UNIT SIT790 - MAJOR THESIS

LEARNING FEATURES EXTRACTION TECHNIQUES TO ENCODE COMPLEX NUSCENES DATASET

STUDENT NAME: SHRUTIK PANCHAL
STUDENT ID: 218412482
SUPERVISOR: GUANGYAN HUANG

WORD COUNT: 8019

Table of Contents

Abstract:

Robust object classification, tracking and segmentation carry a crucial and a significant role in future technological advancements specifically in autonomous vehicles technologies. The success of these technologies solely depends upon accurately classifying diverse weather and traffic conditions using different Artificial Intelligence (AI) methods such as Deep Neural Network (DNN). Majority of autonomous vehicles are equipped with variety of cameras, trackers and sensors; providing combination of inputs to detect and track the surroundings of vehicles hence it is mandatory to develop machine learning methods that perform best with multimodal input data. This study surveys available methods to extract important features from images, lidar and radar data; and ways to apply fusion on the extracted features with various sensors modalities. This study uses first publicly available multimodal dataset 'nuScenes' containing combined data of 1 spinning LIDAR, 6 cameras and 5 radars. Further, it focuses on extracting important features from lidar and radars data using Open3D and combines them with features extracted from image data using max pooling; and evaluates the performance of the deep neural network with extracted features. Various neural network models are designed to work with specific types of data such as image data, pointcloud data and other continuous data; and further, these models were combined with concatenation layer to carry forward learning from individual models. All the evaluation has been carried out with various randomly generated test sets and combined outperforms all the individual models with average accuracy over sixty percent.

Keywords: Neural Network, nuScenes, multimodal, autonomous vehicles, LiDAR, Radar, PointNet.

Introduction:

Autonomous vehicles, a revolution in transport and logistic industry, have potential to reduce congestion and emission while providing various benefits such as enhancing safety, driver productivity and road utilization; enabling efficient transportation and logistics services [1], [2]. A triumph in success of such vehicles is in accurately detecting various traffic and weather conditions in real vivid environments and situations such as locating and recognizing obstacles (for example: traffic lights, pedestrians, cross-ways, surrounding vehicles, cyclists and others), particularly, in urban environment [3]–[5]. The previous development in autonomous vehicles was largely based on traditional Artificial Intelligence concepts such as hierarchical image processing, probabilistic/automatic reasoning and others however, in current era, there is a large shift towards neural network, specifically, because of their deep configuration [6]. DNN helps to provide high quality robust solutions to many challenging computational areas such as computer vision, language modelling, data and speech analytics, natural language processing, ocean engineering and others [6]–[8] with ability to provide high prediction accuracy and automatically extracting real-time learning parameters and data features [9], hence, specifically, CNN models are largely accepted in autonomous vehicles industry providing realistic image and video classification [1]–[3], [5]–[8], [10].

Related work:

The paper [11] discusses performances of PointPillars (PointCloud) and MonoDIS methods to detect 3D objects with multimodal dataset (nuScenes) based on lidar, image and object tracking baselines where PointPillars outperforms MonoDIS with more than 55% accuracy in detection of common classes such as cars, pedestrians and others whereas MonoDIS, on other-end, classifies smaller classes such as traffic cones and bicycles with more than 45% and 25% respectively. The conclusion states that small/thin objects return few/less lidar points, traffic cones are small hence ignored in pointcloud whereas detected in images. The author(s) have proposed nuTonomy scenes (nuScenes) multimodal dataset with full 360-degree field of view data captured with autonomous vehicles using sensor suite of 5 radars, 6 cameras and 1 lidar. Similarly, the paper [5] discusses performance of PointPillars over other state of the art techniques including fusion methods to detect objects in both BEV (Bird Eye View) and 3D view with KITTI benchmarks where PointPillars, only using lidar data, outperforms various encoders with accuracy and speed resulting PointPillars being the best suited encoding method in PointCloud for object detection. On another-end, the paper [12] discusses about neural network performance on sensors (radars and lidars), least explored areas, collected data that are mounted on autonomous vehicles where the authors have utilized the image data with sensors data during the training of the neural networks resulting in promising results classification with test data (with or without image data). In the paper [10], the author(s) have stated that CNN has significant role in object detection, key-point detection and instance segmentations however traditional ways to process images cannot be applied to sensors data directly as 3D data are down sampled to 2D image data without extracting features from z-axis (important features loss) resulting in 90% average precision (AP) in 2D car detection over only 15% AP in 3D car image-based detection as majority of previous work has been done with unimodal (from one sensor or one dimensional) or bimodal (from combination of two sensors or two dimensional) datasets using traditional image processing techniques to extract features however important

features are being ignored in earlier stages with these techniques resulting lower classification performance in AI methods.

Background:

This section provides background overview/analysis of deep multimodal techniques for autonomous driving including summary of various multimodal datasets and testing vehicles, sensors and their sensing modalities, deep object detection and semantic segmentation; and some methodologies to extract features and fusion of those extracted features focused on camera images, lidar and radar sensors data. Furthermore, based on observed literatures, challenges and open questions are discussed in context of multimodal data preparations and fusion methods.

Autonomous vehicles sensing modalities:

I. Cameras:

Images and video clips captured by cameras such as visual and thermal helps to detect nearby objects including their speed and distance hence enabling autonomous vehicles to fabricate accurate real-world visual representation of surroundings with in-depth texture information. Unlike visual cameras, thermal cameras are more robust in variety of sensitive lightening and weather conditions such as snow, rain, night-time and fog as they identify and discover infrared heat radiations coming from objects. However, cameras are unable to provide object depth information [13] whereas [14] able to extract additional sensing details such as optical flow and other multi-spectral images [15], [16].

II. Light detection and ranging (LiDARs):

LiDAR utilizes reflections of invisible laser light at eye-safe levels with a certain frequency in calculation of distance and elevation to objects [17]; providing robust and accurate depth information in various lightening and weather conditions including fog, rain, snow and others up to 100 meters distance than visual and thermal cameras as they are unable to capture objects with fine textures and increase in sparseness with the increase of object distance. Recent development of LiDAR such as flash and FMCW (Frequency Modulated Continuous Wave) produces detailed object information with velocity [13] including three-dimensional environment creation using a point cloud.

III. Radio detection and ranging (Radars):

Radar collects emitted radio waves that are reflected by objects/ obstacles enabling measurement of signal runtime and object velocity estimation utilising the Doppler effect. It provides robust object information in variety of lighting and weather conditions however due to low resolutions, it is very challenging to be used in object classifications [13]. They are best employed in traffic jam assistance system and adaptive cruise control systems [18].

IV. Other sensors:

Other types of sensors are [13]:

1) Ultrasonics:
- Object distance measurement with high-frequency sound waves
- Employed in near-range and low velocity object detection
- Affected by vivid environment conditions such as temperature, humidity and others [18]

2) Global Navigation Satellite System (GNSS):
- Accurate three-dimensional object location
- Introduced to help in automotive navigation
- Currently utilized in ego-vehicle localization and path planning

3) Inertial Measurement Units (IMU) and odometers:
- Used to extract vehicle internal information
- Measurement of vehicle accelerations, rotational rates and odometry

Multimodal datasets and testing vehicles:

Deep multimodal recognition methods require labelled ground-truth datasets to compare and test modal accuracy. Majority of available multimodal datasets include RGB camera images with LiDAR point cloud [19], [20] and thermal images [21]. The Multispectral KAIST dataset includes LiDAR data and thermal image data [22]. Recently available multimodal datasets nuScenes [11], HiRes2019 [23] and Oxford RobotCar [24] comes with combination of more than two sensor suits such as multiple cameras, LiDARs and Radars. Large number of multimodal datasets that are available and being used in autonomous driving research are recorded low variety of weather and traffic conditions such as KITTI dataset is recorded in Karlsruhe, Germany and only during daytime [25]. Similarly, the Oxford RobotCar dataset [24] is collected by driving vehicle in Oxford area for a year hence includes data in various weather and lighting conditions. The Eurocity is among the most diverse dataset that is available today and includes cameras and LiDARs information recorded while driving in various European countries during variety of seasons [26]. The largest multimodal dataset that is now available with ground-truth labels is nuScenes dataset with over 1M frames however the number of multimodal dataset is relatively low in computer vision community [11].

Autonomous testing vehicles are normally equipped with various sensors such as cameras, Ultrasonics, LiDARs, odometers, Radars and others. Some examples are: Autonomous vehicle “Boss” (developed by Tartan Racing Team): Contains cameras, LiDARs and Radars, Google Waymo, Mercedes Benz S-class, Uber, GM Cruise, Baidu Apollo and many others [13].

Deep object detection and semantic segmentation:

Deep object detection is a part of many computer vision tasks where recognition and localization of multiple objects in vision or scene takes place by estimating classification probabilities and localizing with bounding boxes; and now considered as a benchmark for many popular object detection tasks [13] widely utilized in autonomous vehicles to detect various object such as traffic lights, people, road signs, vehicles and others. In extension to this, deep semantic segmentation/ object instance segmentation helps to highlight the pixels of recognized objects instead of bounding boxes hence helping in partitioning a scene in meaningful parts where it can be pixel-level, instance-level or panoptic-level (unified: pixel and instance level) [27] semantic segmentation enabling them to be centre of attention in autonomous vehicles industry [13].

Deep object detection has two classic state-of-the-art approaches: one-stage and two-stage object detection. Two-stage approach outperforms one-stage methods with better accuracy because of refinement paradigm and region proposal generation while costing network training time whereas one-stage methods are easier and faster to optimise [13], [28]. Similar to deep object detection, object sematic segmentation can be categorised in to two methods: one-stage and two-stage pipeline. One-stage pipeline is common/ traditional approach as based on hierarchical processing whereas two-stage pipeline generates region proposals followed by fine-tuning them for instance-level segmentation [29], [30].

Methodology:

In this section, various available methods and techniques have been discussed to work with multimodal dataset.

1) Features extraction:

a) LiDAR Point Clouds:

An extraction of features from LiDAR point clouds could provide reflectance and depth of objects nearby autonomous vehicles where reflectance information can be collected by intensity and depth can be encoded by Cartesian coordinates, density, horizontal disparity, height, angle, distance or any other three-dimensional coordinate system. It is possible to gather information from LiDAR point clouds mainly by three ways.

The rich three-dimensional shape and environment information can be gathered and preserved by discretising three-dimensional space into three-dimensional voxels and attaching points to voxels [31]. Furthermore, [32] proposed voxel features encoding (VFE) works with 3D object detection by converting sparse voxels to two-dimensional however contains only proven one-stage extraction hence saves computational costs while compromising on features loss. The method results in many empty voxels as LiDAR data are irregular and sparse and it is fast but possible higher features loss.

Another way is to project three-dimensional point cloud data to two-dimensional grid-based feature maps hence can be feed to two-dimensional convolutional layers. Spherical map represents point cloud data in dense and

compact way however representation size will vary from input images hence hard to fuse. The camera-plane map (CPM) is generated by projecting three-dimensional points in camera coordinate system and can be directly fused with camera images as sizes are same however leaves many empty pixels. Both the methods project LiDAR data to front-view plane and in contrast to this, bird-eye view (BEV) preserves additional information such as object length and with hence avoids occlusion and provides specific object location details making BEV widely accepted in three-dimensional perception with PointNet [13].

The last method is to skip voxelization and directly learn from three-dimensional LiDAR point clouds. PointNet [33] and PointNet++ [34] learn from individual feature points and combine/aggregate gathered features with max pooling and in addition to this, [35] proposed aggregation of points with a weighted sum. Hence, PointNet architecture with BEV makes the best candidate to work with LiDAR point cloud data.

b) Camera images:

Majority of literature works with RGB images gathered from various types of cameras and extracting sensing information such as rich texture information, depth, optical flow and others however based on cameras, objects might scale differently and can be occluded hence usage of bird-eye view architecture might result in better representation [13]. In order to project camera image features onto BEV plane map, [36] proposed OFT (Orthographic Feature Transformation) where gathered maps are further processed for three-dimensional object detection. Using the similar idea, [37] proposed to convert images into pseudo-lidar representation utilizing depth information and further processing with state-of-the-art LiDAR BEV detectors resulting in improvement in detection.

c) Radar data:

Radars have higher ranges and amplitudes hence provide rich information of environment and can be employed to accurately detect distant vehicles by representing gathered data into feature maps. Radar data maps can be generated by accumulating gathered Radar data during various time-stamps in static object semantic segmentation [37] however in autonomous driving, it is required to be dynamic as decision need to be made in low timespan and vivid environments. As representation of Radar data can be done as point cloud, PointNet++/PointNet is the perfect match to process these data for dynamic object segmentation [34], [38].

2) Features fusion:

The paper [13] states in-depth analysis on fusion of features gathered from various sensors suites. It addresses what sensors data need to be fused, how it can be fused together and when in network this fusion is possible. It states various mathematical features transformation that need to be applied before fusing the extracted features such as average mean, concatenation, ensemble, mixture of experts and others. As features representation is hierarchical, DNN provides various possibilities to fuse these features into network such as early stage, middle stage or later stage however there is no evidence stating that one stage fusion is better than the other stages. Early fusion works with pre-processed or raw sensors data, late fusion combines the output of each sensors feature extraction and middle fusion works in-between of early and late fusion where it combines various sensors data at intermediate layers however it is flexible, yet it is hard to find optimal way to fuse gathered results. Modern multimodal networks are either one-stage pipeline or two-stage pipeline allowing them to have multiple ways to do network fusion.

Open questions and challenges:

As discussed in previous sections, advancements in future of autonomous vehicles technologies depends on robust object detection, tracking and segmentation of surrounding environment hence development in deep multimodal perception system plays a crucial role as predictions are transferred to other modules in decision making. The current challenges and open questions are as follows:

In context of multimodal data preparation:

Challenges:

- ✓ Limited variety in sensors, driving and weather conditions
- ✓ Small amount of training data, object class imbalance and labelling errors
- ✓ Misalignment of various sensors

Open questions:

- ✓ Development of realistic virtual datasets; and optimal way to combine real and virtual dataset
- ✓ Improvement in labelling and robust networks over noisy data and labels

In context of fusion methods:

Challenges:

- ✓ Limited sensors data have been successfully fused without losing important features and lack of studies for various features representations
- ✓ Lack of global measurements in each sensor channel and simple fusion operations
- ✓ No guideline to achieve optimal fusions
- ✓ Lack of memory or robustness trade-offs

Open questions:

- ✓ Fusion of various sensors with similar modality
- ✓ Fusion of least explored sensor modalities such as ultrasonics, radars and others
- ✓ Comparison of various features representations; and uncertainty estimations and propagating them to other modules
- ✓ Enabling network pruning and compression including anomaly detection and optimal fusion architecture using visual analytics

Apart from mentioned above, current evaluation metrics are focused on comparison of network accuracy hence need to be modified to test robustness of networks whereas current networks can't guarantee consistency over time and are designed for modular autonomous driving [13].

The further work in this research will be focused on the following points:

1) Extracting required information from vast nuScenes dataset (as development kit has variety of information in collection) (Weightage 20%)
2) Features extraction from LiDAR and RADAR pointcloud (Weightage 20%)
    a. Features extractions from individual sensors data
    b. Aggregating/combining both sensors data either max pooling or weighted sum
3) Features extraction from images with montage image representation (Weightage 25%)
4) Combining individual models for better detection performance (Weightage 25%)
5) Evaluation of deep neural networks over various test sets (Weightage 10%)

Research experiments and performance evaluations:

This section provides in-depth information about dataset, data pre-processing methods used, created custom neural network models from scratch and performance of models over various randomly generated test sets.

Dataset:

The current study assesses recently available multimodal nuScenes dataset [11] developed and collected by Aptiv Autonomous Mobility (nuTonomy), with autonomous vehicles in Boston (Seaport) and Singapore (Queenstown, Holland Village and One North) from highly challenging and dense traffic conditions. It contains 23 object classes such as animal, pedestrian, movable objects, vehicle and others with tracking attributes of respective classes such as moving, stopped, parked, standing and others. There are manually annotated 1000 scenes in dataset with 20 seconds duration each resulting more than 1 million records however this research will focus development on the mini version of the same dataset with limited number of samples. The following image "Image m1" shows information about the selected mini version of the dataset where it is visible that dataset has 23 categories ("Table m1"): human pedestrian (adult/ child), human pedestrian (construction worker), human pedestrian (personal mobility), human pedestrian (police officer), movable object (barrier), movable object (debris), movable object (push-able/ pull-able), movable object (traffic cone), static object (bicycle rack), vehicle (bicycle), vehicle bus (bendy/ rigid), vehicle (car/ motorcycle) and vehicle (construction/ trailer/ truck). In addition to these, the classes contain attributes such as: vehicles (moving, stopped, parked), cycle (with rider/ without rider) and pedestrian (sitting/ lying down/ standing/ moving). There are 404 samples in this mini-version of dataset where it contains 18538 annotation in these 404 samples. The categories and their annotations counts are shown in "Table m1" below. It is visible that there are some categories/ classes that contains large number of annotations samples in this dataset (adult), movable object vehicles (car). Apart from mobility, child, police (debris), vehicle bus bendy contains few numbers of possible that these classes the training of neural this, data annotations are visibility from the ego near to far (80 to 100 percent visibility, 40 to 60 lastly, 0 to 40 percent provides taxonomy of with their description. 'sample_data' and dataset mainly being 'sample_data' table about lidar, radar version such as humans (barrier, traffic cone) and these, human (personal officer), movable objects and vehicle trailer annotations hence it is might get ignored during network. In addition to categorized by their vehicle into 4 bins: from percent visibility, 60 to 80 percent visibility and visibility). Also, dataset object category classes There are two tables 'sample_annotation' from used in this research. As provides information

```
======
Loading NuScenes tables for version v1.0-mini...
23 category,
8 attribute,
4 visibility,
911 instance,
12 sensor,
120 calibrated_sensor,
31206 ego_pose,
8 log,
10 scene,
404 sample,
31206 sample_data,
18538 sample_annotation,
4 map,
Done loading in 7.2 seconds.
======
Reverse indexing ...
Done reverse indexing in 0.1 seconds.
======
```

Image m 1

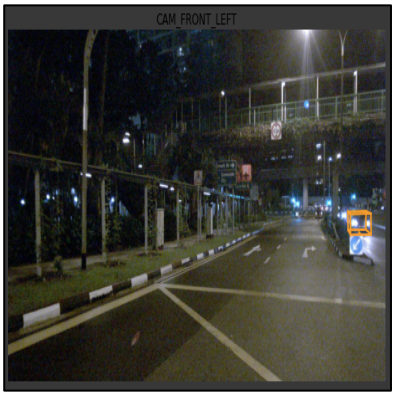
Categories	Counts
human.pedestrian.adult	4765
human.pedestrian.child	46
human.pedestrian.construction_worker	193
human.pedestrian.personal_mobility	25
human.pedestrian.police_officer	11
movable_object.barrier	2323
movable_object.debris	13
movable_object.pushable_pullable	82
movable_object.trafficcone	1378
static_object.bicycle_rack	54
vehicle.bicycle	243
vehicle.bus.bendy	57
vehicle.bus.rigid	353
vehicle.car	7619
vehicle.construction	196
vehicle.motorcycle	471
vehicle.trailer	60
vehicle.truck	649
Grand Total	**18538**

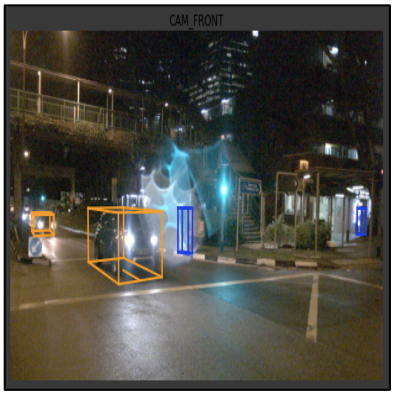
Table m2

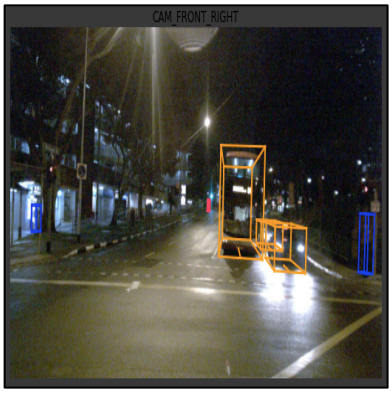
and image data associated with respective samples, timestamp, is data keyframe or not and others. Similarly, 'sample_annotation' provides information specific to each annotation in sample such as visibility, number of lidar points, number of radar points, rotation, size and translation. Table m2 shows information about samples associated with each scene in the mini dataset. There are 10 scenes in this version of dataset, containing approximately 40 samples in each scene. The dataset compromises of data captured from 12 sensors and the sensors are: lidar top, radar left front, radar front, radar right front, radar right back, radar right left, camera front left, camera front, camera front right, camera back right, camera back and camera back left. Following are some examples from one sample in dataset (showing information captured from various sensors).

Scenes	Samples
scene-0061	39
scene-0103	40
scene-0553	41
scene-0655	41
scene-0757	41
scene-0796	40
scene-0916	41
scene-1077	41
scene-1094	40
scene-1100	40

Table m2







Data preprocessing:

1. Object class generalization and encoding:

As mentioned in "Table m1", there are 23 object classes in this mini-version of 'NuScenes' dataset however there are many object sub-classes contain only handful of annotated samples hence there is a need to generalise the classes into high-level classes.

As shown in the 'image m2', classes human pedestrian adult, child, construction worker, personal mobility and police officers are combined into one and generated generalized category of "pedestrian". Similarly, movable objects such as barriers, debris, push-able, pull-able, traffic cone and bicycle rack are generalized into 'small objects' category.

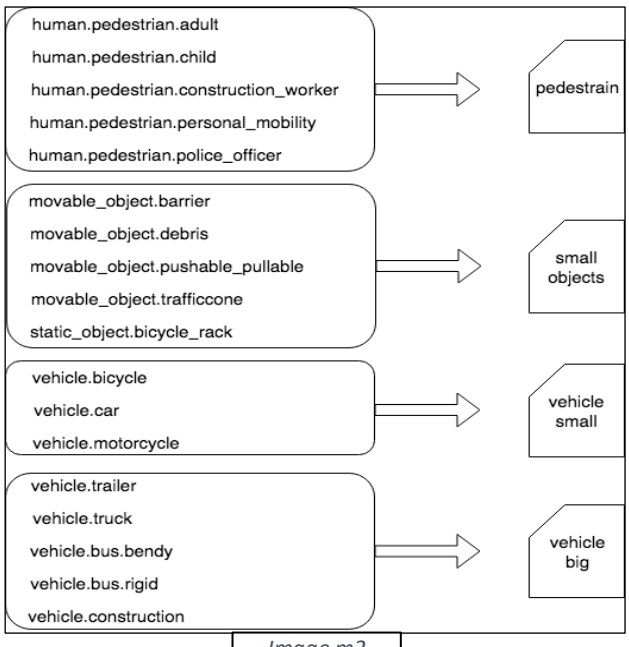


Image m2

Labels	Counts
pedestrian	5040
small_objects	3850
vehicle_big	1315
vehicle_small	8333

Table m3

On the other-end, vehicles category generalized into two sub-categories: vehicles small and vehicles big. 'vehicle small' category consists of bicycle, car and motorcycle annotations and 'vehicle big' category contains trailer, truck, bus (bendy and rigid) and construction vehicles. The 'Table m3' shows sample counts of each generalized categories where pedestrian contains 5040, small objects contains 3850, vehicle big has 1315 and vehicle small has 8333 samples.

After generalizing the labels, Label Encoder from sklearn-preprocessing has been used to encode target labels into values between 0 to number of classes-1. It helps to normalize class variables between o to number of classes hence

helping in writing efficient scripts and routine and in addition, it can also transform non-numerical labels to numerical labels however they should be comparable and hashable.

2. Image data:

As mentioned in dataset description, this dataset contains input data from various sensors such as camera, lidar and radar. The dataset contains data from six camera sensors providing each image of size width: 1600px by height: 900px. Hence, processing 1600px X 900px = 1440000 pixel per annotation is really big task as each annotation contains 6 images. To overcome this stepping stone, this research focuses on processing montage image per annotation where montage image is created using 6 images captured from 6 camera sensors which are: camera front left, camera front, camera front right, camera back right, camera back and camera back left. While creation of montage image, each camera image is resized to 128px X 108px and staked into 384px X 216px output image along with other 5 images following the same resize techniques. Here, output image has two rows first row contains resized images from camera front left, camera front and camera front right sensors starting from top left of output image to top right of output image. Similarly, second row contains resized images from camera back left, camera back and camera back right starting from bottom left of output image to bottom right. Following graph ('Image m3') shows the whole montage image creation process from individual sensors images with output image size of 384px X 216px. In the image, it is visible that all the six images from 1 to 6 are stacked in sequence from 1 to 3 in row 1 and images 4 to 6 are stacked in row 2. Final output montage image is of size 384px and 216px. Following are the examples of montage images:

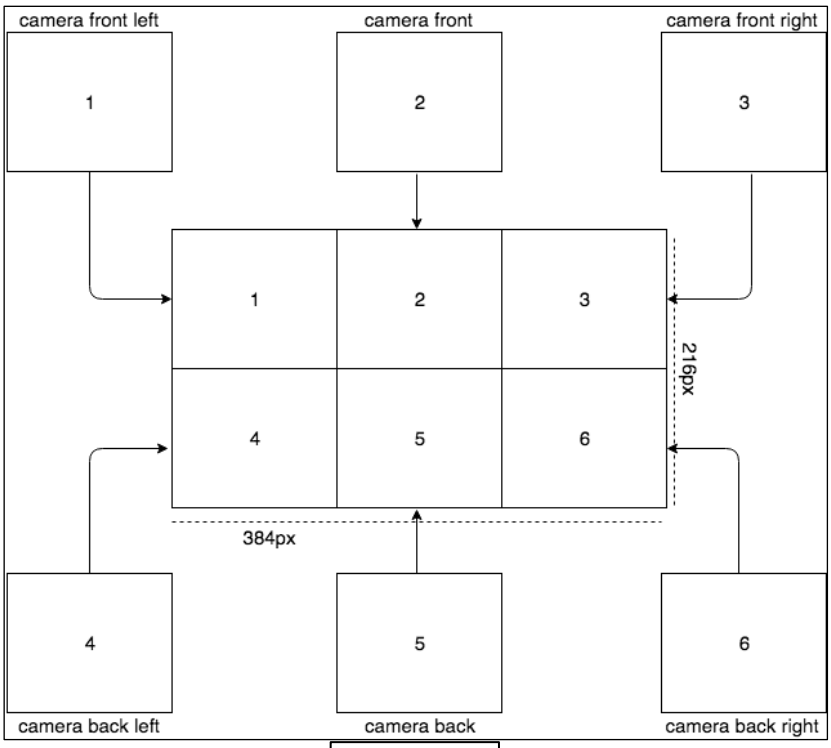


Image m3

For each sample/annotation, open cv libraries are being used to read images from the directories and helps them to create numpy matrix while resizing images to 128px X 128px and extracting features by employing SIFT – Scale Invariant Feature Transform to extract key points and computation of descriptors.

3. Point cloud and continuous data:

As mentioned in introduction of the dataset, it provides data from various sensors suits hence we end up with 6 pointcloud data and some other continuous variable data such as number of lidar points and number of radar points in annotations; and visibility of the objects from autonomous/ego vehicle while capturing samples in real-world. The six pointcloud data come from five radar sensors and one lidar sensor where number of lidar points reflects the number of points returned from the object in one lidar sweep and in similar way, number of radar points reflects the number of points returned from object in one radar sweep.

To extract useful features from lidar and radar data, Open3D packages have been utilized where first after reading the pointcloud data from received respective files (lidar or radar) associated with each sample, pointcloud data gets down sampled and returns the less dense pointcloud representation. With the help of Open3D packages, from down sampled pointcloud representation, axis wise normal are being derived and then these derived data along with number of lidar and radar points and visibility rate are stored in numpy array for further encoding. From sklearn scientific package, MinMaxScaler has been utilized to normalize gathered pointcloud and continuous data resulting values from 0 to 1.

Neural network models:

The whole research contains various sensors data that need to be processed to gather features, insights and make predictions hence there are separate models that have been designed to take care of such data. The research does not use any available pretrained models hence all the models used in this paper are custom designed for the methods used in this paper. The models based on respective data type are as follows:

1. Image data processing model:

This model focuses on image data processing including features extractions from input image, model training and validation with the use of train (80 percent) and test (20 percent) splits from sklearn; and object detection performance. Following diagram gives general idea of model structure and depth:

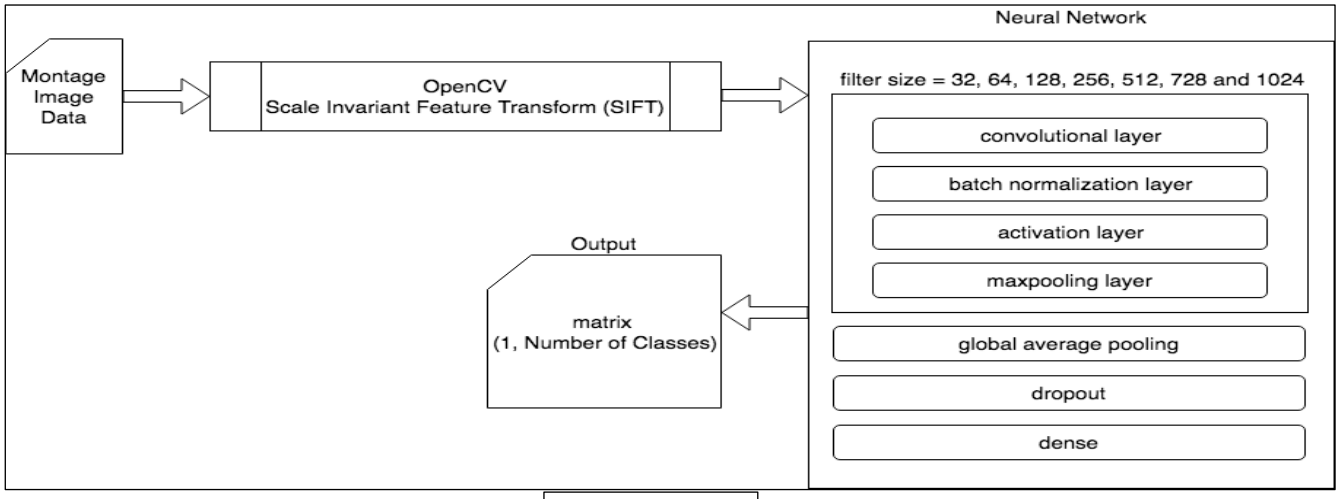


Image m4

As visible in an image 'image m4', the model reads montage image data using OpenCV and applies SIFT (Scale Invariant Feature Transform) algorithm to extract key points and descriptors. Following features extraction algorithm, neural network model comes into action where neural network consists of various filter sizes from 32, 64 to 1024 processes the image data to extract features, learn from them and detect objects. The neural network contains convolutional layer, batch normalization layer, activation layer and max pooling layer. After looping through all filter sizes, extracted features are passed for global average pooling, dropout (0.5) and dense (number of classes) layers. Resulting output of 1 by number of classes matrix. After creation of model, it was compiled with SGD optimizers (learning rate = "1e-3"), loss function as "sparse_categorical_crossentropy" and tracking "accuracy" metrics. Further, the model was trained and validated with batch size of 64 and epochs as 25 using training and validation datasets. The following image shows performance of the model on train and validation data with loss and accuracy graph for epochs:

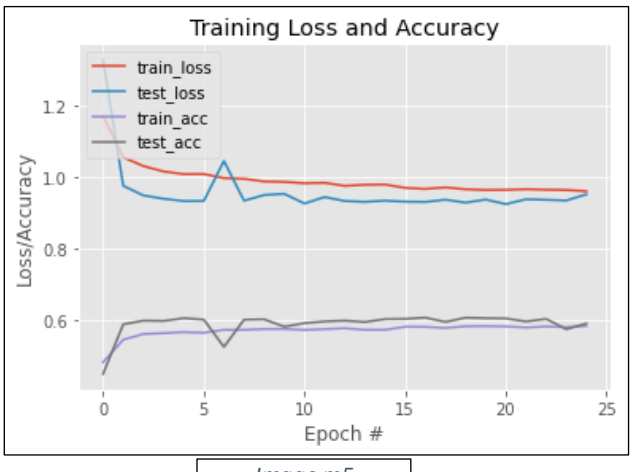


Image m5

Image 'image m5' shows the performance of model during each epoch by plotting loss and accuracy for training and validation dataset. It is visible that, on the start, till epoch five, there is sudden fall in the loss stabling around 1.0 and on the other end, there is increase in model accuracy and that is stabling around 60 percent. Apart from this, model loss has steadily decreased till last epoch and similarly model accuracy has shown steady growth during lifespan. The model was trained and validated on 14830 and 3708 samples respectively. The

training data had 4031 pedestrian, 3078 small_objects, 1055 vehicle_big and 6666 vehicle_small samples. And similarly, validation data had 1009 pedestrian, 772 small_objects, 260 vehicle_big and 1667 vehicle_small samples.

2. Pointcloud data processing model:

This model focuses on pointcloud data processing that are retrieved from one lidar and five radar sensors including features extractions from input pointcloud, model training and validation with the use of train (80 percent) and test (20 percent) splits from sklearn; and object detection performance. Following diagram states general idea of model structure and depth:

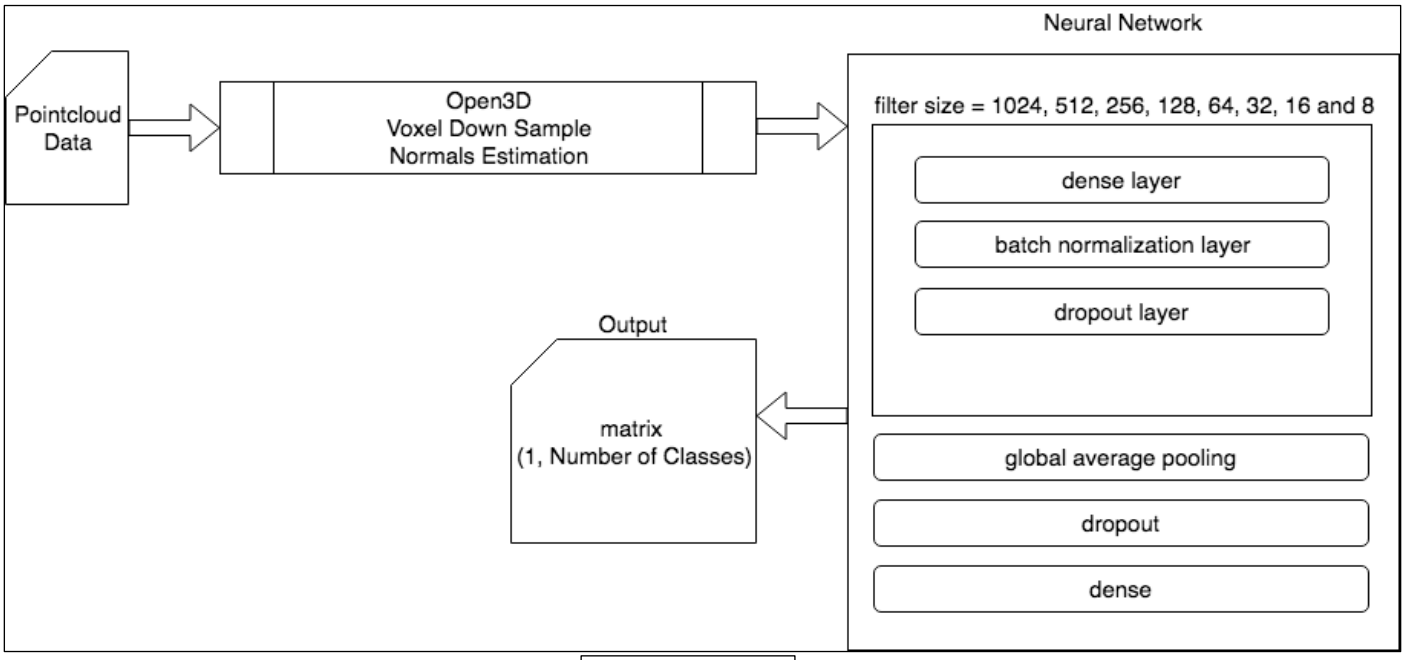


Image m6

As visible in an image 'image m6', the model reads pointcloud data using Open3D and performs voxel down sampling and normal estimation to extract important object position information. Following information extraction with Open3D, neural network model comes into action where neural network consists of various filter sizes from 1024, 512 to 8 processes the extracted information to extract features, learn from them and detect objects. The neural network contains dense layer, batch normalization and dropout (0.5) layer. After looping through all filter sizes, extracted features are passed for global average pooling, dropout (0.5) and dense (number of classes) layers. Resulting output of 1 by number of classes matrix. After creation of model, it was compiled with Adam optimizers (learning rate = "1e-3"), loss function as "sparse_categorical_crossentropy" and tracking "accuracy" metrics. Further, the model was trained and validated with batch size of 64 and epochs as 100 using training and validation datasets. The following image shows performance of the model on train and validation data with loss and accuracy graph for epochs:

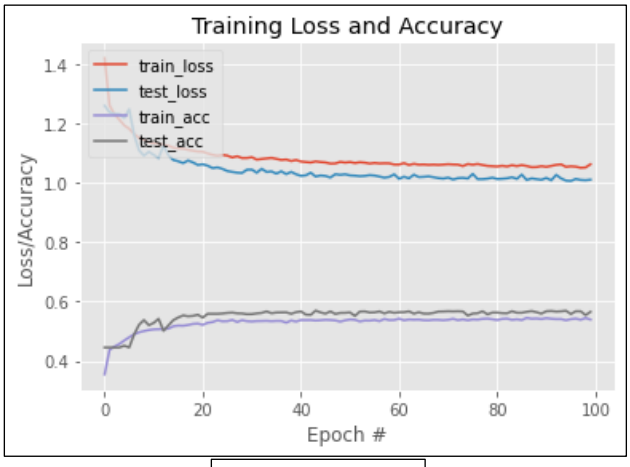


Image m7

Image 'image m7' shows the performance of model during each epoch by plotting loss and accuracy for training and validation dataset. It is visible that, on the start, till epoch twenty, there is sudden fall in the loss stabling around 1.1 and on the other end, there is increase in model accuracy and that is stabling around 55 percent. Apart from this, model loss has steadily decreased till last epoch and similarly model accuracy has shown steady growth during lifespan. The model was trained and validated on 14830 and 3708 samples respectively. The training data had 4060 pedestrian, 3031 small_objects, 1058 vehicle_big and 6681 vehicle_small samples. And similarly, validation data had 980 pedestrian, 819 small_objects, 257 vehicle_big and 1652 vehicle_small samples.

3. Continuous data processing model:

This model focuses on other continuous data processing that are retrieved from nuScenes dataset annotations including normalization, features extractions, model training and validation with the use of train (80 percent) and test (20 percent) splits from sklearn; and object detection performance. Following diagram states general idea of model structure and depth:

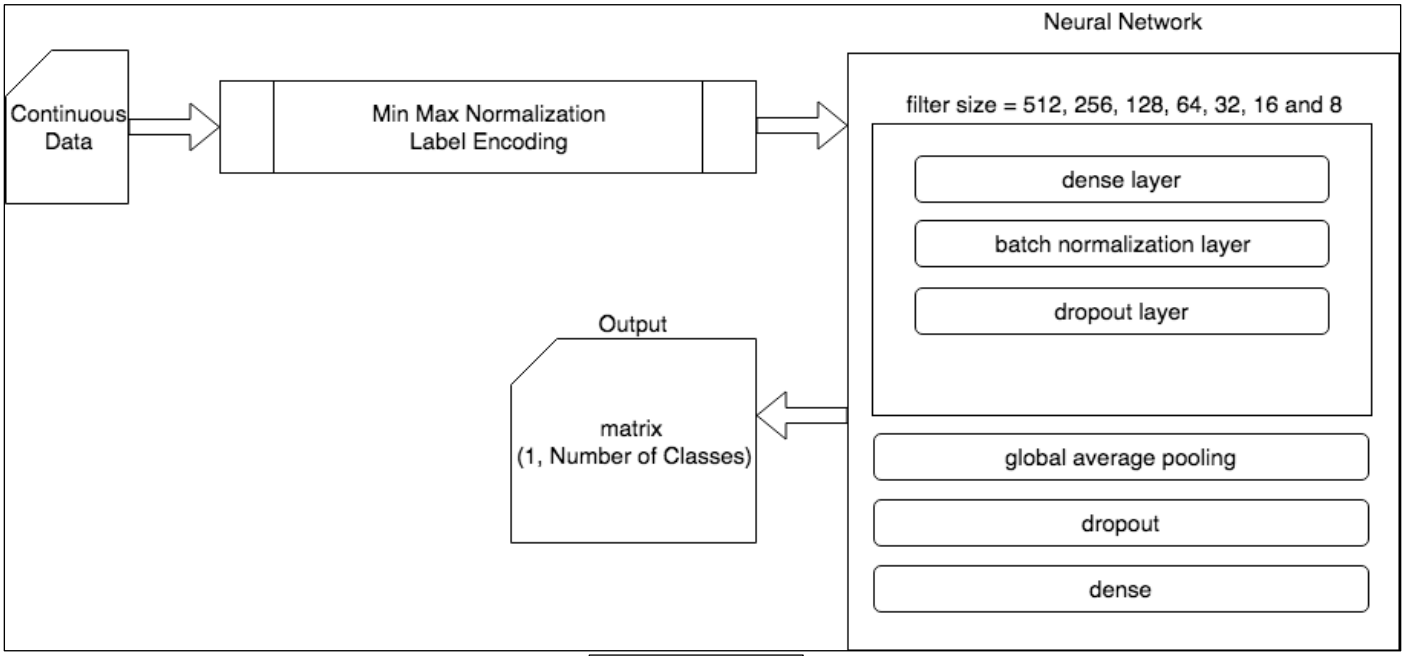


Image m8

As visible in an image 'image m8', the model reads continuous data from csv file and performs Min Max Normalization on continuous data and Label Encoding on prediction labels. Following information extraction, neural network model comes into action where neural network consists of various filter sizes from 512, 256 to 8 processes the extracted information to extract features, learn from them and detect objects. The neural network contains dense layer, batch normalization and dropout (0.5) layer. After looping through all filter sizes, extracted features are passed for global average pooling, dropout (0.5) and dense (number of classes) layers. Resulting output of 1 by number of classes matrix. After creation of model, it was compiled with Adam optimizers (learning rate = "0.01"), loss function as "sparse_categorical_crossentropy" and tracking "accuracy" metrics. Further, the model was trained and validated with batch size of 32 and epochs as 100 using training and validation datasets. The following image shows performance of the model on train and validation data with loss and accuracy graph for epochs:

Image 'image m9' shows the performance of model during each epoch by plotting loss and accuracy for training and validation dataset. It is visible that, on the start, till epoch six, there is sudden fall in the loss stabling around 1.15 and on the other end, there is increase in model accuracy and that is stabling around 49 percent. Apart from this, model loss has steadily decreased till last epoch and similarly model accuracy has shown steady growth during lifespan. The model was trained and validated on 14830 and 3708 samples respectively. The training data had 4031 pedestrian, 3078 small_objects, 1055 vehicle_big and 6666 vehicle_small samples. And similarly, validation data had 1009 pedestrian, 772 small_objects, 260 vehicle_big and 1667 vehicle_small samples.

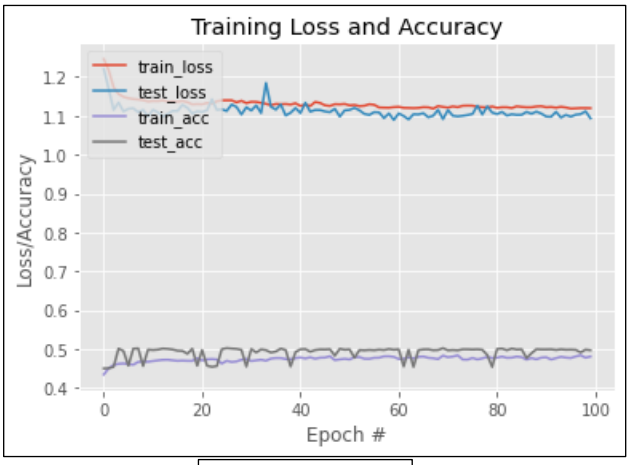


Image m9

4. Combined model:

As name suggests, this model aims to improve predictions with the use of previously created three models' learnings (image data processing model, continuous data processing model and pointcloud data processing model). As each model generates output matrix of one by number of classes, the model concatenates the output of all the models and takes them as an input to extract features based on leaning gathered. Model training and validation has been done with the use of train (80 percent) and test (20 percent) splits from sklearn. Following diagram gives general idea of model structure and depth:

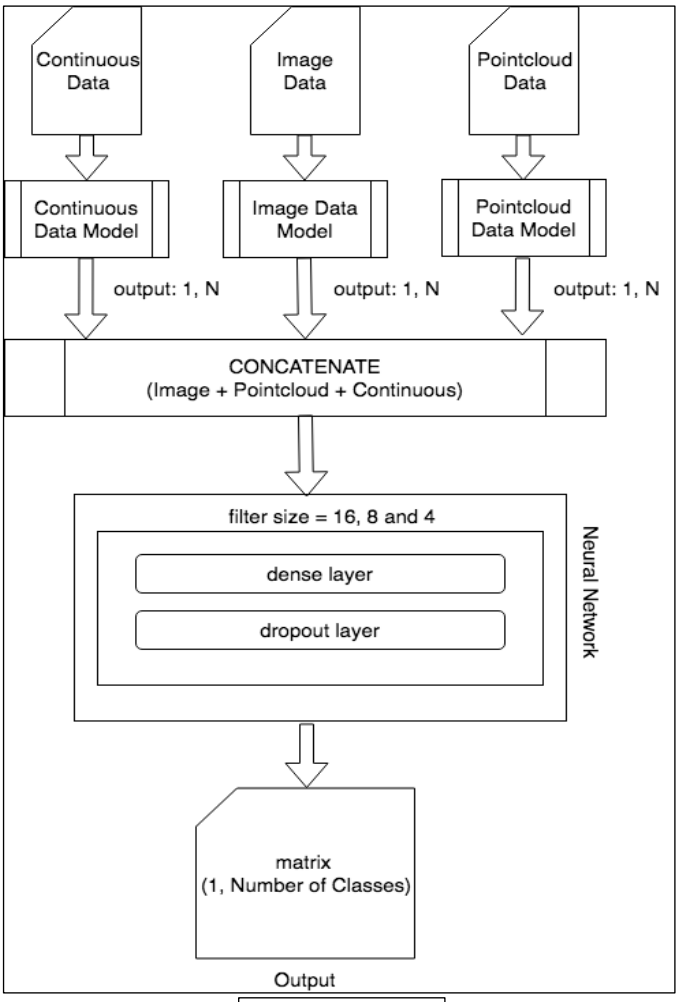


Image m10

As visible in an image 'image m10', the model concatenates the output processed by all three individual models (Image, Continuous and Pointcloud) and take them as an input for further processing.

Following, neural network model comes into action where neural network consists of various filter sizes from 16, 8 and 4 processes the received information to extract features, learn from them and detect objects. The neural network contains dense layer and dropout (0.5) layer. After looping through all filter sizes, extracted features are passed to final dense layer to receive results of one by number of classes matrix. After creation of model, it was compiled with Adam optimizers (learning rate = "1e-3"), "sparse_categorical_crossentropy" as loss function and tracking "accuracy" metrics. Further, the model was trained and validated with batch size of 64 and epochs as 15 using training and validation datasets.

The following image shows performance of the model on train and validation data with loss and accuracy graph for epochs:

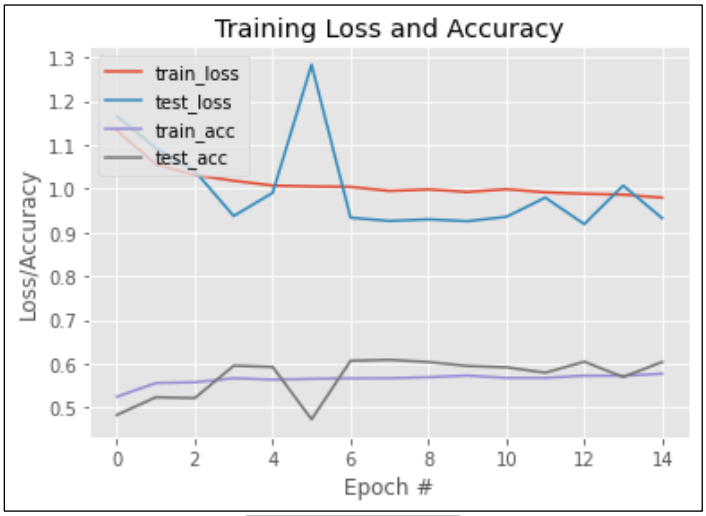


Image m11

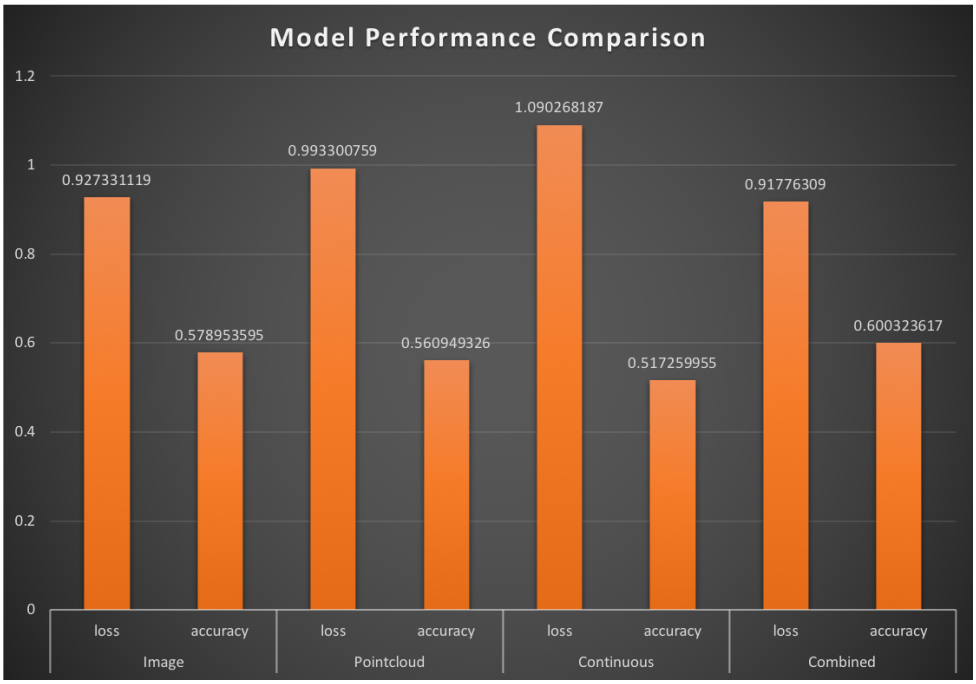
Image 'image m11' shows the performance of combined model during each epoch by plotting loss and accuracy for training and validation dataset. It is visible that, on the start, till epoch four, there is sudden fall in the loss stabling around 0.93 and on the other end, there is increase in model accuracy and that is stabling around 58 percent. Apart from this, model loss has steadily decreased till last epoch and similarly model accuracy has shown steady growth during lifespan. The model was trained and validated on 14830 and 3708 samples respectively. The training data had 4060 pedestrian, 3031 small_objects, 1058 vehicle_big and 6681 vehicle_small samples. And similarly, validation data had 980 pedestrian, 819 small_objects, 257 vehicle_big and 1652 vehicle_small samples.

Model performance and results:

The research focuses on testing all the models' performance on various randomly generated test sets. For each model, the model accuracy score and loss score are being considered to evaluate and compare all the individual models' performances with combined model.

1. Test set 1 (number of samples = 1854 and batch size = 64):

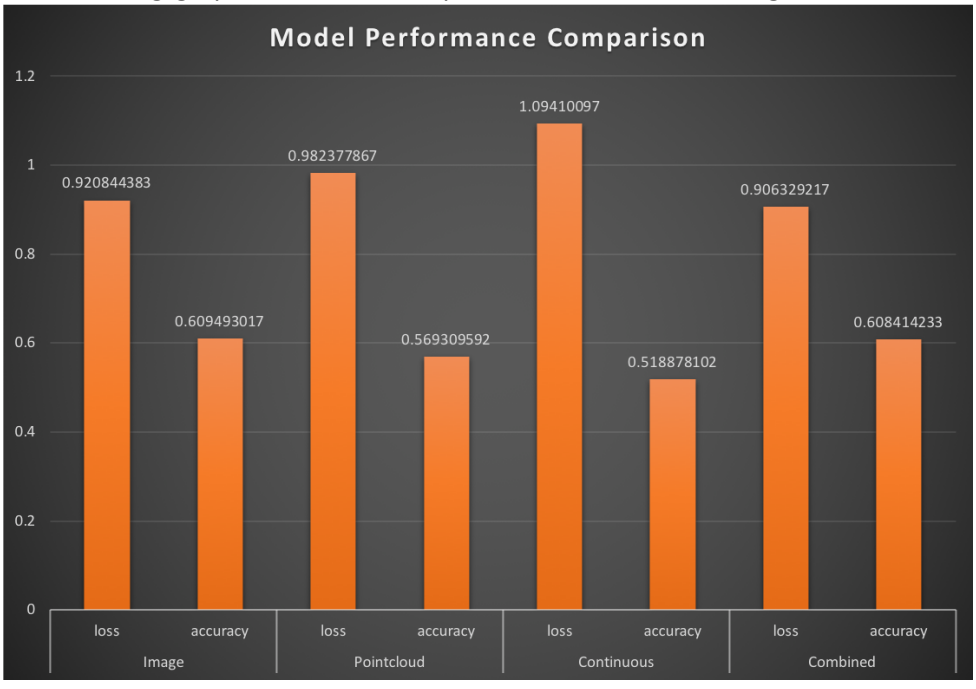
The following graph shows the comparison of all the models generated in this research. It is clearly visible that combined model performs better over other three individual models as theoretically it is likely to happen because combined model learns from the individual models and extract extra useful information from input data.



The combined model provides highest accuracy of 60 percent with loss score around 0.90 whereas second most accurate model is image model performing best over other individual models with accuracy score of 57 percent and loss score around 0.92. The pointcloud model is 56 percent accurate with 0.99 loss score and continuous model performs 51 percent accurately with approximate loss of 1.01.

2. Test set 2 (number of samples = 3708 and batch size = 64):

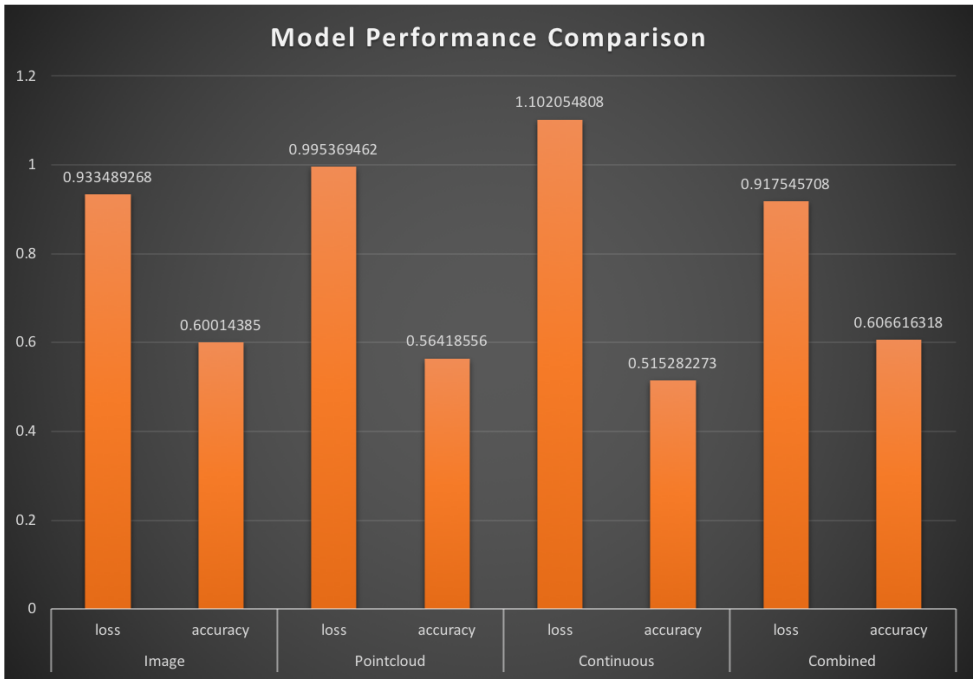
The following graph shows the comparison of all the models generated in this research. It is clearly visible that combined model performs better over other three individual models as theoretically it is likely to happen because combined model learns from the individual models and extract extra useful information from input data.



The combined model provides highest accuracy of 61 percent with loss score around 0.90 whereas second most accurate model is image model performing best over other individual models with accuracy score of 59.5 percent and loss score around 0.921. The pointcloud model is 57 percent accurate with 0.98 loss score and continuous model performs 52 percent accurately with approximate loss of 1.09.

3. Test set 3 (number of samples = 5562 and batch size = 64):

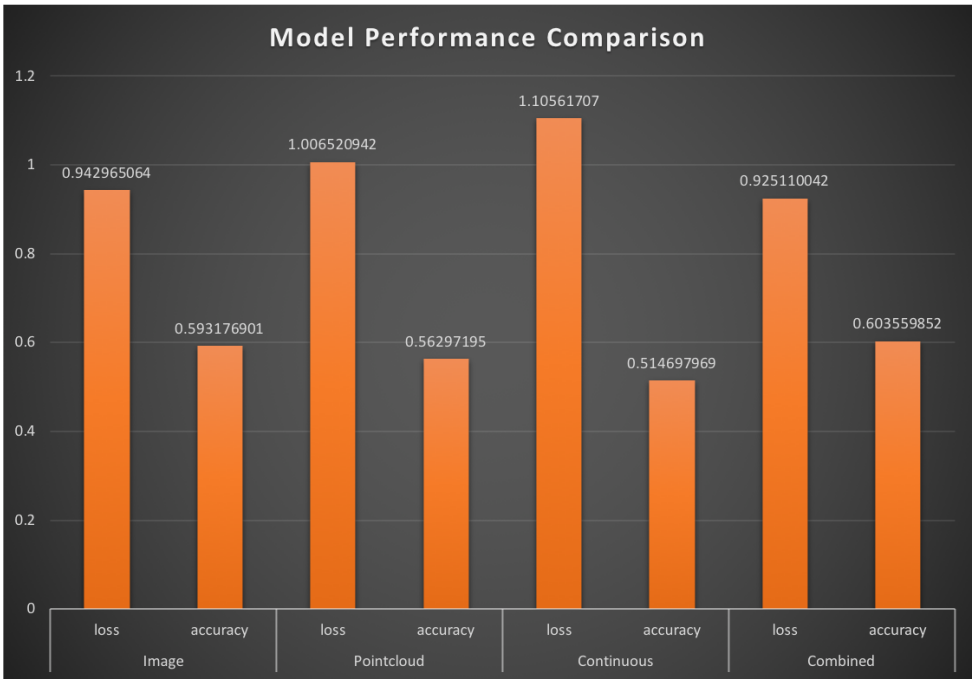
The following graph shows the comparison of all the models generated in this research. It is clearly visible that combined model performs better over other three individual models.



The combined model provides highest accuracy of 61 percent with loss score around 0.91 whereas second most accurate model is image model performing best over other individual models with accuracy score of 60 percent and loss score around 0.933. The pointcloud model is 56 percent accurate with 0.995 loss score and continuous model performs 51.5 percent accurately with approximate loss of 1.102.

4. Test set 4 (number of samples = 7416 and batch size = 64):

The following graph shows the comparison of all the models generated in this research. It is clearly visible that combined model performs better over other three individual models.



The combined model provides highest accuracy of 61 percent with loss score around 0.925 whereas second most accurate model is image model performing best over other individual models with accuracy score of 59 percent and loss score around 0.943. The pointcloud model is 56.3 percent accurate with 1.005 loss score and continuous model performs 51.46 percent accurately with approximate loss of 1.1056.

Summary:

In this section, this paper has provided information on chosen dataset, the data pre-processing methods that are being utilized, the individual models designed to take care of various types of data, the combined model designed to carry forward learning from each individual model and models' performance comparison with various randomly generated test samples. The combined model performs accurately over other individual models hence learning from various data helps in improving object detection in real-world applications.

Conclusion and future work:

This paper focuses on various research that has been carried out to gather important features from distinct sensors and their modalities. Furthermore, it describes the literature that is available to fuse the gather feature during deep neural network composition. With regards to the studies that are mentioned in this paper, majority of the previous development in autonomous vehicles have been focused on fusing the features extracted from camera data with extracted features from LiDARs data on the multimodal datasets that are captured in limited vivid environments. There is least to no remarkable research has been done to other available sensors data such as radars, ultrasonics and others hence main area is to explore features extractions technique(s) that are well suited for variety of input sensors data and development of deep multimodal perception networks that are robust and guarantee accuracy over time.

The current study has been carried out by developing custom neural networks from scratch to deal with various types of data such as image data, pointcloud data (from LiDARs and RADARs) and other continuous data (such as number of LiDAR points in annotations, number of RADAR points in annotations and object visibility while capturing data with autonomous vehicles). Further using learning from these individual models, the combined model has been designed that provides better prediction. The testing has been carried out on various randomly generated test sets and performance comparison has been shown. As this research has been carried out with custom created model, the prediction can be improved by utilizing pretrained models that are designed for specific object detection tasks such as RCNN, Fast RCNN, Faster RCNN and others. Plus, features extraction from the pointcloud can be explored more to extract more useful information.

References:

[1] J. B. Greenblatt and S. Saxena, "Autonomous taxis could greatly reduce greenhouse-gas emissions of US light-duty vehicles," *Nat. Clim. Chang.*, vol. 5, no. 9, pp. 860–863, 2015.

[2] D. Miculescu and S. Karaman, "Polling-systems-based autonomous vehicle coordination in traffic intersections with no traffic signals," *IEEE Trans. Automat. Contr.*, vol. 65, no. 2, pp. 680–694, 2020.

[3] G. Mahesh and T. Satish Kumar, "Real time traffic light detection by autonomous vehicles using artificial neural network techniques," *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 10, pp. 2129–2133, 2019.

[4] Y. Ma, Z. Wang, H. Yang, and L. Yang, "Artificial Intelligence Applications in the Development of Autonomous Vehicles: A Survey," *IEEE/CAA J. Autom. Sin.*, vol. 7, no. 2, pp. 315–329, 2020.

[5] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2019-June, pp. 12689–12697, 2019.

[6] A. Plebe, M. Da Lio, and D. Bortoluzzi, "On Reliable Neural Network Sensorimotor Control in Autonomous Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 2, pp. 711–722, 2020.

[7] X. Mu, B. He, X. Zhang, Y. Song, Y. Shen, and C. Feng, "End-to-end navigation for Autonomous Underwater Vehicle with Hybrid Recurrent Neural Networks," *Ocean Eng.*, vol. 194, no. October, 2019.

[8] P. M. Kebria, A. Khosravi, S. M. Salaken, and S. Nahavandi, "Deep imitation learning for autonomous vehicles based on convolutional neural networks," *IEEE/CAA J. Autom. Sin.*, vol. 7, no. 1, pp. 82–95, 2020.

[9] A. Zhou, Z. Li, and Y. Shen, "Anomaly detection of CAN bus messages using a deep neural network for autonomous vehicles," *Appl. Sci.*, vol. 9, no. 15, 2019.

[10] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors (Switzerland)*, vol. 18, no. 10, pp. 1–17, 2018.

[11] H. Caesar *et al.*, "nuScenes: A multimodal dataset for autonomous driving," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, no. March, 2019.

[12] V. Vaquero, A. Sanfeliu, and F. Moreno-Noguer, "Deep Lidar CNN to Understand the Dynamics of Moving Vehicles," *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 4504–4509, 2018.

[13] D. Feng *et al.*, "Deep Multi-Modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges," *IEEE Trans. Intell. Transp. Syst.*, pp. 1–20, 2020.

[14] O. Mees, A. Eitel, and W. Burgard, "Choosing smartly: Adaptive multimodal fusion for object detection in changing environments," *IEEE Int. Conf. Intell. Robot. Syst.*, vol. 2016-Novem, pp. 151–156, 2016.

[15] T. Karasawa, K. Watanabe, Q. Ha, A. Tejero-De-Pablos, Y. Ushiku, and T. Harada, "Multispectral object detection for autonomous vehicles," *Themat. Work. 2017 - Proc. Themat. Work. ACM Multimed. 2017, co-located with MM 2017*, pp. 35–43, 2017.

[16] A. Valada, G. L. Oliveira, T. Brox, and W. Burgard, "Deep Multispectral Semantic Scene Understanding of Forested Environments Using Multimodal Fusion," pp. 465–477, 2017.

[17] X. Du and A. Zare, "Multiresolution Multimodal Sensor Fusion for Remote Sensing Data with Label Uncertainty," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 4, pp. 2755–2769, 2020.

[18] K. Bengler, K. Dietmayer, B. Färber, M. Maurer, C. Stiller, and H. Winner, "Three Decades of Driver Assistance Systems," *XXVIII Encontro da Assoc. Nac. Pós-Graduação e Pesqui. em Adm.*, vol. 6, no. 4, pp. 1–9, 2004.

[19] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 3354–3361, 2012.

[20] L. Schneider *et al.*, "Multimodal neural networks: RGB-D for semantic segmentation and object detection," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10269 LNCS, pp. 98–109, 2017.

[21] S. Hwang, J. Park, N. Kim, Y. Choi, and I. S. Kweon, "Multispectral pedestrian detection: Benchmark dataset and baseline," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07-12-June, pp. 1037–1045, 2015.

[22] Y. Choi *et al.*, "KAIST Multi-Spectral Day/Night Data Set for Autonomous and Assisted Driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 3, pp. 934–948, 2018.

[23] M. Meyer and G. Kuschk, "Automotive radar dataset for deep learning based 3D object detection," *EuRAD 2019 - 2019 16th Eur. Radar Conf.*, pp. 129–132, 2019.

[24] D. Barnes, M. Gadd, P. Murcutt, P. Newman, and I. Posner, "The Oxford Radar RobotCar Dataset: A Radar Extension to the Oxford RobotCar Dataset," 2019.

[25] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset. The International Journal of Robotics Research," *Int. J. Rob. Res.*, no. October, pp. 1–6, 2013.

[26] M. Braun, S. Krebs, F. Flohr, and D. M. Gavrila, “EuroCity Persons: A Novel Benchmark for Person Detection in Traffic Scenes,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 8, pp. 1844–1861, 2019.
[27] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollar, “Panoptic segmentation,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2019-June, pp. 9396–9405, 2019.
[28] K. Mukherjee and S. Natesan, “Speed/accuracy trade-offs for modern convolutional object detectors Jonathan,” *Comput. (Vienna/New York)*, vol. 84, no. 3–4, pp. 209–230, 2017.
[29] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 386–397, 2020.
[30] Y. Xing *et al.*, “Learning Rich Features from RGB-D Images for Object Detection and Segmentation,” *2019 IEEE Int. Conf. Image Process.*, vol. 36, no. 2, pp. 899–903, 2019.
[31] B. Li, “3D fully convolutional network for vehicle detection in point cloud,” *IEEE Int. Conf. Intell. Robot. Syst.*, vol. 2017-Septe, pp. 1513–1518, 2017.
[32] L. Baker and J. L. Santa, “VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection,” *Mem. Cognit.*, vol. 5, no. 1, pp. 151–154, 1977.
[33] I. Cherabier, C. Hane, M. R. Oswald, and M. Pollefeys, “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation,” *Proc. - 2016 4th Int. Conf. 3D Vision, 3DV 2016*, pp. 601–610, 2016.
[34] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “PointNet++: Deep hierarchical feature learning on point sets in a metric space,” *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, pp. 5100–5109, 2017.
[35] S. Wang, S. Suo, W. C. Ma, A. Pokrovsky, and R. Urtasun, “Deep Parametric Continuous Convolutional Neural Networks,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 2589–2597, 2018.
[36] T. Roddick, A. Kendall, and R. Cipolla, “Orthographic Feature Transform for Monocular 3D Object Detection,” 2018.
[37] Y. Wang, W. L. Chao, Di. Garg, B. Hariharan, M. Campbell, and K. Q. Weinberger, “Pseudo-lidar from visual depth estimation: Bridging the gap in 3D object detection for autonomous driving,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2019-June, pp. 8437–8445, 2019.
[38] O. Schumann, M. Hahn, J. Dickmann, and C. Wöhler, “Semantic Segmentation on Radar Point Clouds,” *2018 21st Int. Conf. Inf. Fusion, FUSION 2018*, pp. 2179–2186, 2018.