of developing, training and validating driving systems, CARLA evolved to become an ecosystem of projects, built around the main platform by the community. In this context, it is important to understand some things about how does CARLA work, so as to fully comprehend its capabilities.

First Writer Tuning Hyperparameter YOLO V 5 Convolutional Neural Network to Detect Smart Traffic Management System Vehicles. Tuning Hyperparameter YOLO V5 Convolutional Neural Network to Detect Smart Traffic Management System Vehicles Can Be Seen in Figure 8.



Result of Hyper Parameter Tuning for YOLO V5 Predicting Smart Traffic Management Vehicles in Smart Traffic Management Images							
Parameter_Tuning	Parameter_lr0	Parameter_momentum	Parameter_mosaic	Parameter_degrees	Parameter_scale	Parameter_weight_decay	Parameter_mAP_Metrics
0	0.0406	0.595	0.36	72.6	1.32	0.0005	**0.40474**
1	0.0901	0.516	0.36	108.4	1.48	0.0203	**0.28462**
2	0.0406	0.279	0.76	251.6	1.8	0.0104	**0.27445**
3	0.0604	0.595	0.36	215.8	1.08	0.00545	**0.26532**
4	0.0406	0.832	0.68	180	1.8	0.00545	**0.25843**
5	0.0208	0.753	0.36	180	1	0.0104	**0.28342**
6	0.0703	0.832	0.52	251.6	1.08	0.01535	**0.28281**
7	0.0703	0.595	1	72.6	1.08	0.04505	**0.25508**
8	0.0406	0.832	0.44	1	1.64	0.0005	**0.29593**
9	0.0901	0.279	0.28	359	1.08	0.03515	**0.29554**
10	0.0802	0.911	0.44	144.2	1.8	0.01535	**0.21581**
11	0.1	0.279	0.36	180	1.32	0.04505	**0.26103**
12	0.0208	0.753	0.52	251.6	1.8	0.04505	**0.23363**
13	0.1	0.911	0.28	108.4	1.24	0.01535	**0.20983**
14	0.0703	0.911	0.44	108.4	1.32	0.00545	**0.29335**
15	0.0406	0.832	0.28	1	1.24	0.0005	**0.21678**
16	0.0604	0.753	0.92	180	1.32	0.00545	**0.27808**
17	0.1	0.832	1	36.8	1.64	0.0401	**0.21777**
18	0.0802	0.753	1	1	1.48	0.0401	**0.24363**
19	0.0901	0.279	0.52	1	1.72	0.05	**0.25942**
20	0.001	0.753	0.44	1	1.56	0.02525	**0.20096**
21	0.0109	0.358	0.44	251.6	1.24	0.0005	**0.21959**
22	0.0307	0.911	0.6	359	1.64	0.0401	**0.27651**
23	0.0406	0.358	0.36	287.4	1	0.0302	**0.27790**
24	0.0109	0.279	0.68	180	1.32	0.0302	**0.27828**
25	0.0406	0.516	0.76	36.8	1.4	0.02525	**0.29876**
26	0.0307	0.358	0.2	287.4	1.16	0.02525	**0.25481**
27	0.0703	0.832	1	108.4	1.48	0.00545	**0.29353**
28	0.0901	0.99	0.28	36.8	1.72	0.0302	**0.23394**
29	0.1	0.279	1	251.6	1.32	0.04505	**0.23151**

Fig. 7. Result Table of Tuning Hyperparameter YOLO V5 to Predicting Smart Traffic Management System Vehicles in Smart Traffic Management Images

From the Result Table of Tuning Hyperparameter YOLO V5 Convolutional Neural Network to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images We Can See the best Tuning Hyperparameter are Tuning Hyperparameter Search 1 with Parameter Learning Rate 0 0.0052 , Parameter Momentum 0.595 , Parameter Mosaic 0.36 , Parameter Degrees 72 , Parameter Scale 1.32 , and Parameter Weight Decay 0.0005 with Mean Average Precision 0.4034 that is Almost 50 % more higher than other Tuning Hyperparameter YOLO V5 Convolutional Neural Network to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images (Around 0.2852). Result of Best Tuning Hyperparameter YOLO V5 Convolutional Neural Network to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images can be seen as below.

Then Writer Training YOLO V5 Convolutional Neural Network Best Tuning Hyperparamneter to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images.

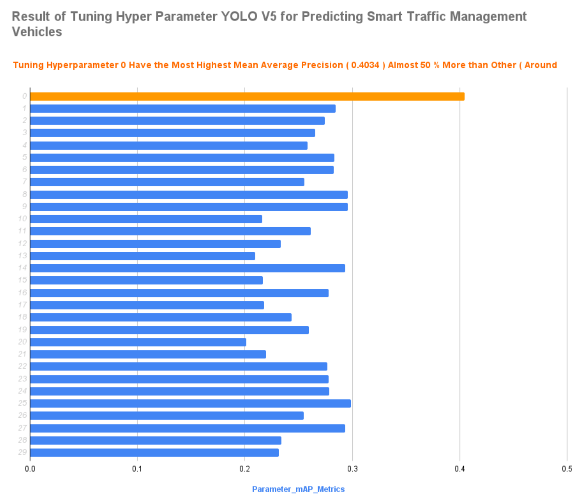


Fig. 8. Result Best Tuning Hyperparameter YOLO V5 to Predicting Smart Traffic Management System Vehicles in Smart Traffic Management Images

Writer using Training YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images using Best Tuning Hyperparameter Parameter Learning Rate 0.00406 , Parameter Momentum 0.595 , Parameter Mosaic 0.36 , Parameter Degress 72.6 , Parameter Scale 1.32 and Parameter Weight Decay Tuning YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images 0.0005. Result of Writer training YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images can be seen in Figure 9.

From Training YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images We Can Have YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter can detect Smart Traffic Management Vehicles in Smart Traffic Management Images with Mean Average Precision 0.88561 and Mean Average Precision for Predicting Smart Traffic Management Cars in Smart Traffic Management Images is 0.95251. From Training YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter Training Loss We Can See that YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter have Loss 0.0010 that is 50 % lower than YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter Loss in Epoch 1 (around 0.0250) because YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter training YOLO V5 Convolutional Neural Network well. We Can also see that YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter Validation loss is 0.0020 that is same with YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter Training loss (aroound 0.0025) because YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter uses Best Tuning Hyperparameter to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images so there is no Overfitting in YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images. The YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images can be seen in Figure 10.

Then Writer training Recurrent Neural Network Long Short Term Memory to predict Smart Traffic Management Vehicles. Writer training Recurrent Neural Network Long Short Term Memory to predict Smart Traffic Management Vehicles in next 12 hours using Recurrent Neural Network Long Short Term Memory using Number of Traffic Management Vehicles last hour , Year of Traffic Management Vehicles , Month of Traffic Management Vehicles , Day of Traffic Management Vehicles , Hour of Traffic Management Vehicles and Minutes of Traffic Management Vehicles. The Result of Recurrent Neural Network Long Short Term Memory to predict Smart Traffic Management Vehicles Next 12 Hours can be seen as below.

From Result of Recurrent Neural Network Long Short Term Memory to Predict Smart Traffic Management Vehicles in Next 12 Hours We Can see that Recurrent Neural Network Long Short Term Memory Can predict Smart Traffic Management Vehicles in Next 12 Hours with Mean Squarred Error 4.521 Vehicles and Root Mean Squared Error (RMSE) 2.232 Vehicles. From Result of Recurrent Neural Network Long Short Term Memory to predict Smart Traffic Management Vehicles in Next 12 Hours We Can See that Recurrent Neural Network Long Short Term Memory to Predict Smart Traffic Management Vehicles in Next 12 Hours can predict increase and decrease of Smart Traffic Management Vehicles in next 12 hours.

Then Writer evaluate Smart Traffic Management System using Artificial Intelligence using Traffic Management Simulator CARLA. In evaluate Smart Traffic Management System using Artificial Intelligence using Traffic Management Simulator CARLA Writer evaluates 100 Scenarios of Smart Traffic Management Vehicles in each day by using Smart Traffic Management System using Artificial Intelligence and by using Traffic Management System. The Evaluation of Smart Traffic Management System using Artificial Intelligence using Simulator CARLA can be seen as below.

After Simulation of Smart Traffic Management System in Traffic Management System Simulation CARLA We got Traffic Management Congestion Flow Rate with Traffic Management System (21 Vehicles / Minutes) is 50 % Higher than Traffic Management Congestion Rate without Smart Traffic Management System (Around 15 Vehicles / Minute) and Traffic Management Vehicles Pass Delay with Smart Traffic Management System (5 Seconds / Vehicles) are 70 % Less Than Traffic Management Vehicles Pass Delay without Smart Traffic Management System (Around 12 Seconds / Vehicles). Then We Can See that Smart Traffic Management System

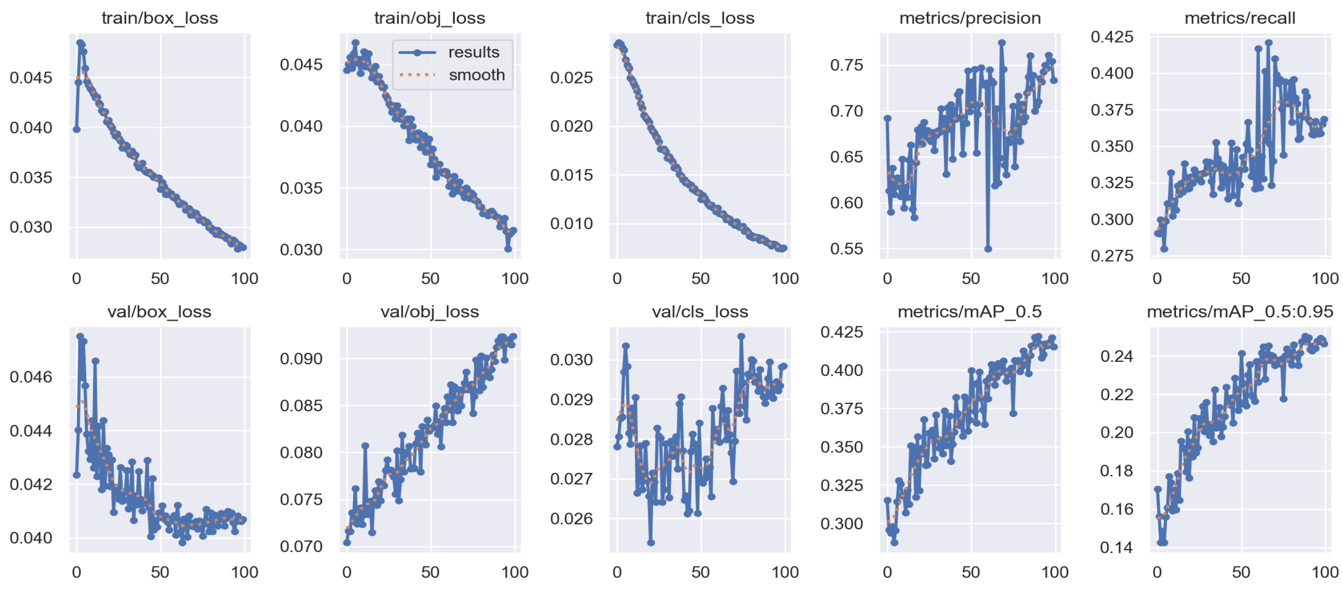


Fig. 9. Result of Training Best Tuning Hyperparameter YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images

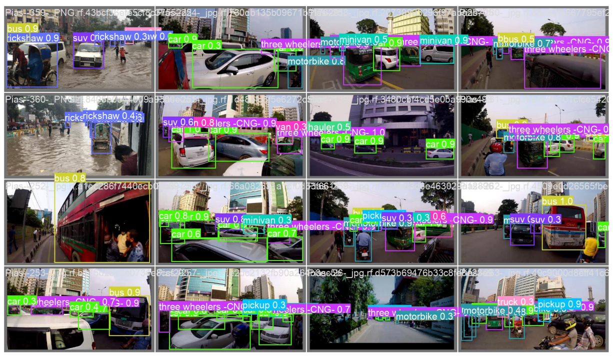


Fig. 10. Result of YOLO V5 Convolutional Neural Network Predicts Smart Traffic Management Vehicles in Smart Traffic Management Images

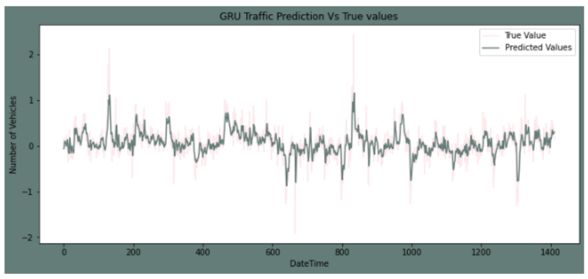


Fig. 11. Result of Recurrent Neural Network Long Short Term Memory Predicst Smart Traffic Management Vehicles Next 12 Hours

using Artificial Intelligence can increase Traffic Management Congestion Flow Rate with 50 % Traffic Management Congestion Flow Rate and decrease Traffic Management Vehicles Pass Delay with 70 % Traffic Management Vehicles Pass Delay. In the Simulation of Smart Traffic Management System in

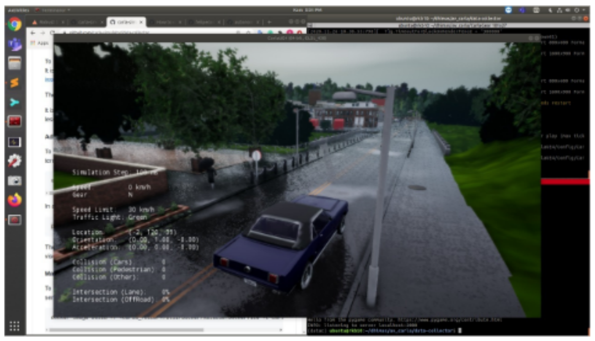


Fig. 12. Result of Evaluation Smart Traffic Management System using Artificial Intelligence using Traffic Management Simulator CARLA

Traffic Management Simulator CARLA We Can Have Smart Traffic Management System using Artificial Intelligence can increase Traffic Management Congestion Rate significantly and decrease Traffic Management Vehicles Pass Delay significantly when the number of Smart Traffic Management System Vehicles is more than 10 Smart Traffic Management System Vehicles.

III. Conclusion

This paper presents Autonomous Smart Traffic Management System using Artificial Intelligence to increase Traffic Management congestion flow rate. Autonomous Smart Traffic Managements System used YOLO V5 Convolutional Neural Network to detects Smart Traffic Management Vehicles in Smart Traffic Management Images. Autonomous Smart Traffic Management System tuning Hyperparameter of Learning Rate Start , Momentum YOLO V5 , Mosaic Parameter YOLO V5 , Rotation Degree Parameter YOLO V5 , Scale Parameter

YOLO V5 and Weight Decay Parameter YOLO V5. From the Result Table of Tuning Hyperparameter YOLO V5 Convolutional Neural Network to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images We Can See the best Tuning Hyperparameter are Tuning Hyperparameter Search 1 with Parameter Learning Rate 0 0.0052 , Parameter Momentum 0.595 , Parameter Mosaic 0.36 , Parameter Degrees 72 , Parameter Scale 1.32 , and Parameter Weight Decay 0.0005 with Mean Average Precision 0.4034 that is Almost 50 % more higher than other Tuning Hyperparameter YOLO V5 Convolutional Neural Network to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images (Around 0.2852). From Training YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter to Detect Smart Traffic Management Vehicles in Smart Traffic Management Images We Can Have YOLO V5 Convolutional Neural Network Best Tuning Hyperparameter can detect Smart Traffic Management Vehicles in Smart Traffic Management Images with Mean Average Precision 0.88561 and Mean Average Precision for Predicting Smart Traffic Management Cars in Smart Traffic Management Images is 0.95251.

Then Autonomous Smart Traffic Management predicts Smart Traffic Management Vehicles next 12 hours using Recurrent Neural Network Long Short Term Memories. Writer training Recurrent Neural Network Long Short Term Memory to predict Smart Traffic Management Vehicles in next 12 hours using Recurrent Neural Network Long Short Term Memory using Number of Traffic Management Vehicles last hour , Year of Traffic Management Vehicles , Month of Traffic Management Vehicles , Day of Traffic Management Vehicles , Hour of Traffic Management Vehicles and Minutes of Traffic Management Vehicles. From Result of Recurrent Neural Network Long Short Term Memory to Predict Smart Traffic Management Vehicles in Next 12 Hours We Can see that Recurrent Neural Network Long Short Term Memory Can predict Smart Traffic Management Vehicles in Next 12 Hours with Mean Squarred Error 4.521 Vehicles and Root Mean Squared Error (RMSE) 2.232 Vehicles. From Result of Recurrent Neural Network Long Short Term Memory to predict Smart Traffic Management Vehicles in Next 12 Hours We Can See that Recurrent Neural Network Long Short Term Memory to Predict Smart Traffic Management Vehicles in Next 12 Hours can predict increase and decrease of Smart Traffic Management Vehicles in next 12 hours.

Then Autonomous Smart Traffic Management evaluates Smart Traffic Management System using Artificial Intelligence using Traffic Management Simulator CARLA. After Simulation of Smart Traffic Management System in Traffic Management System Simulation CARLA We got Traffic Management Congestion Flow Rate with Traffic Management System (21 Vehicles / Minutes) is 50 % Higher than Traffic Management Congestion Rate without Smart Traffic Management System (Around 15 Vehicles / Minute) and Traffic Management Vehicles Pass Delay with Smart Traffic Management System (5 Seconds / Vehicles) are 70 % Less Than Traffic Management Vehicles Pass Delay without Smart Traffic Management System (Around 12 Seconds / Vehicles). Then We Can See that Smart Traffic Management System using Artificial Intelligence can increase Traffic Management Congestion Flow Rate with 50 % Traffic Management Congestion Flow Rate and decrease Traffic Management Vehicles Pass Delay with 70 % Traffic Management Vehicles Pass Delay. In the Simulation of Smart Traffic Management System in Traffic Management Simulator CARLA We Can Have Smart Traffic Management System using Artificial Intelligence can increase Traffic Management Congestion Rate significantly and decrease Traffic Management Vehicles Pass Delay significantly when the number of Smart Traffic Management System Vehicles is more than 10 Smart Traffic Management System Vehicles.

Then We can see Autonomous Smart Traffic Management System can predicts number of Smart Traffic Management Vehicles next 12 hours with Mean Squarred Error 4.521 Vehicles and Root Mean Squared Error (RMSE) 2.232 Vehicles. Autonomous Smart Traffic Management System can Traffic Management Congestion Flow Rate with Traffic Management System (21 Vehicles / Minutes) is 50 % Higher than Traffic Management Congestion Rate without Smart Traffic Management System (Around 15 Vehicles / Minute) and Traffic Management Vehicles Pass Delay with Smart Traffic Management System (5 Seconds / Vehicles) are 70 % Less Than Traffic Management Vehicles Pass Delay without Smart Traffic Management System (Around 12 Seconds / Vehicles). Then We Can See that Smart Traffic Management System using Artificial Intelligence can increase Traffic Management Congestion Flow Rate with 50 % Traffic Management Congestion Flow Rate and decrease Traffic Management Vehicles Pass Delay with 70 % Traffic Management Vehicles Pass Delay. In the Simulation of Smart Traffic Management System in Traffic Management Simulator CARLA We Can Have Smart Traffic Management System using Artificial Intelligence can increase Traffic Management Congestion Rate significantly and decrease Traffic Management Vehicles Pass Delay significantly when the number of Smart Traffic Management System Vehicles is more than 10 Smart Traffic Management System Vehicles.

ACKNOWLEDGMENT

The author would like to thank you the Korean Advanced Institute of Science and Technology (KAIST) who hosted the Autonomous Vehicles and Smart Traffic Mobility Program, and the whole team of Fundamental of Vehicular Electric System program who relentlessly strive to conduct this Fundamental of Vehicular Electric System Final Project in this desolate pandemic environment. Special thanks also for Prof. Syeung Heun Kong as the head of Autonomous Vehicle Lab who lead, mentoring, advised and motivated the author during the Autonomous Smart Traffic Management System Research program and give invaluable support. Sincere thanks also for Cho Sang Jae and Paek Dong Hee as the mentor of author who always there to help and motivate the author in the Smart Traffic Management System Research , and for Ryan

Gallagher as teammate of author who helped author from the beginning either in the topic or in other activity.

References

[1] Linux Foundation *Acumos AI : Open Source Artificial Intelligence Platform* Linux Foundation Publication , 2020.

[2] Aurelion Geron, *Hands- On Machine Learning with Scikit- Learn Tensorflow*, 1st ed.California, America: O'Reilly Media, 2017.

[3] Rockwell Anyoha *The History of Artificial Intelligence* Harvard University : Science in The News, 2017.

[4] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, *"Dermatologist-level classification of skin cancer with deep neural networks* .Nature, vol. 542, no. 7639, p. 115, 2017.

[5] Y. Wu, M. Schuster *et al*, *Google's neural machine translation system: Bridging the gap between human and machine translation* arXiv preprint arXiv:1609.08144, 2016.

[6] J. Schmidhuber, *Deep learning in neural networks: An overview* Neural networks, vol. 61, pp. 85–117, 2015.

[7] C. Wang, S. Zhao, *et al*, *Webpage depth viewability prediction using deep sequential neural networks* IEEE Transactions on Knowledge and Data Engineering, 2018.

[8] D. Sculley, G. Holt, *et al*, *Hidden technical debt in machine learning systems*, Advances in Neural Information Processing Systems, 2015, pp. 2503–2511.

[9] Shuai Zhao, Manoop Talasila *et al*, *Packaging and Sharing Machine Learning Models via the Acumos AI Open Platform*, New Jersey, USA : AT&T Research Labs, 2019.

[10] M. Vartak, H. Subramanyam *et al*, *Model db: A System for Machine Learning Model Management*, in Proceedings of the Workshop on Human-In-theLoop Data Analytics. ACM, 2016.

[11] ATT and Linux Foundation , *Acumos AI: An Open Source Ai Machine Learning Platform*, Available: https://www.acumos.org/ , 2017.

[12] Crawshaw, James, *Democratizing AI : How Acumos AI Fast Track Your AI Development*, Tech Mahendara , Available: https://cache.techmahindra.com/static/img/pdf/throught-leadership/thought-leadership-how-acumos.pdf , 2018.

[13] Crawshaw, James, *Acumos AI Project Adds Smarts With Boreas*, Available: https://www.lightreading.com/artificial-intelligence-machine-learning/acumos-ai-project-adds-smarts-with-boreas/a/d-id/752359 , 2019.

[14] O-RAN Alliance, *O-RAN Working Group 2 : Non-RT RIC and AI Interface*, 2nd ver.2019.

[15] Linux Foundation, *Acumos AI Boreas Releases Notes*, 2019.

[16] C. Olston, N. Fiedel , *et al*, *Tensorflow-serving: Flexible, highperformance ml serving*, arXiv preprint arXiv:1712.06139, 2017.

[17] D. Merkel, *et al*, *Docker: Lightweight Linux Containers for Consistent Development and Deployment*, Linux Journal, vol. 2014, no. 239, 2014.

[18] Open Neural Network Exchange Community, *ONNX Release Notes*, 2020.

[19] Data Mining Group , *Portable Format Analytics (PFA) : What is PFA?*, 2020.

[20] F. Pedregosa, G. Varoquaux, *et al*, *Scikit-Learn: Machine Learning in Python*, Journal of Machine Learning Research, Vol. 12, No. Oct, pp. 2825–2830, 2011.

[21] M. Abadi, P. Barham, *et al*, *Tensorflow: A System For Large-Scale Machine Learning*, in OSDI, vol. 16, pp. 265–283, 2016.

[22] A. Paszke , S. Gross, *et al*, *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, Canada : 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), 2019.

[23] X. Meng, J. Bradley, *et al*, *Mllib: Machine learning in apache spark*, The Journal of Machine Learning Research, vol. 17, no. 1, pp. 1235–1241, 2016.

[24] M. Hofmann and R. Klinkenberg , *RapidMiner: Data mining use cases and business analytics applications*, CRC Press, 2013.

[25] C. Olston, N. Fiedel , *et al*, *Tensorflow-Serving: Flexible, High Performance ML Serving*, arXiv preprint arXiv:1712.06139, 2017.

[26] D. Crankshaw, X. Wang *et al*, *Clipper: A Low-Latency Online Prediction Serving System*, in NSDI, pp. 613–627 , 2017.

[27] R. A. McDougal, T. M. Morse *et al*, *Twenty Years of Modeldb and Beyond: Building Essential Modeling Tools for the Future of Neuroscience*, Journal of computational neuroscience, vol. 42, no. 1, pp. 1–10, 2017.

[28] J. N. Van Rijn, B. Bischl *et al*, *Openml: A Collaborative Science Platform*, Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, 2013, pp. 645–649

[29] M. Ribeiro, K. Grolinger *et al*, *Mlaas: Machine Learning As A Service*, in Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on. IEEE, 2015, pp. 896–902 , 2015.

[30] O-RAN Alliance, *O-RAN Working Group 3 : Near-RT RIC and E2 Interface*, 2nd ver.2019.

[31] M. Talasila , G. Jacobson, *O-RAN With AI/ML Flow*, October 2019 Costa Rica F2F Meeting , 2019.

[32] M. Skorupski , J. Keeney, *Non-RT RIC Notes*, Confluence : Wiki O-RAN , 2020.

[33] Linux Foundation, *ONAP: Architecture Overview*, Open Network Automation Platform (ONAP) Architecture White Paper , 2018.

[34] Linux Foundation, *Acumos AI All-in-One (AIO) User Guide*, 2020.