C O M M U N I C A T I O N S

FACULTY OF SCIENCES
UNIVERSITY OF ANKARA

DE LA FACULTE DES SCIENCES
DE L'UNIVERSITE D'ANKARA

Series A2-A3: Physical Sciences and Engineering

VOLUME: 66 **Number: 1** **YEAR: 2024**

Faculy of Sciences, Ankara University
06100 Beşevler, Ankara –Türkiye
ISSN 1303-6009 E-ISSN 2618-6462

C O M M U N I C A T I O N S

FACULTY OF SCIENCES
UNIVERSITY OF ANKARA

DE LA FACULTE DES SCIENCES
DE L'UNIVERSITE D'ANKARA

Series A2-A3: Physical Sciences and Engineering

Volume: 66 **Number: 1** **Year: 2024**

This Journal is published two issues in a year by the Faculty of Sciences, University of Ankara. Articles and any other material published in this journal represent the opinions of the author(s) and should not be construed to reflect the opinions of the Editor(s) and the Publisher(s).

Correspondence Address:
COMMUNICATIONS
EDITORIAL OFFICE
Ankara University, Faculty of Sciences,
06100 Beşevler, ANKARA – TÜRKİYE
Tel: (90) 312-216 87 00 Fax: (90) 312-216 89 00
e-mail: commun@science.ankara.edu.tr
http://communications.science.ankara.edu.tr/index.php?series=A2A3

Print:
Ankara University Press
İncitaş Sokak No:10 06510 Beşevler
ANKARA – TÜRKİYE

C O M M U N I C A T I O N S

FACULTY OF SCIENCES
UNIVERSITY OF ANKARA

DE LA FACULTE DES SCIENCES
DE L'UNIVERSITE D'ANKARA

Series A2-A3: Physical Sciences and Engineering

Volume: 66 **Number: 1** **Year: 2024**

Research Articles

http://communications.science.ankara.edu.tr

Commun.Fac.Sci.Univ.Ank.Series A2-A3
Volume 66, Number 1, Pages 01-25 (2024)
DOI: 10.33769/aupse.1292652
ISSN 1303-6009 E-ISSN 2618-6462

Research Article; Received: May 4, 2023; Accepted: August 5, 2023

END-TO-END, REAL TIME AND ROBUST BEHAVIORAL PREDICTION MODULE WITH ROS FOR AUTONOMOUS VEHICLES

Tolga KAYIN[1] and Çağatay Berke ERDAŞ[1]

[1]Department of Computer Engineering, Başkent University, Ankara, TÜRKİYE

Abstract. In the world where urbanization and population density are increasing, transportation methods are also diversifying and the use of unmanned vehicles is becoming widespread. In order for unmanned vehicles to perform their tasks autonomously, they need to be able to perceive their own position, the environment and predict the possible movements/routes of environmental factors, similar to living things. In autonomous vehicles, it is extremely important for the safety of the vehicle and the surrounding factors to be able to predict the future position of the objects around it with high performance so that the vehicle can plan correctly. Due to the stated reasons, the behavioral prediction module is a very important component for autonomous vehicles, especially in moving environments. In this study, fast and successful robotic behavioral prediction module has been developed to enable the autonomous vehicle to plan more safely and successfully.

1. Introduction

The function and importance of autonomous vehicles are increasing day by day. It is foreseen that autonomous vehicles will play an important role in the future in order to reduce the density in transportation and to eliminate human-induced accidents. Apart from transportation, autonomous vehicles are becoming more and more common in areas such as agriculture, health and education.

Autonomous vehicles, inspired by living things; It consists of modules such as perception to detect the environment, localization to determine its own position, planning to where and how to go, control for its movement and behavioral prediction for possible movement routes of surrounding objects. Middleware such as Robotic Operating System (ROS) [1], ZeroMQ (ZMQ) [2], Robotic Operating System 2 (ROS2) [3] are needed for these modules to communicate with each other correctly and completely. These middlewares enable modules to transmit the desired message to the relevant module. Thanks to the ROS middleware tools that is used in the study, it also provides benefits such as visualizing, recording and observing data.

The behavioral prediction module is the one of the most important factor for the accurate result of the planning module in autonomous vehicles. The behavioral

Keywords. Behavioral prediction, trajectory prediction, autonomous vehicles, ROS.
tlg_kayin@hotmail.com-Corresponding author; 0009-0005-6232-5059
berdas@baskent.edu.tr; 0000-0003-3467-9923.

prediction module generates output that predicts future positions by keeping the past positions of objects around the ego vehicle. This output creates an input to the planning module by combining with the objects found by the detection module. An autonomous vehicle without a behavioral prediction module will consider all objects as static and plan accordingly, but in highway conditions or urban traffic scenarios, an accident will be inevitable if the possible routes of vehicles or pedestrians are not taken into account. To give an example from scenarios that are frequently experienced in daily life, in order for an autonomous vehicle to consider a pedestrian preparing to cross the street, the autonomous vehicle must know the pedestrian's possible route. Similarly, while the autonomous vehicle is changing lanes, it must calculate the possible route according to the speed of the vehicle from behind, otherwise there will most likely be an accident.

The developed behavioral prediction module is based on ROS and works in real time. Features such as ROS middleware, dynamic history hold and release structure, direction error correction, covariance distribution visualization, and message type matching suitable for planning have been added to the multi modal CVAE-based model [4].Thus, an end-to-end autonomy module structure was created that will send the possible routes of the surrounding vehicles to the planning module.

In the next part of the study, first of all, behavioral prediction approaches and studies in the literature will be summarized, then information about the methodology used in the study will be given and the developed module will be explained in detail. Afterwards, the results obtained with the test data will be shared. Finally, the result of the study will be expressed and suggestions for future studies will be shared.

2. Materials and Method

2.1. **Related Methods.** There is less research in the field of behavioral prediction compared to areas such as perception, localization, and planning of autonomous vehicles. The biggest reason for this is that it is more difficult to determine the location of environmental factors in the future than the problems in other areas. When the trajectory prediction approaches are examined [5] [6], although there are approaches such as representation, output types, modeling, situational awareness, the modeling approach will be used as the main approach in categorizing the studies in this article. In addition, information will be provided in terms of representation, output and situational awareness types for the studies. When the studies are examined in terms of modeling methods; Behavioral prediction methods as shown in Figure 1; it consists of physics-based, machine learning-based, deep learning-based and reinforcement learning-based methods.

Physics-based methods take information from the dynamics and kinematics of the vehicle. It consists of Single Trajectory, Kalman Filtering, Monte Carlo Methods.

Models using single trajectory mostly use the kinematic information of the vehicle. Although there are also models that use the dynamic information of the vehicle, these models are more complex. Dynamic models consider all forces that govern motion. Dynamic models are highly complex due to the factors involved. For example, for a vehicle, the dynamic model considers the forces acting on the tires, the driver’s actions and their effects on the vehicle’s engine and transmission. For trajectory prediction, it doesn't make much sense to use a dynamic model to model such complex behavior unless you intend to run a control-oriented application [7]. Kinematic models are more commonly used than dynamic models due to their simpler structure. One of the most commonly used is Constant Velocity (CV). A simple example of a kinematic model is the CV model used in [8]. The CV model assumes that the recent relative motion of an object determines its future trajectory. Similarly Ammoun et al. [9] and Schubert et al. [10] They estimated the possible trajectories of the vehicle using the Constant Acceleration (CA) method. The CA method estimates the future acceleration of the vehicle from the past acceleration data, these acceleration estimates are converted into position information and the possible position of the vehicle is found. Lytrivis et al. [11] using Constant Turn Rate and Velocity (CTRL) and Constant TurnRate & Acceleration (CTRA) with a similar approach, Batz et al. [12] Constant SteeringAngle & Velocity (CSAV) and Constant SteeringAngle & Acceleration (CSAA) models were used by adding wheel data to the model.

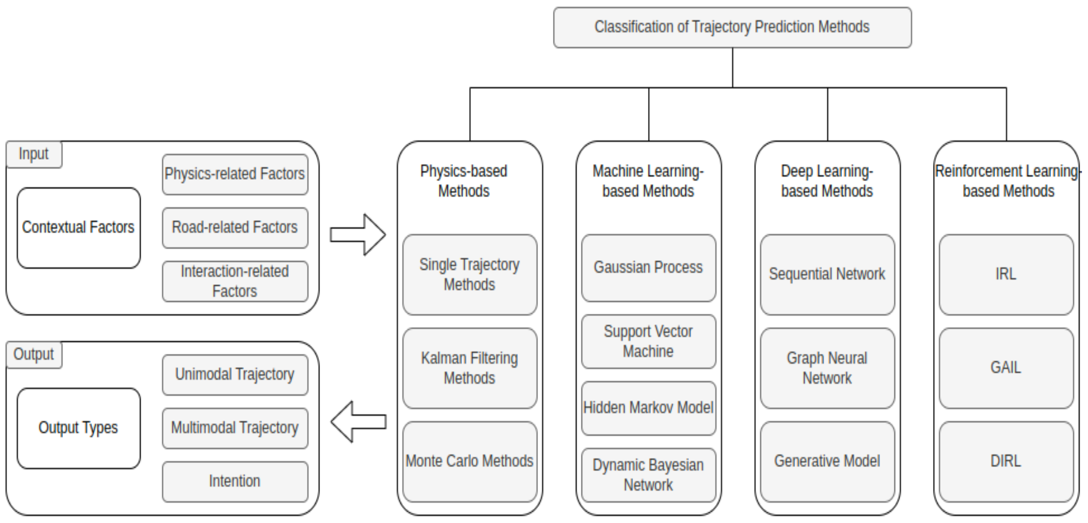


FIGURE 1. Trajectory prediction methods.

Unlike physics-based methods, machine learning methods are based on the principle of obtaining predicted trajectory by data mining. In comparison to physics-based models which are limited to low-level properties of motion and cannot estimate well the long term dependencies in motion, the learning-based models on the other hand tend to capture and incorporate long term dependencies and changes caused by external factors. The most widely used main machine learning methods are Gaussian Process (GP), Support Vector Machine (SVM), Hidden Markov Model (HMM), Dynamic Bayesian Network (DBN), K-Nearest Neighbors (KNN), Decision Tree methods. Since SVM can output the characteristics of classification probability, Kumar et al. [13] propose a layered architecture method combining SVM and Bayesian filtering to identify lane-changing maneuvers to obtain more accurate identification results

In real life, it could only be observed the tangible states that are visible on surface, but we cannot intuitively express the hidden states. Thus, it is needed to develop a Markov process involving hidden states and find the intrinsic state of an event by the set of observable states related to the probability of the hidden state. This is the so-called hidden Markov model. Based on HMM, Qioa et al. [14] presents an algorithm named HMTP* that adaptively chooses parameters to imitate real scenery with dynamically changing speeds. In [15], her HMM connection with fuzzy logic is applied to predict driver maneuvers. The author of [16] presents his DBN representing driver behavior and vehicle trajectory. DBNs have Markov properties. We can extend the state with more information to satisfy the Markov assumption. In [16] this is done by adding all relevant information of the process to the DBN in the form of a vector.

Although the outputs of the studies in this method are mostly multimodal, it has been observed that the model's performance increases as the situational awareness states such as map-aware, scene aware, interaction aware increase.

Deep-learning based methods consist of sequential network, graph neural network (GNN) and Generative Model methods. Sequential network methods consist of Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), RNN and CNN and Attention Mechanism, while Generative Model methods consist of Generative Adversarial Network and Conditional Variational Auto Encoder methods.

One of the most popular study with RNN & CNN is DESIRE [17], whose goal is to predict the future positions of multiple interacting agents in a dynamic (driving) scene. This takes into account the multimodal nature of future projections. For example even in the same situation, the future can be different. It can predict potential future outcomes and make strategic predictions based on them, making inferences based not only on past movement history, but also on scene context and agent interactions.

An example of work with sequential network is [18] a modified version of LSTM i.e. ST-LSTM (Spatio-temporal LSTM) is used in [18] where the interaction of multiple vehicles and its effect on trajectory of Value of Information (VOI) is estimated.

Deep learning-based studies can provide more comprehensive output and input compared to physics and machine learning-based studies. These studies mostly take interaction-aware inputs and provide multimodal or intention type output.

The reinforcement learning approach, which has been extensively studied in recent years, also appears in predicting trajectory. Reinforcement learning method is based on the decision-reward principle, focusing on finding the decision that will maximize the reward.

According to studies using these approaches; In Sun et al. [19] study, interaction related elements are taken into account to achieve probabilistic estimation for AVs by using IR. Future trajectory distribution is defined by driving manoeuvers. Kufler et al. [20] Extending GAIL to his RNN optimization to show the behavior of a human driver, discriminators evaluate steps and actions. Choi et al. [21] combine the partially observable Markov decision process (POMDP) within the GAIL framework and propose a method to train the model using the discriminator reward function. The prediction problem is nonlinear, so nonlinear mapping should be used for generalizable function approximation. Wulfmeier et al. [22] propose a deep inverse reinforcement learning (DIRL) framework for approximating complex nonlinear reward functions. Some D-IRL approaches get history tracks as input. Considering driving characteristics and route shape, the authors [23] initially practiced RL to develop MDP, then learned the optimum driving procedure from IRL, and used deep neural network (DNN) to generate a reward function. In Jung et al., [24] this work proposes a convolutional LSTM to extract feature maps from his LIDAR and trajectory data considering inertia, environment, and society. This feature map is integrated with output reward map to forecast the traversability map.

Reinforcement learning methods, similar to deep learning methods, can take extensive inputs such as road and scene related factors and provide comprehensive outputs in the form of unimodal and intention.

2.2. **Datasets.** Datasets are required for training or testing the above mentioned methods or models. These datasets consist of various sensor data, map data and their annotations. The main ones are KITTI [25], nuScenes [26], Argoverse [27] and NGSIM [28] datasets. The model that is worked on in this article, trained on nuScenes dataset. NuScenes dataset is a large-scale autonomous driving dataset which has several distinct dataset such as nuPlan [29] for planning, nuScenes for perception, nuImages [30] for image level operations. In the following sections, comparisons of various studies on these data sets will be given.

2.3. **Evaluation Metrics.** Reliable and generic metrics are needed to measure the success of the studies. Some metrics that can be used to compare related works are given below.

Root Mean Squared Error (RMSE): “RMSE computes the square root of the mean squared forecast error” ([6], p.14). As shown in the equation (1) while $\hat{y}_i$ stands for estimated value (m), y_i indicates observed value and n is the number of samples.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}} \tag{1}$$

Average displacement error (ADE): “The average distance between the predicted trajectory and the ground truth” ([6], p.14). In the formulas (2) given below, x_i and y_i stand for predicted trajectory for one second interval in meters at x and y axes respectively, x_i^{GT} and y_i^{GT} indicate observed trajectory for one second interval in meters at *x* and *y* axes respectively, and *T* is time in seconds.

$$ADE = \frac{1}{T}\sum_{t=1}^{T} \sqrt{(xi - x_i^{GT})^2 + (y_i - y_i^{GT})^2} \tag{2}$$

Final displacement error (FDE): “The distance between the final prediction results and the corresponding ground truth location” ([6], p.14). In the equation (3) given below, x_T and y_T stand for predicted trajectory for one second interval in meters at x and y axes respectively, x_T^{GT} and y_T^{GT} indicate observed trajectory for one second interval in meters at *x* and *y* axes respectively, and *T* is time in seconds.

$$\text{FDE} = \sqrt{(xT - x_T^{GT})^2 + (y_T - y_T^{GT})^2} \tag{3}$$

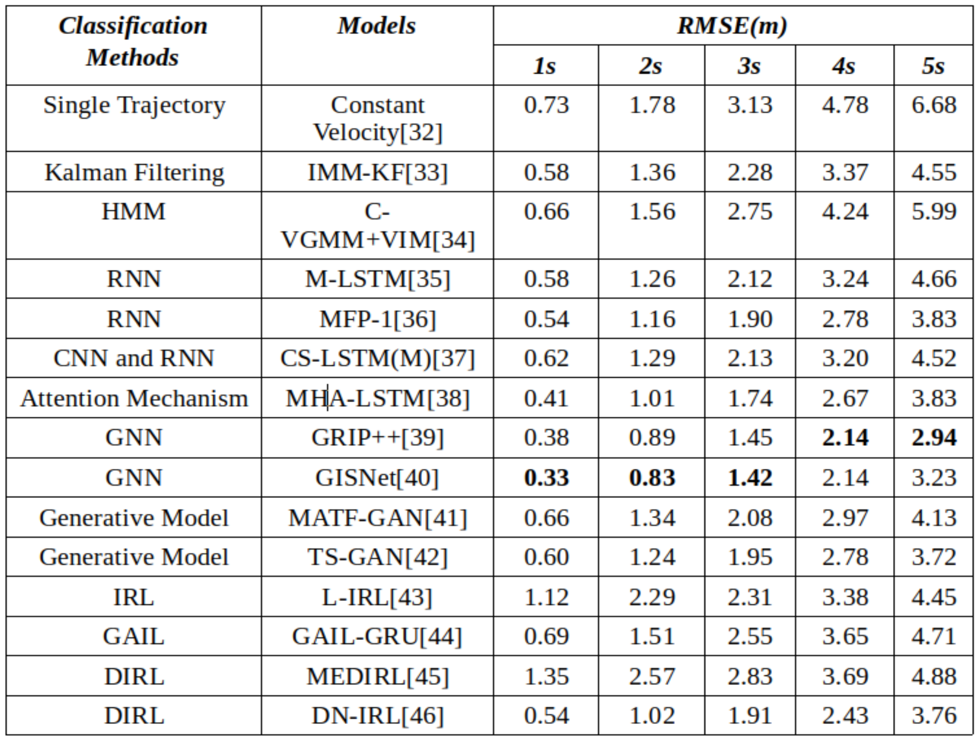
Miss Rate (MR): “Based on the L2 distance of the final positions, the ratio of cases where the estimated trajectory isn’t within 2.0 meters of the ground truth” ([6], p.14). In the equation (4) given below, ‘misses’ is the predicted trajectory not within 2.0 meters of the ground truth and ‘hits’ is the predicted trajectory within 2.0 meters of the ground truth.

$$\text{MR} = \frac{misses}{hits + misses} \tag{4}$$

When the Tables 1 and 2 are examined, although physics-based and machine-learning-based methods require low computational load, their performance decreases as the estimated time (2s>) increases. Compared to these two methods, deep learning and reinforcement-based learning methods can predict longer time successfully, although they overlay more computation load.

When deep learning and reinforcement-based methods are compared, it is seen that the deep learning-based method is more successful. An open source, ROS structured, end to end, configurable, robust, multi-class behavioral prediction module could not be found. The most related work found [31] is one that predicts pedestrian-only trajectories with ROS.

TABLE 1. Comparison of the trajectory prediction RMSE results of models using various methods trained on NGSIM dataset under highway condition.



Classification Methods	***Models***	***RMSE(m)***				
		1s	***2s***	***3s***	***4s***	***5s***
Single Trajectory	Constant Velocity[32]	0.73	1.78	3.13	4.78	6.68
Kalman Filtering	IMM-KF[33]	0.58	1.36	2.28	3.37	4.55
HMM	C-VGMM+VIM[34]	0.66	1.56	2.75	4.24	5.99
RNN	M-LSTM[35]	0.58	1.26	2.12	3.24	4.66
RNN	MFP-1[36]	0.54	1.16	1.90	2.78	3.83
CNN and RNN	CS-LSTM(M)[37]	0.62	1.29	2.13	3.20	4.52
Attention Mechanism	MHA-LSTM[38]	0.41	1.01	1.74	2.67	3.83
GNN	GRIP++[39]	0.38	0.89	1.45	**2.14**	**2.94**
GNN	GISNet[40]	**0.33**	**0.83**	**1.42**	2.14	3.23
Generative Model	MATF-GAN[41]	0.66	1.34	2.08	2.97	4.13
Generative Model	TS-GAN[42]	0.60	1.24	1.95	2.78	3.72
IRL	L-IRL[43]	1.12	2.29	2.31	3.38	4.45
GAIL	GAIL-GRU[44]	0.69	1.51	2.55	3.65	4.71
DIRL	MEDIRL[45]	1.35	2.57	2.83	3.69	4.88
DIRL	DN-IRL[46]	0.54	1.02	1.91	2.43	3.76

TABLE 2. Comparison of the trajectory prediction FDE, ADE and MR results of models using various methods trained in Argoverse data set under urban condition.

Classification Methods	***Models***	***K[1]=6***			***K[1]=1***		
		minFDE	***minADE***	***MR***	***minFDE***	***minADE***	***MR***
Physics-based	CV[47]	7.57	3.39	0.82	7.89	3.53	0.84
Machine Learning-based	NN+map[47]	4.03	2.08	0.58	8.12	3.65	0.84
RNN	LSTM+map[47]	5.44	2.34	0.69	6.81	2.96	0.81
RNN	Jean[48]	1.49	0.93	0.19	4.18	1.86	0.63
Attention Mechanism	SceneTransformer[49]	**1.23**	**0.80**	0.13	-	-	-
Attention Mechanism	mmTransformer[50]	1.34	0.84	0.15	-	-	-
GNN	LaneGCN[51]	1.36	0.87	0.16	3.78	1.71	0.59
GNN	DenseTNT[52]	1.45	0.93	**0.11**	-	-	-
GNN	LaneRCNN[53]	1.45	0.90	0.12	**3.69**	**1.69**	**0.57**
Generative Model	PRIME[54]	1.56	1.22	0.12	3.82	1.91	0.59

[1] output trajectory numbers

3. SOFTWARE DESCRIPTION

Trajectron++ has been used as the model in this article for the following reasons;

- There are many studies that show that graph-structured recurrent models are more successful,
- As it is stated in the literature review section it has a multi-classification structure,
- It is trained on a new and comprehensive data set for vehicle route estimation (nuScenes),
- It outputs both unimodal and intentional and is a scene aware model,

3.1. **Software Architecture.** Trajectron++ is in a map and interaction aware structure as shown in Figure 2, Studies on this model;

ROS middleware has been added, the data from the perception module has been adapted to the input data format in the model. In addition, a dynamic history hold/drop structure has been created, so the history of tracked objects is accumulated, and the untracked object is prevented from entering the model.

3.2. **Software Functionalities.** Due to the added ROS middle-ware, this software processes the incoming data from the perception subsystem, finds possible trajectories and sends them to the planning subsystem. In this way, the vehicle also considers the trajectories of its moving objects while planning.

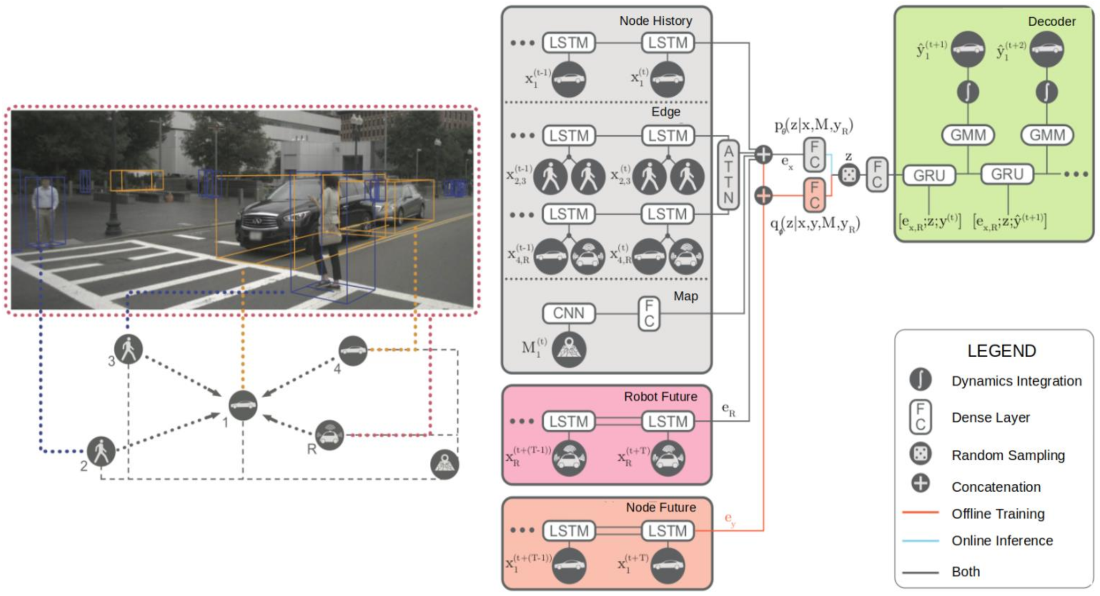
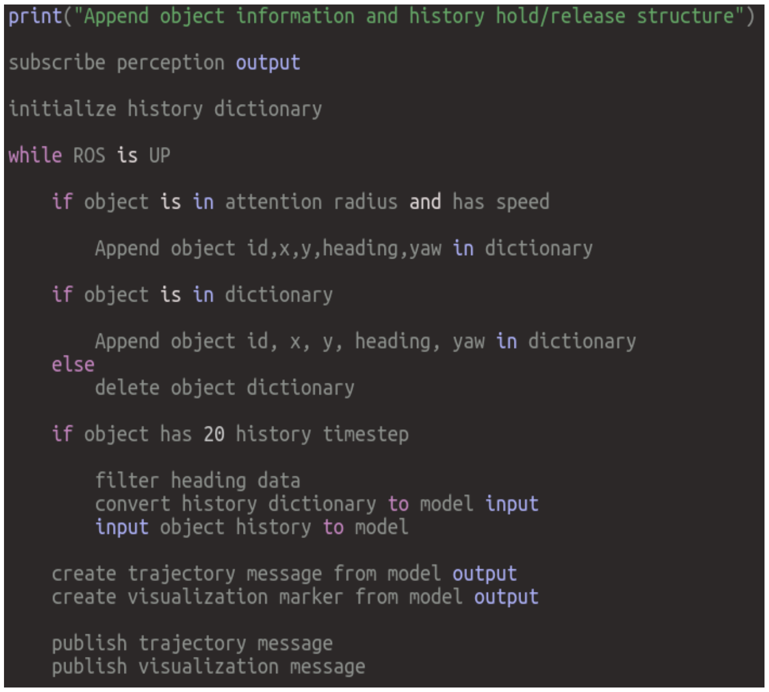


FIGURE 2. Trajectron++ model architecture.

3.3. **Sample Code Snippets Analysis.** The developed Behavioral prediction module is shown in Figure 3 as a pseudo code. First of all, by listening to the output of the perception module, information such as the position, class and orientation of the surrounding objects is obtained, and then dictionaries are created for the object at a certain speed and attention radius. As long as the ROS connection is open, the object information from the perception output is accumulated in the relevant dictionaries. When the objects reach enough history, they are converted into a data structure suitable for the model and entered as an input to the model. 6-second predicted trajectories are output from the model. This estimated trajectory information is converted into the message types for planning module and visualization. These messages are published to relevant modules by ROS middleware.

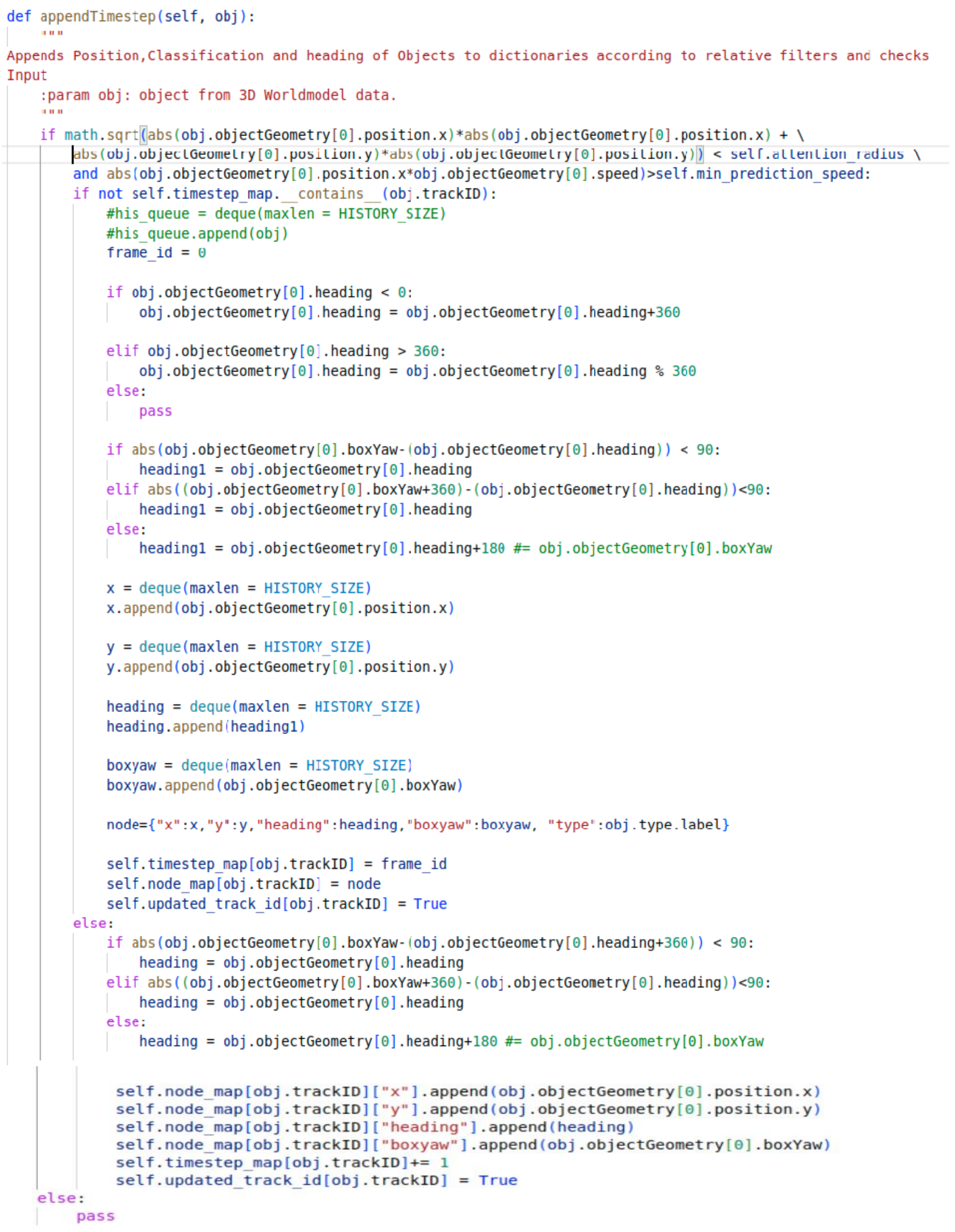
If the functions are examined in more detail, with the append history function shown in Figure 4, the heading, position and classification information of the tracked objects are passed by the distance and speed filters, and the history is appended in the related python dictionaries. With the function shown in Figure 5, the untracked object is deleted from the dictionary and only vehicle and pedestrian type entries are entered into the model.

Heading data entered as input to the model is obtained from both the direction information coming from the segmentation output and the direction information found from the vehicle position as shown in Figure 4. Although the segmentation output gives the right direction of the vehicle, it often reverses the direction by 180 degrees. While heading from the position information, the heading value is incorrect, especially when the vehicle is maneuvering. If the heading value from the position information and the value from the segmentation are more than 90 degrees, the direction information from the segmentation is rotated 180 degrees, thus a more robust heading information structure is created. In addition, a velocity and attention radius filter has been created for the tracked objects, and objects that are far from the ego vehicle or at very low speeds from the detection system do not enter the model, so that the algorithm works more efficiently. The predicted trajectory information from the model was converted into a message type suitable for the planning module and visualized. A configuration file was created as in Figure 6 in order to easily understand and configure Rostopic and message types.



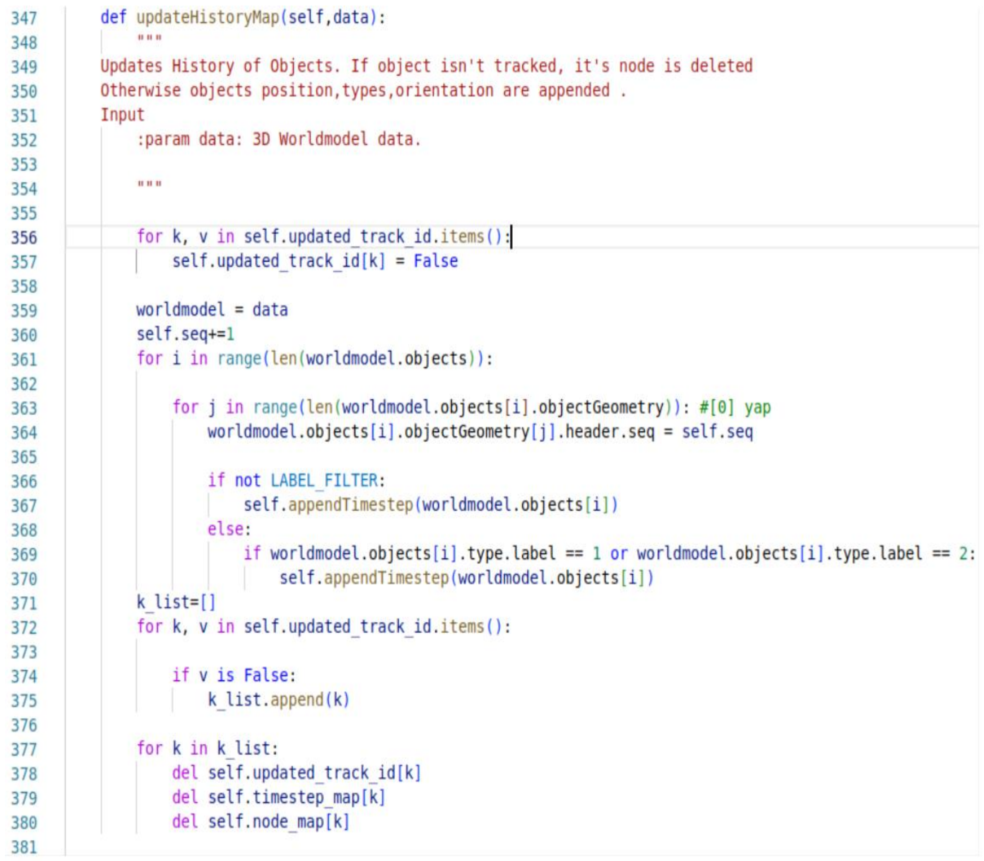
```
print("Append object information and history hold/release structure")

subscribe perception output

initialize history dictionary

while ROS is UP

    if object is in attention radius and has speed

        Append object id,x,y,heading,yaw in dictionary

    if object is in dictionary

        Append object id, x, y, heading, yaw in dictionary
    else
        delete object dictionary

    if object has 20 history timestep

        filter heading data
        convert history dictionary to model input
        input object history to model

    create trajectory message from model output
    create visualization marker from model output

    publish trajectory message
    publish visualization message
```

FIGURE 3. Pseudo code of behavioral prediction module.



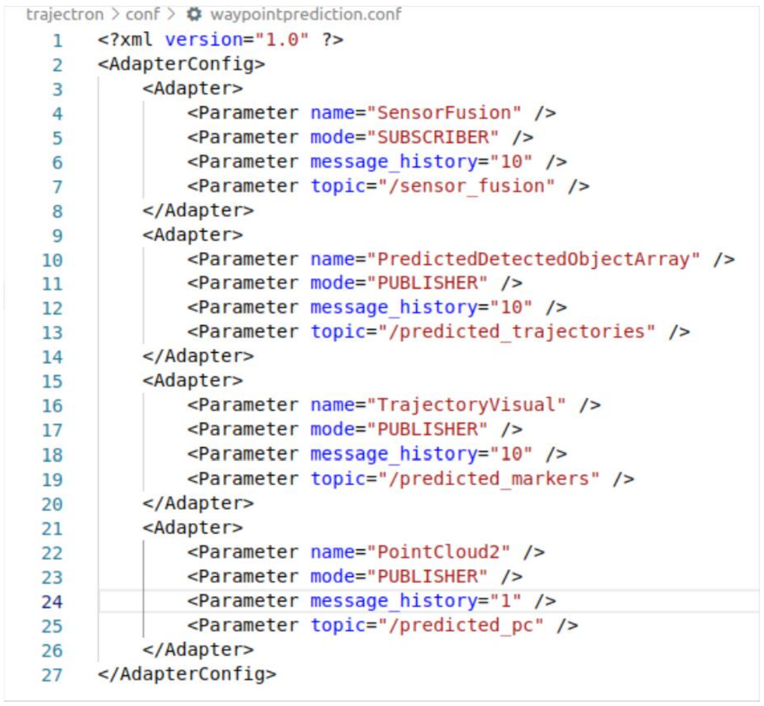
```
def appendTimestep(self, obj):
    """
Appends Position,Classification and heading of Objects to dictionaries according to relative filters and checks
Input
    :param obj: object from 3D Worldmodel data.
    """
    if math.sqrt(abs(obj.objectGeometry[0].position.x)*abs(obj.objectGeometry[0].position.x) + \
        abs(obj.objectGeometry[0].position.y)*abs(obj.objectGeometry[0].position.y)) < self.attention_radius \
        and abs(obj.objectGeometry[0].position.x*obj.objectGeometry[0].speed)>self.min_prediction_speed:
        if not self.timestep_map.__contains__(obj.trackID):
            #his_queue = deque(maxlen = HISTORY_SIZE)
            #his_queue.append(obj)
            frame_id = 0

            if obj.objectGeometry[0].heading < 0:
                obj.objectGeometry[0].heading = obj.objectGeometry[0].heading+360

            elif obj.objectGeometry[0].heading > 360:
                obj.objectGeometry[0].heading = obj.objectGeometry[0].heading % 360
            else:
                pass

            if abs(obj.objectGeometry[0].boxYaw-(obj.objectGeometry[0].heading)) < 90:
                heading1 = obj.objectGeometry[0].heading
            elif abs((obj.objectGeometry[0].boxYaw+360)-(obj.objectGeometry[0].heading))<90:
                heading1 = obj.objectGeometry[0].heading
            else:
                heading1 = obj.objectGeometry[0].heading+180 #= obj.objectGeometry[0].boxYaw

            x = deque(maxlen = HISTORY_SIZE)
            x.append(obj.objectGeometry[0].position.x)

            y = deque(maxlen = HISTORY_SIZE)
            y.append(obj.objectGeometry[0].position.y)

            heading = deque(maxlen = HISTORY_SIZE)
            heading.append(heading1)

            boxyaw = deque(maxlen = HISTORY_SIZE)
            boxyaw.append(obj.objectGeometry[0].boxYaw)

            node={"x":x,"y":y,"heading":heading,"boxyaw":boxyaw, "type":obj.type.label}

            self.timestep_map[obj.trackID] = frame_id
            self.node_map[obj.trackID] = node
            self.updated_track_id[obj.trackID] = True
        else:
            if abs(obj.objectGeometry[0].boxYaw-(obj.objectGeometry[0].heading+360)) < 90:
                heading = obj.objectGeometry[0].heading
            elif abs((obj.objectGeometry[0].boxYaw+360)-(obj.objectGeometry[0].heading))<90:
                heading = obj.objectGeometry[0].heading
            else:
                heading = obj.objectGeometry[0].heading+180 #= obj.objectGeometry[0].boxYaw

            self.node_map[obj.trackID]["x"].append(obj.objectGeometry[0].position.x)
            self.node_map[obj.trackID]["y"].append(obj.objectGeometry[0].position.y)
            self.node_map[obj.trackID]["heading"].append(heading)
            self.node_map[obj.trackID]["boxyaw"].append(obj.objectGeometry[0].boxYaw)
            self.timestep_map[obj.trackID]+= 1
            self.updated_track_id[obj.trackID] = True
    else:
        pass
```

FIGURE 4. Function to append object information.



```
def updateHistoryMap(self,data):
    """
Updates History of Objects. If object isn't tracked, it's node is deleted
Otherwise objects position,types,orientation are appended .
Input
    :param data: 3D Worldmodel data.

    """

    for k, v in self.updated_track_id.items():
        self.updated_track_id[k] = False

    worldmodel = data
    self.seq+=1
    for i in range(len(worldmodel.objects)):

        for j in range(len(worldmodel.objects[i].objectGeometry)): #[0] yap
            worldmodel.objects[i].objectGeometry[j].header.seq = self.seq

            if not LABEL_FILTER:
                self.appendTimestep(worldmodel.objects[i])
            else:
                if worldmodel.objects[i].type.label == 1 or worldmodel.objects[i].type.label == 2:
                    self.appendTimestep(worldmodel.objects[i])
    k_list=[]
    for k, v in self.updated_track_id.items():

        if v is False:
            k_list.append(k)

    for k in k_list:
        del self.updated_track_id[k]
        del self.timestep_map[k]
        del self.node_map[k]

```

FIGURE 5. History hold/release function.


trajectron > conf > waypointprediction.conf

```
<?xml version="1.0" ?>
<AdapterConfig>
    <Adapter>
        <Parameter name="SensorFusion" />
        <Parameter mode="SUBSCRIBER" />
        <Parameter message_history="10" />
        <Parameter topic="/sensor_fusion" />
    </Adapter>
    <Adapter>
        <Parameter name="PredictedDetectedObjectArray" />
        <Parameter mode="PUBLISHER" />
        <Parameter message_history="10" />
        <Parameter topic="/predicted_trajectories" />
    </Adapter>
    <Adapter>
        <Parameter name="TrajectoryVisual" />
        <Parameter mode="PUBLISHER" />
        <Parameter message_history="10" />
        <Parameter topic="/predicted_markers" />
    </Adapter>
    <Adapter>
        <Parameter name="PointCloud2" />
        <Parameter mode="PUBLISHER" />
        <Parameter message_history="1" />
        <Parameter topic="/predicted_pc" />
    </Adapter>
</AdapterConfig>
```

FIGURE 6. Rostopic configuration file.

4. RESULTS

In order to test the work done, a Rosbag which was shown at Table 3, was collected with size 19.2 GB and 17.8 GB by 10 and 12 minutes driving in Mustafa Kemal district Ankara/Turkey. Rosbag data consist of ego vehicle's localization output, perception output obtained by sensor fusion which includes positions, orientations, speeds, sizes of adjacent objects, transform information, camera images and visualization markers which includes bounding box markers. In detail ego vehicle's localization output includes ego vehicle's position, orientation and speed, perception output includes positions, orientations, speeds, sizes of adjacent objects, transform information includes relative positions and orientations of global map, local map and sensors, visualization markers includes bounding box markers of vehicles, pedestrians and unknown objects.

Trajectory prediction module was tested by playing this Rosbag on the laptop which has T1000 graphic card, i7 9th Gen. Processor, 16 GB Ram. The developed software was run Docker and Conda virtual environment with CUDA 11.3 and PyTorch.

The part of the collected Rosbag under highway conditions (Sabancı Boulevard) and the inner city (Mustafa Kemal Mah.) parts are handled separately. The part in urban conditions is between 1.min and 8.min in Rosbag, and the part in highway conditions is between 8.min and 13.min. Trajectory predictions were obtained and visualized with green sphere markers by running the trajectory prediction ROS module on the Rosbag.

As can be seen in Figure 7 when driving in an urban area, the vehicle has entered dense environments many times. Although the speed and attention radius filter were added, incorrect speed information from the perception layer to the standing vehicles caused the algorithm to work slowly in a dense environment. Sudden changes of direction of vehicles and pedestrians in dense areas are a factor that reduces the performance of the behavior prediction module.

Rosbag's highway condition driving between 8 min and 13 min has achieved better results compared to the urban section. Similar to the urban roads section, although the incorrect perception output or the inability of the perception module to track the surrounding vehicle is a factor that reduces my performance, erroneous data has not been received from the perception module very often. One of the reasons for incorrect data coming from the perception module is incorrect data coming from GPS and IMU. However, since the data was collected at an off-peak time of the day, there was few erroneous data. Approximately 3 km of driving has been done under highway condition. Nine vehicles have been tracked by perception without error. From time to time, trajectory predictions of two or three vehicles have been made at the same time without loss of performance. The average and each 5 seconds RMSE of these nine vehicles are given in Table 4. The longer the estimation period, the greater the amount of error. Visualization of multi vehicle and four single vehicle are given in Figure 8, 9, 10, 11, 12 respectively.

During the tests, when there is one vehicle/pedestrian in the environment, the module works at 5fps, and as the environmental factors increase, the operating frequency of the module decreases inversely with the number of factors. In order not to reduce the performance, model parallelization methods such as converting Tensorrt can be used. However, the Tensorrt library does not yet support the conversion of some modules of the Trajectron++ model, such as GRU and GMM. By installing an attention radius and a speed filter, distant, stationary or very slow objects were prevented from entering the model, so it was seen that the model worked more efficiently.

Considering that the distance between the two lines is about 3m and considering that lane changes take about 2-4 seconds, according to the results of the vehicles on the highway given in Table 4, it has been found that vehicles up to two seconds have an acceptable level of RMSE. RMSE can be reduced by getting more accurate data from the perception layer. Speed and heading information, which is one of the data from the perception layer, comes to the perception layer from the localization layer and it comes to localization layer from GPS and IMU sensors. Therefore, accurate data should be received from GPS and IMU sensors for the successful performing of the behavior prediction module, otherwise incorrect results will emerge from the speed and heading data entered as input to the model.

At the drives under urban condition, as seen at the dense environment Figure 7 and above inputs are entered into the model at the same time, which slows down the operating speed of the model. However, the fact that the route that both pedestrians and vehicles will take is more uncertain than highway roads is one of the negative factors affecting this structure in urban use. Similar to highway roads, speed and heading information from GPS and IMU are also important in urban areas. Even in the city, since the speeds of vehicles are lower, errors in speeds affect the system more. Although the slow speed makes it difficult to find the heading data correctly, more accurate heading data can be obtained thanks to the added binary heading data structure. In order for the behavior prediction module to be used in the city, the environment must be controlled and run on more advanced computers. Controlled environment refers to the environments in which the algorithm is fed with HDMap, traffic rules, traffic signs, node interactions and objects move within the framework of these rules.

TABLE 3. Collected Rosbag for testing.

Dataset Type: Rosbag	
Size : 39 GB	**Duration: 13 min**
Data	***Data Description***
• Image Data	• Image data from camera
• Localization data	• Position and orientation data from localization module
• Camera Object Detections	• Object detections from camera
• Sensor fusion output	• Object detections from perception module
• Tf and Tf static	• Static and dynamic transformations of sensors, local and global map
• Lidar Data	• Pointcloud data from Lidar
• Visualization Markers	• Visualization Markers for 3D World model

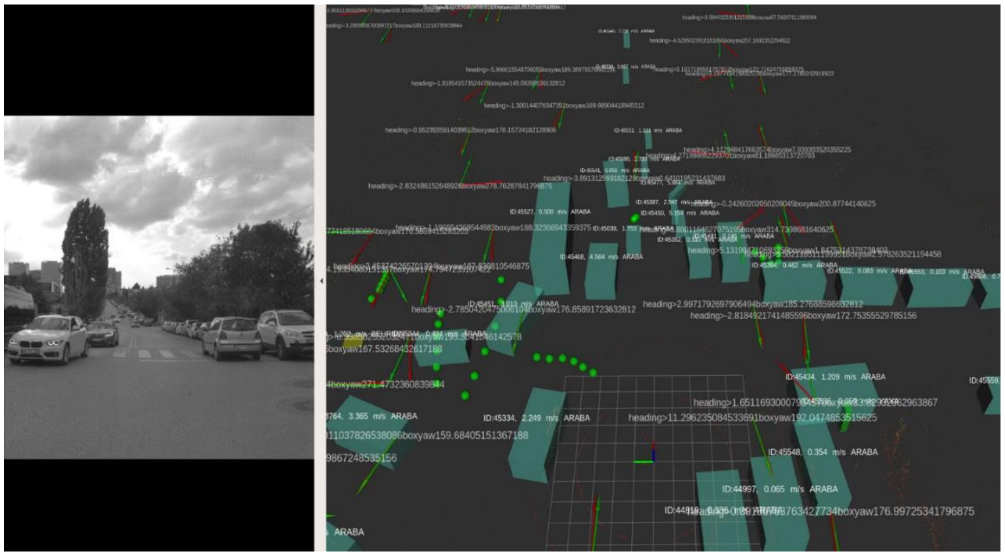


FIGURE 7. Visualization of dense environment trajectory prediction.

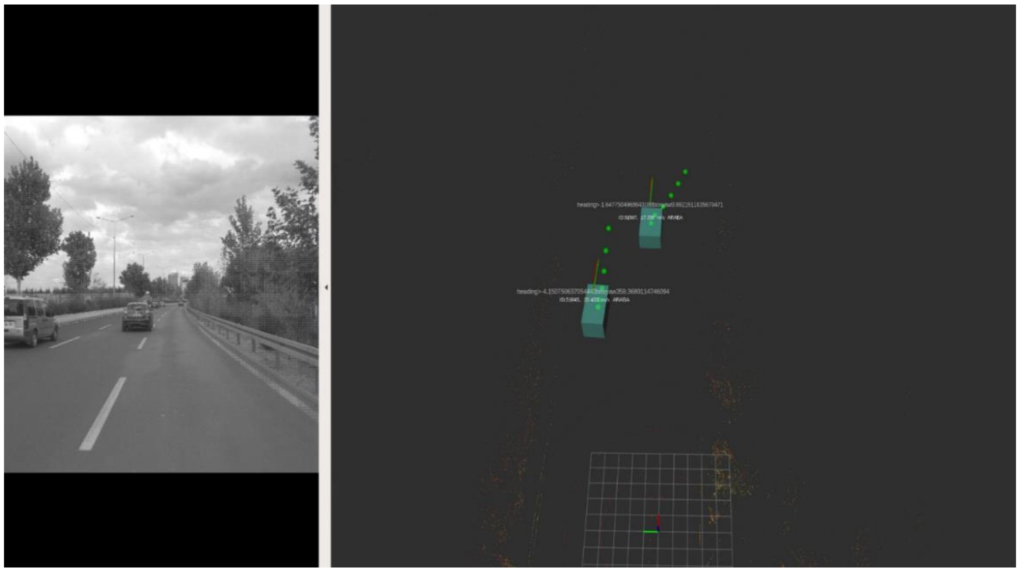


FIGURE 8. Visualization of multi vehicle trajectory prediction.

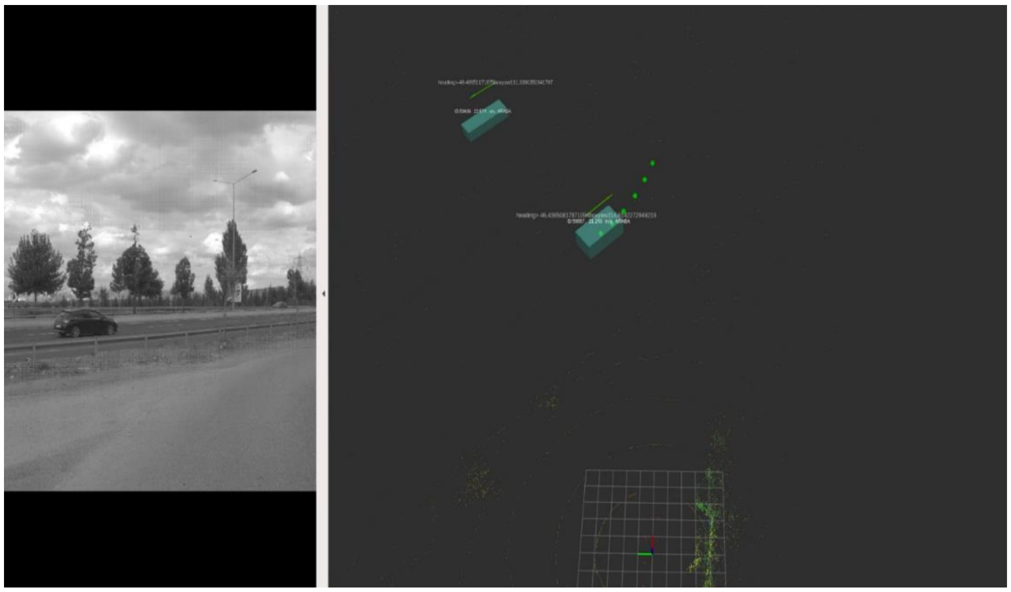


FIGURE 9. Visualization of vehicle-1 trajectory prediction.

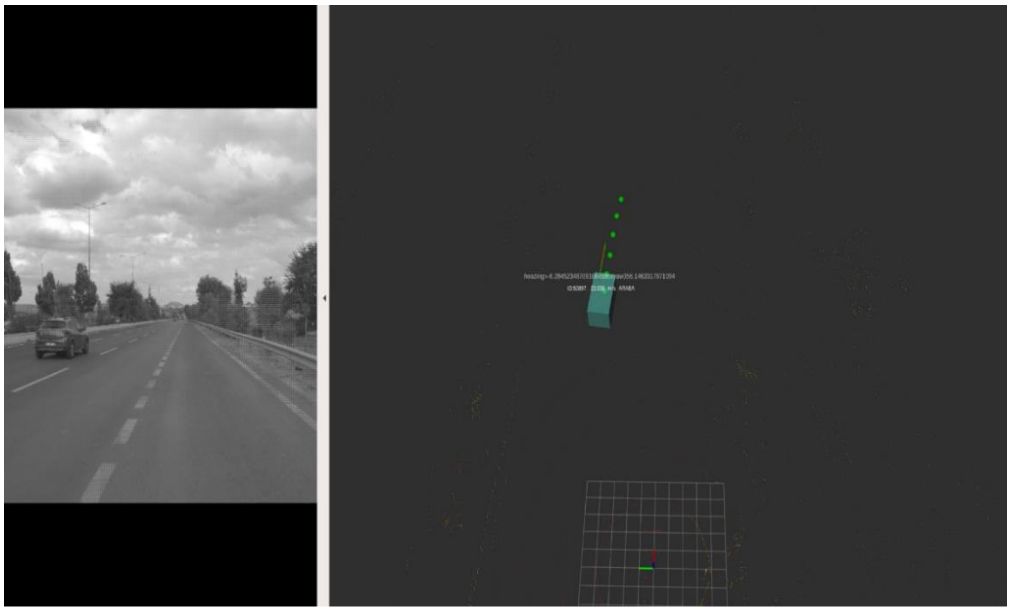


FIGURE 10. Visualization of vehicle-2 trajectory prediction.

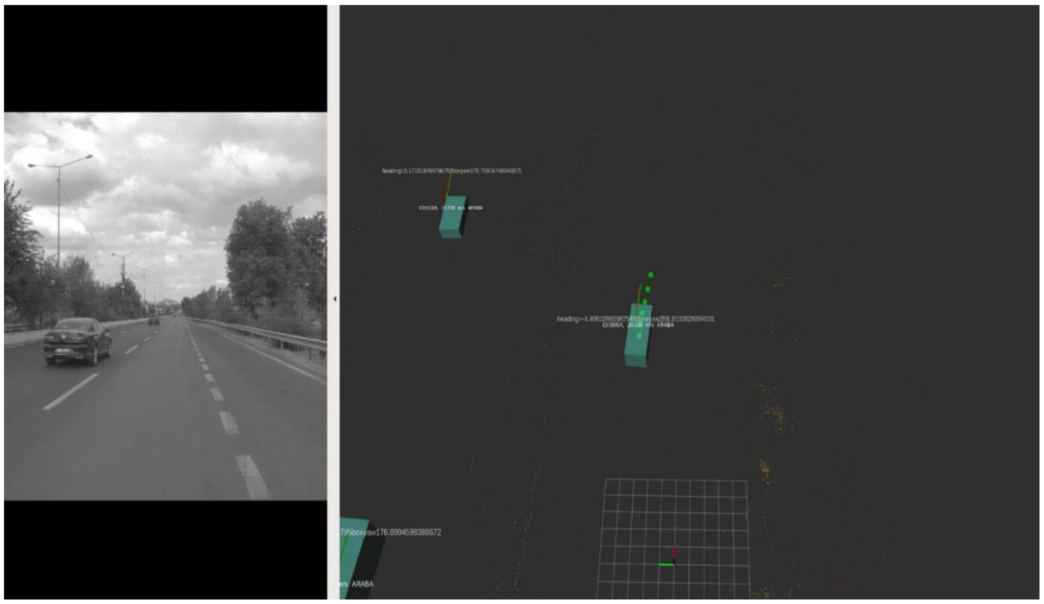


FIGURE 11. Visualization of vehicle-3 trajectory prediction.

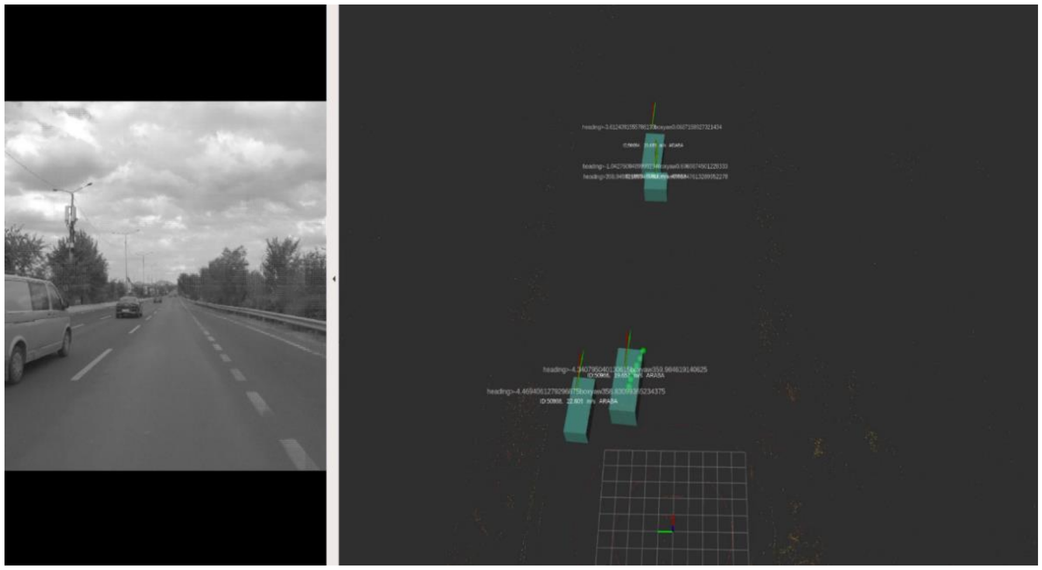


FIGURE 12. Visualization of vehicle-4 trajectory prediction.

TABLE 4. RMSE results of tested Rosbag.

Objects	***RMSE(m)***(1)				
	1s	***2s***	***3s***	***4s***	***5s***
Vehicle1	0.87	2.13	4.14	5.68	7.25
Vehicle2	0.92	2.09	4.28	5.82	6.43
Vehicle3	0.70	1.90	3.91	5.11	6.12
Vehicle4	1.20	2.42	4.72	6.23	8.04
Vehicle5	0.78	1.75	3.78	5.02	6.10
Vehicle6	1.45	2.10	3.99	4.97	7.05
Vehicle7	0.75	2.78	3.41	6.01	7.73
Vehicle8	1.28	2.35	4.66	5.28	6.87
Vehicle9	1.12	1.99	4.03	6.36	8.80
Overall	1.07	2.17	4.10	5.61	7.15

5. CONCLUSION

With the behavioral prediction module obtained as a result of this study, the probability of an autonomous vehicle making a mistake in the environment of moving objects has been significantly reduced. An end-to-end, real-time and robust behavioral prediction structure has been established from the detection module to the planning module, with contributions such as writing real-time inference to the current model, passing it to the ROS infrastructure, correcting or filtering faulty data.

From a broader perspective apart from transportation vehicles, this software can be used in any autonomous ground vehicle such as health care robots, agricultural robots, shuttle services etc. and it increases accuracy of planning module. In the future, it is planned to add ConvLSTM to the model for increasing accuracy and accelerate this model using TensorRT.

In the field of behavioral prediction, as it can be understood from the related work section, many models have been studied and continue to be studied. Although the studied models generally give successful results for the test sets of their own data-sets, their speed and performance decrease in dense environments such as urban areas. Successful results can be obtained by creating more complex data-sets and developing better model architectures.

In this study, trajectory prediction methods were examined, compared with each other, and as a result of these comparisons, graph structured structures were seen to be more successful. Afterwards, the data was collected and the content of the data was explained. The developed software is explained, tested and the results are expressed.

Once for all, an open-source behavioral prediction module for autonomous vehicle is developed. Although Trajectron++ is used as the model, there is no open source study for end-to-end integration of a model into an autonomous vehicle. The study is innovative in this aspect. The module works more efficiently and robustly due to the added history hold/drop structure and data filtering and correction features.

Author Contribution Statements The authors equally worked on the study. All authors read and approved the final copy of the manuscript.

Declaration of Competing Interests The authors declare that there is no conflict of interest regarding the publication of manuscript.

References

[1] Koubaa, A., Robot Operating System (ROS): The Complete Reference (Volume 2), Springer, 2017, https://doi.org/10.1007/978-3-319-54927-9.

[2] Hintjens, P., ZeroMQ: Messaging for Many Applications, O'REILLY, CA, 2013.

[3] Macenski, S., Foote, T., Gerkey, B., Lalancette, C., Woodall, W., Robot operating system 2: Design, architecture, and uses in the wild, *Sci. Robot.*, 7 (66) (2022), https://doi.org/10.48550/arXiv.2211.07752.

[4] Salzmann, T., Ivanovic, B., Chakravarty, P., and Pavone, M., Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data, *European Conference on Computer Vision*, 12363 (2020), 683-700, https://doi.org/10.1007/978-3-030-58523-5_40.

[5] Huang, Y., Du, J., Yang, Z., Zhou, Z., Zhang, L. and Chen, H., A survey on trajectory-prediction methods for autonomous driving, *IEEE Trans. Intell. Veh.*, 7 (3) (2022), https://doi.org/10.1109/TIV.2022.3167103.

[6] Gulzar, M., Muhammad, Y. and Muhammad, N., A survey on motion prediction of

pedestrians and vehicles for autonomous driving, *IEEE Access*, 9 (2021), 137957-137969, https://doi.org/10.1109/ACCESS.2021.3118224.

[7] Lin, C. F. and Ulsoy, A. G., Vehicle dynamics and external disturbance estimation for vehicle path prediction, *IEEE Trans. Control Syst. Technol.*, 8 (3) (2000), 508-518, https://doi.org/10.1109/87.845881.

[8] Lefèvre, S., Vasquez, D. and Laugier, C., A survey on motion prediction and risk assessment for intelligent vehicles, *ROBOMECH J.*, 1 (1) (2014), 1-14, https://doi.org/10.1186/s40648-014-0001-z.

[9] Schöller, C. Aravantinos, Lay, V. F. and Knoll, A., What the constant velocity model can teach us about pedestrian motion prediction, *IEEE Robot. Autom. Lett.*, 5 (2) (2020), 1696-1703, https://doi.org/10.48550/arXiv.1903.07933.

[10] Ammoun, S. and Nashashibi, F., Real time trajectory prediction for collision risk estimation between vehicles, *IEEE 5th International Conference on Intelligent Computer Communication and Processing*, (2009), 417-422, https://doi.org/10.1109/ICCP.2009.5284727.

[11] Schubert, R., Richter, E. and Wanielik, G., Comparison and evaluation of advanced motion models for vehicle tracking, *11th International Conference on Information Fusio*n, (2008), 1-6.

[12] Lytrivis, P., Thomaidis, G. and Amditis, A., Cooperative path prediction in vehicular environments, *11th International IEEE Conference on Intelligent Transportation Systems*, (2008), 803-808, https://doi.org/10.1109/ITSC.2008.4732629.

[13] Batz, T., Watson, K. and Beyerer, J., Recognition of dangerous situ ations within a cooperative group of vehicles, *IEEE Intelligent Vehicles Symposium*, (2009), 907-912, https://doi.org/10.1109/IVS.2009.5164400.

[14] Kumar, P., Perrollaz, M., Lefevre, S. and Laugier, C., Learning-based approach for online lane change intention prediction, *IEEE Intelligent Vehicles Symposium (IV)*, (2013), 797- 802, https://doi.org/10.1109/IVS.2013.6629564.

[15] Qiao, S., Shen, D., Wang, X., Han, N. and Zhu, W., A self-adaptive parameter selection trajectory prediction approach via hidden markov models, *IEEE Trans. Intell. Transp. Syst.*, 16 (1) (2015), 284-296, https://doi.org/10.1109/TITS.2014.2331758.

[16] Deng, Q. and Soffker, D., Improved driving behaviors prediction based on fuzzy logic-hidden markov model (fl-hmm), *IEEE Intelligent Vehicles Symposium (IV*), (2018), 2003-2008, https://doi.org/10.1109/IVS.2018.8500533.

[17] Gindele, T., Brechtel, S. and Dillmann, R., A probabilistic model for estimating driver behaviors and vehicle trajectories in traffic environments, *13th Int. IEEE Conf. Intell. Transp. Syst.*, (2010), 1625-1631, https://doi.org/10.1109/ITSC.2010.5625262.

[18] Lee, N., Choi, W., Vernaza, P., Chor, C. B., Torr, P. H. S. and Chandraker, M. K., DESIRE: distant future prediction in dynamic scenes with interacting agents, *IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), (2017), 2165-2174,

https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.233.

[19] Dai, S., Li, L. and Li, Z., Modeling vehicle interactions via modified LSTM models for trajectory prediction, *IEEE Access*, 7 (2019), 38287-38296, https://doi.org/10.1109/ACCESS.2019.2907000.

[20] Sun, L., Zhan, W. and Tomizuka, M., Probabilistic prediction of interactive driving behavior via hierarchical inverse reinforcement learning, *IEEE 21st International Conference on Intelligent Transportation Systems (ITSC)*, (2018), 2111-2117, https://doi.org/10.1109/ITSC.2018.8569453.

[21] Kuefler, A., Morton, J., Wheeler, T. and Kochenderfer, M., Imitating driver behavior with generative adversarial networks, *IEEE Intelligent Vehicles Symposium (IV)*, (2017), 204-211, https://doi.org/10.1109/IVS.2017.7995721.

[22] Choi, S., Kim, J. and Yeo, H., Trajgail: Generating urban vehicle trajectories using generative adversarial imitation learning, *Transp. Res. Part C Emerg. Technol.*, 128 (2021), 103091, https://doi.org/10.48550/arXiv.2007.14189.

[23] Wulfmeier, M., Ondruska, P. and Posner, I., Maximum entropy deep in verse reinforcement learning, (2015), https://doi.org/10.48550/arXiv.1507.04888.

[24] You, C., Lu, J., Filev, D. and Tsiotras, P., Advanced planning for autonomous vehicles using reinforcement learning and deep inverse reinforcement learning, *Robot. Auton. Syst.*, 114 (2019), 1-18, https://doi.org/10.1016/j.robot.2019.01.003.

[25] Jung, C. and Shim, D. H., Incorporating multi-context into the traversability map for urban autonomous driving using deep inverse reinforcement learning, *IEEE Robot. Autom. Lett.*, 6 (2) (2021), 1662-1669, https://doi.org/10.1109/LRA.2021.3059628.

[26] Geiger, A., Lenz P. and Urtasun, R., Are we ready for autonomous driving? The KITTI vision benchmark suite, *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, (2012), 3354-3361, https://doi.org/10.1109/CVPR.2012.6248074.

[27] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G. and Beijbom, O., nuScenes: A multimodal dataset for autonomous driving, *CVPR*, (2020), 11618-11628, https://doi.ieeecomputersociety.org/10.1109/CVPR42600.2020.01164.

[28] Chang, M.-F., Lambert, J., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D. and Hays, J., Argoverse: 3D Tracking and Forecasting With Rich Maps, *IEEE Conference on Computer Vision and Pattern Recognition*, (2019), 8748-8757, https://doi.org/10.1109/CVPR.2019.00895.

[29] Coifman, B. A critical evaluation of the next generation simulation (NGSIM) vehicle trajectory dataset, *Trans. Res. B Methodol.*, 105 (2017), 362-377, https://doi.org/10.1016/j.trb.2017.09.018.

[30] Caesar, H., Kabzan, J., Tan, K., nuPlan: A closed-loop ML-based planning benchmark for autonomous vehicles, *CVPR ADP3 workshop*, (2021), https://doi.org/10.48550/arXiv.2106.11810.

[31] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G. and Beijbom, O., nuScenes: A multimodal dataset for autonomous driving, *CVPR*, 2020.

[32] Enyen, Deep Trajectory Prediction, (2017), https://github.com/enyen/Deep-Trajectory-Prediction.

[33] Deo, N. and Trivedi, M. M., Convolutional social pooling for vehicle trajectory prediction, *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, (2018), 1549-15498, https://doi.org/10.1109/CVPRW.2018.00196.

[34] Lefkopoulos, V., Menner, M., Domahidi, A. and Zeilinger, M. N., Interaction-aware motion prediction for autonomous driving: A multiple model kalman filtering scheme, *IEEE Robotics and Automation Letters*, 6 (1) (2021), 80-87, https://doi.org/10.1109/LRA.2020.3032079.

[35] Deo, N., Rangesh, A. and Trivedi, M. M., How would surround vehicles move? a unified framework for maneuver classification and motion prediction, *IEEE Trans. Intell. Veh.*, 3 (2) (2018), 129-140, https://doi.org/10.1109/TIV.2018.2804159.

[36] Deo, N. and Trivedi, M. M., Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms, *IEEE Intelligent Vehicles Symposium (IV)*, (2018), 1179-1184, https://doi.org/10.1109/IVS.2018.8500493.

[37] Tang C. and Salakhutdinov, R. R., Multiple futures prediction, *Adv. Neural Inf. Process. Sys.*, 32 (2019), 15424-15434, https://doi.org/10.48550/arXiv.1911.00997.

[38] Deo, N. and Trivedi, M. M., Convolutional social pooling for vehicle trajectory prediction, *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, (2018), 1468-1476.

[39] Messaoud, K., Yahiaoui, I., Verroust-Blondet, A. and Nashashibi, F., Attention based vehicle trajectory prediction, *IEEE Trans. Intell. Veh.*, 6 (1) (2020), 175-185, https://doi.org/10.1109/TIV.2020.2991952.

[40] Li, X., Ying, X. and Chuah, M. C., Grip++: Enhanced graph-based interaction-aware trajectory prediction for autonomous driving, arXiv:1907.07792, (2019), https://doi.org/10.48550/arXiv.1907.07792.

[41] Zhao, Z., Fang, H., Jin, Z. and Qiu, Q., Gisnet: Graph-based information sharing network for vehicle trajectory prediction, *IEEE International Joint Conference on Neural Networks (IJCNN)*, (2020), 1-7, https://doi.org/10.48550/arXiv.2003.11973.

[42] Zhao, T., Xu, Y., Monfort, M., Choi, W., Baker, C., Zhao, Y., Wang, Y. and Wu, Y. N., Multi-agent tensor fusion for contextual trajectory prediction, *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (2019), https://doi.org/10.48550/arXiv.1904.04776.

[43] Wang, Y., Zhao, S., Zhang, R., Cheng, X. and Yang, L., Multi-vehicle collaborative learning for trajectory prediction with spatio-temporal tensor fusion, *IEEE Transactions on Intelligent Transportation Systems*, 23 (1) (2022), 236-248, https://doi.org/10.1109/TITS.2020.3009762.

[44] Saleh, K., Hossny, M. and Nahavandi, S., Long-term recurrent predictive model for intent prediction of pedestrians via inverse reinforcement learning, *Digital Image*

Computing: Techniques and Applications (DICTA), (2018), 1-8, https://doi.org/10.1109/DICTA.2018.8615854.

[45] Kuefler, A., Morton, J., Wheeler, T. and Kochenderfer, M., Imitating driver behavior with generative adversarial networks, *IEEE Intelligent Vehicles Symposium (IV)*, (2017), 204-211, https://doi.org/10.1109/IVS.2017.7995721.

[46] Wulfmeier, M., Rao, D., Wang, D. Z., Ondruska, P. and Posner, I., Large-scale cost function learning for path planning using deep inverse reinforcement learning, *Int. J. Rob. Res*., 36 (10) (2017), 1073-1087, https://doi.org/10.1177/0278364917722396.

[47] Fernando, T., Denman, S., Sridharan, S. and Fookes, C., Neighbourhood context embeddings in deep inverse reinforcement learning for predicting pedestrian motion over long time horizons, *IEEE/CVF International Conference on Computer Vision Workshops*, (2019), 1179-1187, https://doi.org/10.1109/ICCVW.2019.00149.

[48] Chang, M.-F., Lambert, J., Sangkloy,P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D. et al., Argoverse: 3d tracking and forecasting with rich maps, *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (2019), 8748-8757.

[49] Mercat, J., Gilles, T., El Zoghby, N., Sandou, G., Beauvois, D. and Gil, G. P., Multi-head attention for multi-modal joint vehicle motion forecasting, *IEEE International Conference on Robotics and Automation (ICRA)*, (2020), 9638-9644, https://doi.org/10.1109/ICRA40945.2020.9197340.

[50] Ngiam, J., Caine, B., Vasudevan, V., Zhang, Z., Chiang, H.-T. L., Ling, J., Roelofs, R., Bewley, A., Liu, C., Venugopal, A., et al., Scene transformer: A unified multi-task model for behavior prediction and planning, *International Conference on Learning Representations (ICLR)*, (2021), https://doi.org/10.48550/arXiv.2106.08417.

[51] Liu, Y., Zhang, J., Fang, L., Jiang, Q. and Zhou, B., Multimodal motion prediction with stacked transformers, *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (2021), 7577-7586, https://doi.org/10.1109/CVPR46437.2021.00749.

[52] Liang, M., Yang, B., Hu, R., Chen, Y., Liao, R., Feng, S. and Urtasun, R., Learning lane graph representations for motion forecasting, *European Conference on Computer Vision (ECCV)*, (2020), 541-556, https://doi.org/10.48550/arXiv.2007.13732.

[53] Gu, J., Sun, C. and Zhao, H., Densetnt: End-to-end trajectory prediction from dense goal sets, *IEEE/CVF International Conference on Computer Vision*, (2021), 15303-15312, https://doi.ieeecomputersociety.org/10.1109/ICCV48922.2021.01502.

[54] Zeng, W., Liang, M., Liao, R. and Urtasun, R., Lanercnn: Distributed representations for graph-centric motion forecasting, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (2021), 532-539, https://doi.org/10.1109/IROS51168.2021.9636035.

[55] Song, H., Luan, D., Ding, W., Wang, M. Y. and Chen, Q., Learning to predict vehicle trajectories with model-based planning, arXiv:2103.04027, (2021), https://doi.org/10.48550/arXiv.2103.04027.

http://communications.science.ankara.edu.tr

Commun.Fac.Sci.Univ.Ank.Series A2-A3
Volume 66, Number 1, Pages 26-44 (2024)
DOI: 10.33769/aupse.1320795
ISSN 1303-6009 E-ISSN 2618-6462

Research Article; Received: June 28, 2023; Accepted: August 31, 2023

USB-IDS-1 DATASET FEATURE REDUCTION WITH GENETIC ALGORITHM

Mustafa Veysel ÖZSARI[1], Şifa ÖZSARI[1], Ayhan AYDIN[1],
Mehmet Serdar GÜZEL[1]

[1]Ankara University, Computer Engineering Department, Ankara, 06830, TÜRKİYE

Abstract. Technology and online opportunities brought by technology are increasing day by day. Many transactions, from banking to shopping, can be done online. However, the abuse of technology is also increasing at the same rate. Therefore, it is very important to ensure the security of the network for data protection. The application of artificial intelligence-based approaches has also become popular in the field of information security. When the data collected for intrusion detection is examined, it is seen that there are many features. In this study, the features in the USB-IDS-1 dataset were reduced by genetic algorithm and its success was examined with various classifiers. Among the selected methods, there are decision trees, random forest, k-NN, Naive Bayes and artificial neural networks. Accuracy, sensitivity, precision and F1-score were used as metrics. According to the results obtained, it was seen that the genetic algorithm was quite successful in the Hulk and Slowloris data set, it was partially effective in the Slowhttptest data, but was not successful in the TCP set. However, the performance of the algorithms was poor as a result of using all features in Slowhttptest and TCP data.

1. Introduction

In the last 10 years, technology has progressed very rapidly. With this progress, many processes from health to defense, from online shopping to engineering applications have started to be done online. This has led to a proportional increase in the number of data entered. However, it has also increased in malicious attacks. Since great number of personal information is stored online, it is crucial to protect this data. One of the popular tools for information security is Intrusion Detection

Keywords. Intrusion detection, USB-IDS-1, genetic algorithm, machine learning.

ozsaris@ankara.edu.tr-Corresponding author; 0000-0002-0531-4645
veysel.ozsari.95@gmail.com; 0000-0002-1997-9971
ayaydin@ankara.edu.tr; 0000-0001-7938-0509
mguzel@ankara.edu.tr; 0000-0002-3408-0083.

Systems (IDS). IDS, in general terms, are devices or software that detect malicious attacks on systems.

As in many fields, studies are carried out on Artificial Intelligence (AI) and especially Machine Learning (ML) based approaches. AI was developed with inspiration from the learning process of human brain. Kaplan et al. [1] defined AI as the ability of a system to accurately interpret data, learn from such data, and use this information to achieve specific goals and tasks. ML is a sub-field of artificial intelligence and was first introduced by Arthur Samuel [2]. Today's ML applications are generally about developing a classifier with using the available data and then with these models to produce predictions for new (unseen) input. When the literature is examined, ML-based studies [3-7] in intrusion detection and KDD Cup99 [8], CAIDA [9], NSL-KDD [10] data sets can be given as examples for data prepared for IDS.

In presented paper, we conducted a study on the USB-IDS-1 [11] dataset introduced in 2021. Section 2 provides detailed information about this data set. It is a big data and has 83 attributes (features). The main problem encountered in processing such large-sized data is hardware inadequacy and time. If the features of the devices used are powerful, faster results can be produced. However, this is not always possible. In this study, we performed feature selection on the USB-IDS-1 dataset using Genetic Algorithm (GA) [12] and then carried out classification with these features.

The genetic algorithm developed by Holland is an optimization algorithm. The main object in optimization approaches is to select the best solution among all alternatives for a specific task. GAs is a widely used and simple optimization method in this field. Therefore, this algorithm was utilized for feature reduction. Thus, the features that have no effect on the classification were identified and eliminated, and a classification system was designed to predict the new data without using these features.

The rest of the article is organized as follows: In Section 2, detailed information about the dataset and algorithms used in the study is given. In Section 3, parameter settings, experiments and test results are explained. Section 4 is the results and the article is concluded.

2. Material and Methods

2.1. **Dataset.** There are various data sets that are widely used for academic studies and examinations on STS in the literature. In this research, we used the USB-IDS-1 dataset introduced by Catillo et al. [11] in 2021. This dataset consists of 83 features and 16 classes. These features are: “Flow ID”, “Src IP”, “Src Port”, “Dst IP”, “Dst Port”, “Protocol”, “Timestamp”, “Flow Duration”, “Total Fwd Packet”, “Total Bwd packets", "Total Length of Fwd Packet", "Total Length of Bwd Packet", "Fwd Packet Length Max", "Fwd Packet Length Min", "Fwd Packet Length Mean", "Fwd Packet Length Std", "Bwd Packet Length Max", "Bwd Packet Length Min", "Bwd Packet Length Mean", "Bwd Packet Length Std", "Flow Bytes/s", "Flow Packets/s", "Flow IAT Mean", "Flow IAT Std", "Flow IAT Max", "Flow IAT Min", "Fwd IAT Total", "Fwd IAT Mean", "Fwd IAT Std", "Fwd IAT Max", "Fwd IAT Min", "Bwd IAT Total", "Bwd IAT Mean", "Bwd IAT Std", "Bwd IAT Max", "Bwd IAT Min", "Fwd PSH Flags", "Bwd PSH Flags", "Fwd URG Flags", "Bwd URG Flags", "Fwd Header Length", "Bwd Header Length", "Fwd Packets/s", "Bwd Packets/s", "Packet Length Min", "Packet Length Max", "Packet Length Mean", "Packet Length Std", "Packet Length Variance", "FIN Flag Count", "SYN Flag Count", "RST Flag Count", "PSH Flag Count", "ACK Flag Count", "URG Flag Count", "CWR Flag Count", "ECE Flag Count", "Down/Up Ratio", "Average Packet Size", "Fwd Segment Size Avg", "Bwd Segment Size Avg", "Fwd Bytes/Bulk Avg", "Fwd Packet/Bulk Avg", "Fwd Bulk Rate Avg", "Bwd Bytes/Bulk Avg", "Bwd Packet/Bulk Avg", "Bwd Bulk Rate Avg", "Subflow Fwd Packets", "Subflow Fwd Bytes", "Subflow Bwd Packets", "Subflow Bwd Bytes", "FWD Init Win Bytes", "Bwd Init Win Bytes", "Fwd Act Data Pkts", "Fwd Seg Size Min", "Active Mean", "Active Std", "Active Max", "Active Min", "Idle Mean", "Idle Std", "Idle Max", "Idle Min", "Label".

The attributes "Flow ID", "Fwd Header Length", "Src IP", "Src Port", "Dst IP", "Dst Port", "Timestamp" are metadata and are not used for classification. These columns were removed from the dataset so that they do not affect the classification and feature selection result. In addition, rows with NaN values were excluded from the data set. Class labels also include the defense module, and the groups for defense are as follows:

1. Hulk-NoDefense
2. Hulk-Reqtimeout
3. Hulk-Evasive
4. Hulk-Security2
5. TCPFlood-NoDefense
6. TCPFlood-Reqtimeout
7. TCPFlood-Evasive
8. TCPFlood-Security2
9. Slowhttptest-NoDefense
10. Slowhttptest -Reqtimeout
11. Slowhttptest -Evasive
12. Slowhttptest -Security2
13. Slowloris-NoDefense

14. Slowloris-Reqtimeout
15. Slowloris-Evasive
16. Slowloris-Security2

In here, the part before – denotes the attack type, and the next part denotes the defense model. Attack and defense types are briefly explained as follows [11]:

- *Hulk: This type of attack generates a large number of unique requests. Thereby, it is intended to prevent the server from recognizing a pattern and filtering the attack. This makes it difficult to detect requests from the signature.*
- *TCPFlood: TCPFlood is a well-known DoS attack tool that is considered as a flood attack. In here, the attacker's requests lock the available ports on the server and cause TCP connections from legitimate clients to not be accepted.*
- *Slowhttptest: The tool used in this attack type allows the launch of slow DoS application layer attacks. HTTP connections can be extended in different ways. In the experiments for this dataset, Slowhttptest was used in "slowloris" mode, which sends incomplete HTTP requests to the target server.*
- *Slowloris: In this attack type, DoS attacks are produced by sending slow HTTP requests to the server. It uses low-bandwidth approaches that exploit a weakness in the TCP fragmentation management of the HTTP protocol.*
- *Reqtimeout: This mod_reqtimeout defense module is intended to protect the HTTP server from slow DoS attacks like Slowloris. This module allows determining the minimum data rate and timeouts required to keep a connection open.*
- *Evasive: This mod_evasive module has been developed to protect the server from attacks (such as Hulk) that try to make the server unusable by consuming the server's resources with a large number of requests. It monitors incoming requests and looks for suspicious IPs and similar activities. For example, events such as multiple requests within a short period of time or multiple requests per second for the same pages. If any of these events are detected, a 403-warning code will be responded to and the IP will be blacklisted for a certain period of time [13].*
- *Security2: The mod_security2 module, which is based on a set of rules related to known attack structures that can be obtained from free or pay-as-you-go repositories, acts as a kind of intrusion detection and prevention system.*

In the presented study, Hulk, TCP, Slowhttptest and Slowloris attack types were taken as a separate group. Because there is not an equal number of rows from all groups in the data set. Figure 1 shows the distribution of the categories.

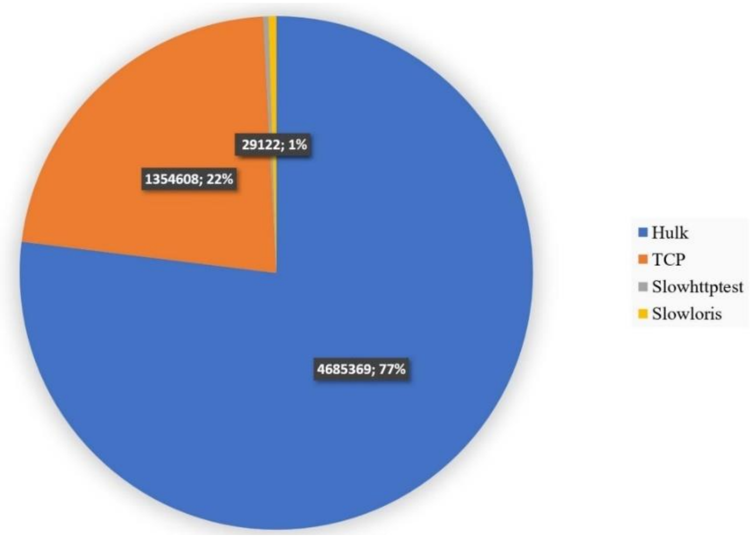


FIGURE 1. Data distribution.

In the literature, researches on USB-IDS-1 are quite few. First of the studies we examined is the [14]. In this paper, experiments were carried out by applying decision trees, random forest and deep neural network algorithms on the USB-IDS-1. Catillo et. al. performed the training process on a different dataset. They carried out the test on presented data set.

Another important study is [15]. A new approach for IoT security monitoring that combines deep autoencoder models with Explainable Artificial Intelligence (XAI) is presented by Kalutharage et al. It is aimed to verify the reliability and robustness of ML-based intrusion detection systems. The proposed method was tested using the USB-IDS-1 dataset.

2.2. **Machine Learning.** Today, machine learning, which has been used in many areas from health to social media, from banks to online shopping, is a system that develops itself according to the data it has acquired. ML algorithms are divided into 3 categories according to types [16]:

- *Supervised learning: They are ML algorithms in which learning is made by using the output data corresponding to the input data.*
- *Unsupervised learning: They are algorithms that process with using only input data. No output data is given to the systems operating in this way.*
- *Reinforced learning: In systems working with this method, the system is rewarded depending on its performance. According to this award, the algorithm updates itself.*

In this study, decision trees, random forest [17], k-Nearest Neighbor (KNN) [18], Navie Bayes and Artificial Neural Networks (ANN) algorithms were employed. All of them are supervised learning approaches.

Decision trees are a tree-like ML algorithm, which basically consists of decision nodes and leaf nodes. They are widely used for classification and regression tasks. In decision trees, data is divided into sub-parts in the form of nodes. The first of these division is the root node. It proceeds from the root node to the child nodes according to the "Yes" or "No" status. Likewise, in child nodes, according to "Yes" or "No", it moves forward to the next node until the leaf node is reached. Leaf nodes are nodes with classes (indicating the class of the data). Decision trees are widely used in IDS studies because they are applicable in complex data sets and have produced successful results [19].

Random forests (random decision forests) are a tree-based ML algorithm developed by Breiman [17]. They are widely used for problems such as classification or regression. A random forest makes classification or regression by creating a large number of decision trees during the training phase. A decision tree is created for each class and classification is made by considering the results of these sub-decision trees. An example for forest structure is shown in Figure 2.

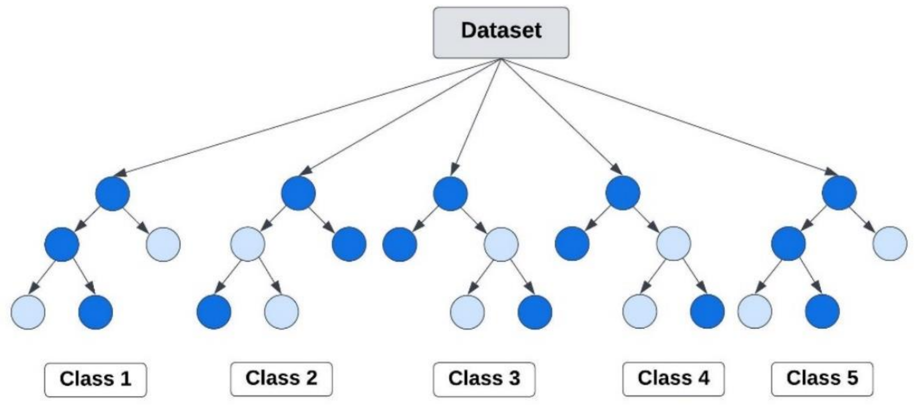


FIGURE 2. Random forest.

The KNN [18] algorithm is an algorithm that classifies a new input data according to its proximity to previous data. The number k represents the number of nearest neighbors to be checked. For example, if k is taken as 10, the class of the new incoming data is determined by looking at the class of 10 nearest neighbors. Here, the k parameter is important and directly affects the result. Various formulas such as Minkowski, Manhattan, Euclidean etc. are used when calculating proximity to neighbors.

Navie Bayes is an ML algorithm that runs based on the conditional probability calculation formula introduced by Thomas Bayes in 1812. This approach, which can also be applied to unbalanced data sets, basically makes a classification by calculating probability for each element. It is tried to determine the category of the new test samples given to the system with the probability operations performed on the data used to train the system. As with other ML-based approaches, the more data in this method, the more reliable the results.

Artificial neural networks, inspired by the neurons in the human brain, are a network of interconnected artificial neurons. Each neuron is connected to each other by weights and information is stored in these weights. A neuron produces output by processing the input it receives in various ways. This output is either the input of another neuron or the output of the net. A basic neural network structure is presented in Figure 3. In here, data is presented to the network from the input layer, then transmitted to the hidden layer(s), finally passing through the output layer to produce an output. The number of neurons in the input and output layer is determined depending on the input and output data of the problem. The number of hidden layers in the network is not fixed, it is adjusted according to the problem. The high number of hidden layers and neurons in the hidden layer may increase accuracy. However, it also causes high computational costs.

2.3. **Genetic Algorithm.** Optimization is the search for the most suitable solution among more than one solution according to a specific purpose. GA [12], a popular optimization algorithm, was first proposed by Holland in 1975. This approach is based on the theory of evolution and consists of 3 basic steps: selection, crossover and mutation. According to the theory of evolution, dominant individuals are passed on to the next generation, while weak individuals perish. Figure 4 shows the pseudocode of the GA.

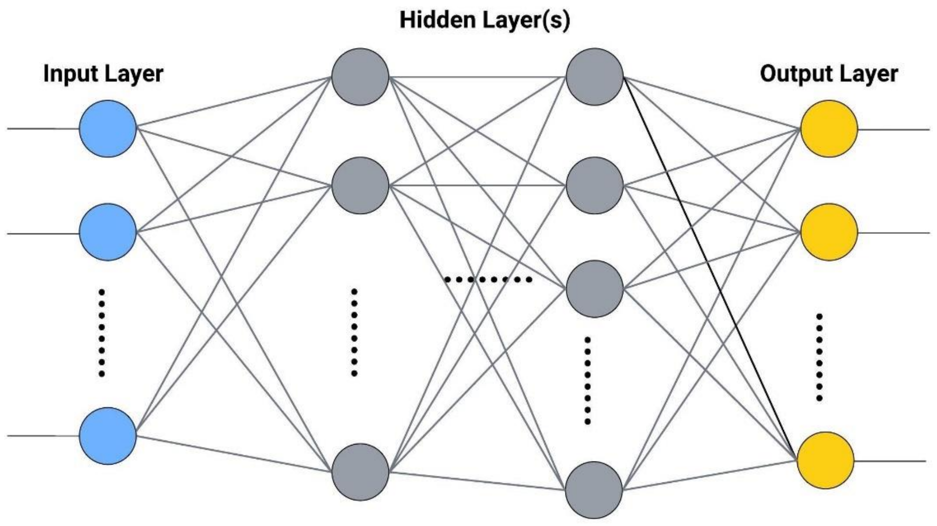
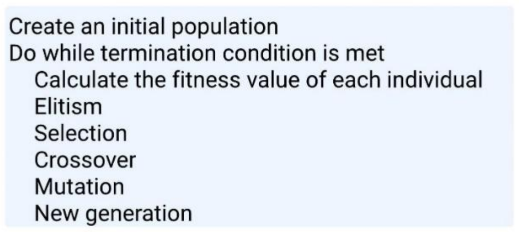


FIGURE 3. Basic ANN structure.



```
Create an initial population
Do while termination condition is met
    Calculate the fitness value of each individual
    Elitism
    Selection
    Crossover
    Mutation
    New generation
```

FIGURE 4. Pseudocode of the GA [20].

When we detail the pseudocode in Figure 4, first the population is randomly generated in accordance with the problem. Until the termination condition is met: The fitness value of individuals is computed. The fitness value is calculated using the objective function for problem. Individuals with the best fitness value (at a certain rate) are passed on to the next generation without any change. This process is called elitism and is not mandatory. Then, the individuals who will form the crossover pool are determined according to a specific selection method. Children are formed by using the crossover method determined between these individuals. The gene exchange process between two individuals is called crossover. Mutation process is applied to some of the offspring individuals. A mutation is a gene change in an individual and is generally applied at a low rate. The high rate of mutation can cause the loss of good individuals. Termination in GAs is done in various ways:

- When a certain number of iterations is reached, the run is terminated.

- ➢ The experiment is terminated when a targeted success rate is achieved.
- ➢ If the best fitness value does not change during a certain iteration, termination is made.

3. Results and Discussion

In this section, parameter settings, experiments and test results are given. All experiments were run in the Google Colaboratory (COLAB) [21] environment. COLAB is a product developed by Google Research where python codes can be written and run online. It allows automatic use of GPU and many libraries, so it is very practical for machine learning related studies [22].

3.1. **Metrics and Parameter Settings.** Accuracy rate, recall, precision and F1-score metrics were used to observe the performance of the methods. Formulas for these metrics are given in Equations 1, 2, 3 and 4, respectively

$$Accuracy\, rate = \frac{100*(TP+FP)}{TP+FP+TN+FN} \tag{1}$$

$$Recall = \frac{TP}{TP+FN} \tag{2}$$

$$Precision = \frac{TP}{TP+FP} \tag{3}$$

$$F1-score = \frac{2*\text{recall}*\text{precision}}{\text{recall}+\text{precision}} \tag{4}$$

For Equations 1, 2 and 3, TP corresponds to True Positive (classify healthy test as sick), FP False Positive (classify healthy sample as sick), TN True Negative (class healthy sample as healthy), and FN corresponds to False Negative (categories sick as healthy). The parameters of the methods used in the study were determined by the preliminary experiments and were adjusted as follows:

- ➢ *GA: The population size was set to 300, the iteration number to be 100, and the mutation rate to be 0.1. Elitism was made by transferring the best 2 individuals to the next generation. Tournament selection method, two-point crossover and value-changing mutation methods were used.*
- ➢ *Decision tree: The root node was determined using the Gini formula.*
- ➢ *Random forest: The number of trees is taken as 100.*
- ➢ *k-NN: Nearest 100 neighbors were checked.*

- *ANN: ADAM [23] was used as the solver and "identity" was selected as the activation function. 4 hidden layers (100-500 nodes each) were added to the net.*

In the tournament selection technique, the individual with the highest fitness value is taken from two randomly selected individuals. The selection process is performed in this way until the parent individuals to be used in the crossover are completed. In a two-point crossover, the genes of two parents are exchanged between two points. Two random points are chosen and the genes between these two points are exchanged between the two parent individuals. In the value-changing mutation technique, the value of a randomly selected gene of the individual is changed. While selecting the individual to be mutated, a random number between 0 and 1 is first generated for this individual. If this number is greater than the mutation rate, this individual is mutated. If it is small, no mutation is applied to the individual.

3.2. **Experiments.** After parameter settings were completed, classification was performed without making any feature selection at first. In other words, algorithms made classification using all attributes. Then, the performance of GA in feature reduction was examined. Firstly, separate experiments were carried out for each attack type. The main reason for conducting experiments in this way is the number of data, and the data distribution is shown in Figure 1. As can be seen from Figure 1, the data numbers of the classes are quite unbalanced. Data distribution is very important in machine learning-based approaches. The unbalanced data can cause the algorithms to make wrong inferences. For example, the Hulk group is dominant and will reduce the impact of other categories in the classification. For this reason, data were taken equally for each class, taking into account its own count of data, and separate experiments were carried out. Then, in order to observe the performance of the approaches in a way that all groups were included, the same experiments were performed again by taking equal amount of data from each group based on the lowest data count. The results of only 10 features were presented for experimentation with all attributes. Both the not enough number of data and the quadrupling of the number of classes affected the results negatively for 5 features and very poor results were obtained. Therefore, in Table 5, the outputs for 5 attributes were not included, only the results for 10 features were mentioned.

In feature selection part, firstly, the number of features was reduced to 5. When poor results were obtained in 5, a performance of the algorithms was examined for 10. A population was formed in accordance with GA as shown in Figure 5. In here, sample individuals of reducing the count of features to 5 are shown. Therefore, an individual consists of 5 genes. Each gene is a number which correspond to an attribute.

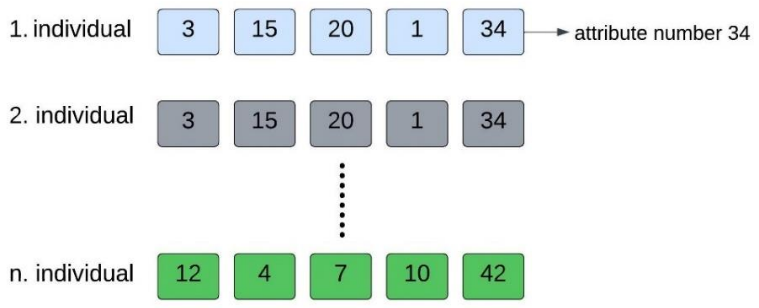


FIGURE 5. Population.

The fitness value is calculated for each individual, then crossover is applied to the individuals selected by the tournament selection method. An example crossover is shown in Figure 6. The genes (in the range indicated by lines) of the two parent individuals in Figure 6 (a) are replaced, and two new offspring individuals in (b) are obtained. Then the mutation stage is passed. A new population is obtained after the value-changing mutation process is completed. These operations (elitism, crossover, mutation) are performed sequentially up to 100 iterations. With the termination of the algorithm, the most successful individual is taken as the solution of the problem.

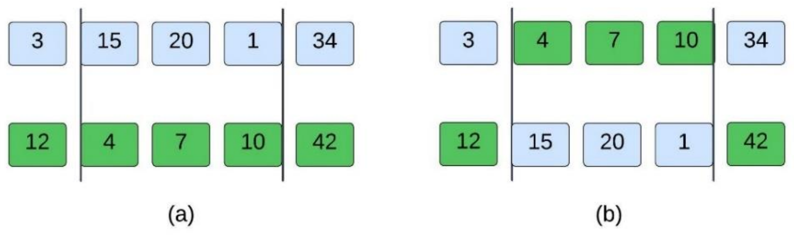


FIGURE 6. Crossover (a) parents (b) children.

After the completion of the experiments, the performance examination was carried out. In Tables 1, 2, 3 and 4, the results obtained with feature reduction and without feature reduction for Hulk, TCP, Slowhttptest, Slowloris are given respectively. In Table 5, the outputs of the experiments in which all groups were included are presented.

TABLE 1. Experiments for Hulk.

Features	**Classifier**	**Accuracy rate**	**Recall**	**Precision**	**F1-score**
5	Decision tree	0,71	0,72	0,72	0,72
All	Decision tree	0,72	0,73	0,73	0,73
5	Random forest	**0,72**	**0,73**	**0,73**	**0,73**
All	Random forest	**0,73**	**0,73**	**0,74**	**0,74**
5	k-NN	0,65	0,65	0,66	0,65
All	k-NN	0,66	0,66	0,68	0,66
5	Bayes	0,71	0,72	0,72	0,72
All	Bayes	0,72	0,73	0,73	0,73
5	ANN	0,62	0,62	0,64	0,61
All	ANN	0,64	0,63	0,65	0,63

When Table 1 is examined, it is seen that the random forest algorithm gives the best result in all metrics in terms of both taking all features and feature reduction. It is seen that GA is successful in feature reduction with results of 0.72 and above. 5 features in terms of their effect on classification accomplishment are:

- “Down/up ratio”
- “Fwd IAT Min”
- “Bwd IAT Mean”
- “Bwd IAT Max”
- “Bwd Packet Length Mean”

When other approaches are examined, it is seen that the decision tree and Bayesian are as successful as the random forest. Next comes k-NN and the least effective method is ANN. When all values in the table are examined, results of 61% and above are acceptable.

When Table 2 is analyzed, it is seen that the algorithms could not produce effective results in terms of both taking all features and feature reduction. Values of 0.25 and below are not acceptable. It is noteworthy that there is a decrease of up to 0.1 in the F1-score. All methods yielded more or less the same outputs. This situation is thought to be caused by the data set rather than the algorithms. However, it can be deduced from the table that the decision tree and random forest algorithm outperform other algorithms.

TABLE 2. Experiments for TCP.

Features	**Classifier**	**Accuracy rate**	**Recall**	**Precision**	**F1-score**
5	Decision tree	0,25	0,25	0,25	0,25
10	Decision tree	0,25	0,26	0,25	0,25
All	Decision tree	0,25	0,25	0,25	0,25
5	Random forest	0,26	0,26	0,26	0,26
10	Random forest	0,25	0,25	0,25	0,25
All	Random forest	0,25	0,25	0,25	0,25
5	k-NN	0,25	0,25	0,25	0,25
10	k-NN	0,25	0,26	0,25	0,14
All	k-NN	0,25	0,25	0,25	0,25
5	Bayes	0,26	0,27	0,26	0,15
10	Bayes	0,26	0,29	0,26	0,14
All	Bayes	0,25	0,26	0,25	0,15
5	ANN	0,25	0.16	0,25	0,1
10	ANN	0,25	0,25	0,19	0,18
All	ANN	0,25	0,25	0,25	0,23

TABLE 3. Experiments for Slowhttptest.

Features	**Classifier**	**Accuracy rate**	**Recall**	**Precision**	**F1-score**
5	Decision tree	0,39	0,39	0,43	0,4
10	Decision tree	**0,52**	**0,53**	**0,55**	**0,54**
All	Decision tree	**0,53**	**0,54**	**0,57**	**0,55**
5	Random forest	0,4	0,4	0,43	0,41
10	Random forest	0,5	0,5	0,53	0,51
All	Random forest	0,51	0,54	0,55	0,54
5	k-NN	0,4	0,51	0,43	0,45
10	k-NN	0,42	0,52	0,45	0,47
All	k-NN	0,38	0,49	0,41	0,44
5	Bayes	0,38	0,5	0,41	0,33
10	Bayes	0,37	0,41	0,39	0,32
All	Bayes	0,35	0,23	0,39	0,27
5	ANN	0,38	0,5	0,41	0,34
10	ANN	0,35	0,25	0,38	0,27
All	ANN	0,37	0,55	0,4	0,34

When Table 3 is interpreted, it is concluded that the best result is obtained from the decision tree in classification with all attributes. Approaches other than random forest was not very successful. When the remaining algorithms are ranked, k-NN, ANN and finally Bayes come.

When the number of features is reduced to 5, the results are again quite low. k-NN gave the best outcomes in 5 attributes. The random forest algorithm has more or less the same outputs. However, it is seen that other algorithms are ineffective with 40% and below results. Here again, Bayes (with a slight difference from ANN) has been the most abortive algorithm, as in all attributes.

The number of features was reduced to 10 and the experiments were repeated. In 10, outputs closer to the experiments with all features were obtained. Although the outcomes are not very good, there is not enough amount of data for Slowhttptest. Therefore, results of 50% and above are acceptable. The best approach was taken as the decision tree. When other approaches are analyzed, it can be concluded that they provide similar performance in random forest. Next comes the k-NN algorithm, Bayes and finally ANN. Contrary to all features and 5 features, Bayes was more effective here than ANN. 10 most effective features selected by GA in classifying Slowhttptest are:

- “Bwd IAT Mean”
- “Flow Bytes/s”
- “RST Flag Count”
- “Bwd IAT Max”
- “Fwd Packet Length Min”
- “Total Fwd Packet”
- “Flow Duration”
- “Bwd PSH Flag”
- “Active Min”
- “FWD Init Win Bytes”

When Table 4 is evaluated, it is deduced that the most effective algorithms are decision tree and random forest when all metrics are taken into account. Values of 96% and above are quite good results. When the outcomes of other approaches are examined, k-NN, Bayes and finally ANN come respectively.

When the experimental results for feature selection are examined, the random forest algorithm comes to the fore, similar to other experiments. Likewise, the success of the decision tree draws attention. Considering the 0.95 and above performance, it is deduced that the GA can predict the 5 most effective features in classification:

- “Bwd IAT Total”
- “Active Max”
- “Bwd IAT Max”
- “Fwd IAT Std”
- “Bwd Packet Length Std”

TABLE 4. Experiments for Slowloris.

Features	**Classifier**	**Accuracy rate**	**Recall**	**Precision**	**F1-score**
5	Decision tree	0,96	0,94	0,94	0,94
All	Decision tree	**0,97**	**0,96**	**0,96**	**0,96**
5	Random forest	**0,97**	**0,96**	**0,96**	**0,96**
All	Random forest	**0,97**	**0,96**	**0,95**	**0,96**
5	k-NN	0,7	0,5	0,61	0,54
All	k-NN	0,7	0,5	0,61	0,54
5	Bayes	0,62	0,41	0,5	0,4
All	Bayes	0,61	0,5	0,33	0,37
5	ANN	0,62	0,41	0,5	0,4
All	ANN	0,55	0,33	0,4	0,31

After the decision tree and random forest comes k-NN, then ANN and finally Bayes. In Figure 7, the accuracy rates before GA application (with all attributes) and accuracy rates after GA application (with feature selection) are shown according to the values in Tables 1, 2, 3 and 4. Here, considering all the outputs, it can be said that the GA was successful.

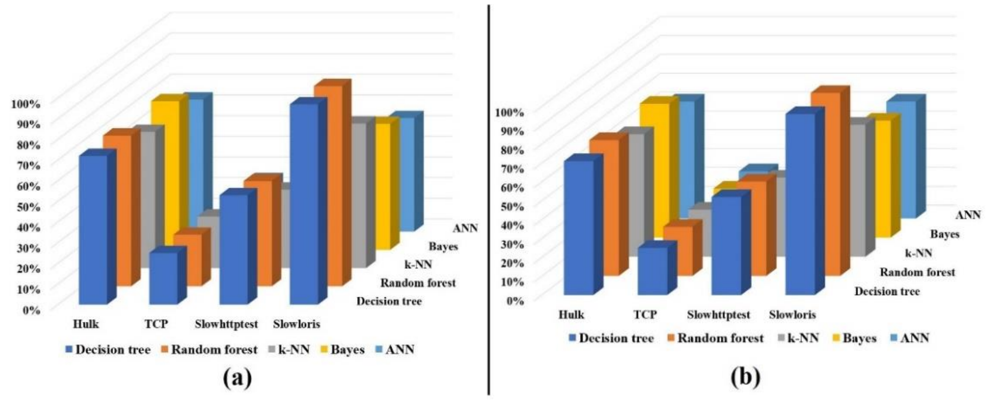


FIGURE 7. Accuracy rate (a) without GA (including all features) (b) with GA.

TABLE 5. Experimental results for all categories.

Features	**Classifier**	**Accuracy rate**	**Recall**	**Precision**	**F1-score**
10	Decision tree	0,54	0,58	0,56	0,55
All	Decision tree	0,58	0,61	0,6	0,59
10	Random forest	**0,55**	**0,59**	**0,57**	**0,56**
All	Random forest	**0,58**	**0,64**	**0,6**	**0,58**
10	k-NN	0,45	0,44	0,44	0,38
All	k-NN	0,48	0,42	0,46	0,41
10	Bayes	0,44	0,35	0,42	0,32
All	Bayes	0,41	0,34	0,39	0,28
10	ANN	0,28	0,23	0,25	0,21
All	ANN	0,43	0,41	0,4	0,33

When the experiments with all groups from Table 5 are examined, it is inferred that the random forest algorithm is the best algorithm in both all features and 10 features, as in the results separately. The decision tree also produced approximate results with the random forest. While there is k-NN, ANN and Bayes order in the remaining algorithms in terms of all attributes, the ranking in terms of 10 attributes is k-NN, Bayesian and ANN. The 10 most effective features estimated (for the random forest classifier) are as follows:

- “Bwd Header Length”
- “Fwd Segment Size Avg”
- “Fwd IAT Total”
- “URG Flag Count”
- “Bwd Packet Length Min”
- “Bwd Packet Length Mean”
- “Bwd IAT Std”
- “Fwd IAT Mean”
- “Bwd PSH Flags”
- “Bwd IAT Min”

It is expected that the results are in average values when the experiments are considered separately. The accuracy rate is lowered due to Slowhttptest and especially TCP. In separate tests for TCP, outputs are around 25%. The poor results in separate experiments (although the test was performed with more data) will cause much lower values when the number of classes increases and the number of data decreases. In the experiments for Slowloris, it was observed that the results were quite good, although there were few samples. Unlike TCP, this category will increase the overall accuracy rate.

4. Conclusion

Technological developments have brought many innovations that make life easier, as well as operations that are used for malicious purposes. It is very important to keep data safe against the online fraud. Therefore, strict precautions are taken and systems are developed to prevent attacks. The effective performance of artificial intelligence-based approaches in making inferences has increased their applicability to problems in real life. Machine learning is a subfield of artificial intelligence and its algorithms are widely utilized.

In this study, classification was made by applying decision tree, random forest, k-NN, Bayesian and ANN approaches on the USB-IDS-1 dataset, which was prepared for attack detection. Then, feature reduction was done with GA and its performance was evaluated. The dataset contains rows of 83 attributes belonging to 4 different attack types (Hulk, TCP, Slowhttptest, Slowloris). However, the data ratio between groups is highly uneven (Figure 1). If experiments are performed in this way, the algorithms will classify according to the dominant group. In this case, it causes misinterpretation of performances. Therefore, separate experiments were conducted for each group. An equal number of lines were taken from each group, taking into account the distribution within itself. Nevertheless, the performances of all groups and algorithms were examined. For this, an equal number of samples (according to the lowest amount of data.) were taken from each class and experiments were carried out.

In the light of experimental results, the decision tree and random forest algorithm were prosperous in the Hulk and Slowloris dataset. Especially in Slowloris, they performed quite well with 95% and above outputs. In general, it is concluded that these two algorithms are more effective than the others in all groups. In the Slowhttptest data, these two algorithms showed average performance, however the other methods again yielded worse results. All algorithms have failed to classify TCP data in terms of both full features and reduced features. When the results including all groups were examined, it was seen that the best values were between 55% and 64%. This is an expected outputs considering the separate classifications.

In future studies, experiments can be performed with different optimization algorithms (Particle Swarm Optimization (PSO) [24], Artificial Bee Colony (ABC) [25], etc.) and different data sets. In addition, the performance of different classification methods and deep learning approaches (Convolutional Neural Networks (CNN)), which are advanced versions of ANN, can be evaluated.

Author Contribution Statements The authors contributed equally to this study.

Declaration of Competing Interests The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Kaplan, A., Haenlein, M., Siri, Siri, in my hand: Who"s the fairest in the land? on the interpretations, illustrations, and implications of Artificial Intelligence, *Bus. Horiz.*, 62 (1) (2019), 15-25, https://doi.org/10.1016/j.bushor.2018.08.004.

[2] Samuel, A. L., Some studies in machine learning using the game of checkers, *IBM J. Res. Dev.*, 3 (3) (1959), 210-229, https://doi.org/10.1147/rd.33.0210.

[3] Aburomman, A. A., Reaz, M. B. I., Ensemble of binary SVM classifiers based on PCAand LDA feature extraction for intrusion detection, *Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, (2016), 636-640.

[4] Al-Jarrah, O. Y., Al-Hammdi, Y., Yoo, P. D., Muhaidat, S., Al-Qutayri, M. Semi-supervised multi-layered clustering model for intrusion detection, *Digit. Commun. Netw.*, 4 (4) (2018), 277-286.

[5] Al-Yaseen, W. L., Othman, Z. A., Nazri, M. Z. A. Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system, *Expert Syst. Appl.*, 67 (1) (2017), 296-303.

[6] An, X., Su, J., Lü, X., Lin, F., Hypergraph clustering model-based association analysis of DDOS attacks in fog computing intrusion detection system, *EURASIP JWCN*, 249 (1) (2018), 1-9.

[7] Belavagi, M. C., Muniyal, B., Performance evaluation of supervised machine learning algorithms for intrusion detection, *Procedia Comput. Sci.*, 89 (1) (2016), 117-123.

[8] KDD, The 1999 KDD intrusion detection, 1999, http://kdd.ics.uci.edu/databases/kddcup99/task.html.

[9] Hick, P., Aben, E., Claffy, K., Polterock, J., The CAIDA DDoS attack 2007 dataset, 2007.

[10] Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A. A., A detailed analysis of the KDD CUP 99 data set, in 2009 *CISDA*, (2009), 1-6.

[11] Catillo, M., Del Vecchio, A., Ocone, L., Pecchia, A., Villano, U., USB-IDS-1: a public multilayer dataset of labeled network flows for IDS evaluation, *51st Annual IEEE/IFIP DSN-W*, (2021), 1-6, https://doi.org/10.1109/DSN-W52860.2021.00012.

[12] Holland, J. H., Genetic algorithms, *Sci. Am.*, 267 (1) (1992), 66-73.

[13] Catillo, M., Pecchia, A., Villano, U., Measurement-based analysis of a DoS defense module for an open source web server, *Testing Software and Systems: 32nd IFIP WG 6.1 International Conference, ICTSS,* (2020), 121-134.

[14] Catillo, M., Del Vecchio, A., Pecchia, A., Villano, U., Transferability of machine learning models learned from public intrusion detection datasets: the CICIDS2017 case study, *Softw. Qual. J.*, (2022), 1-27.

[15] Kalutharage, C. S., Liu, X., Chrysoulas, C., Explainable AI and deep autoencoders based security framework for IoT network attack certainty, *Lect. Notes Comput. Sci.*, (2022), 13745, https://doi.org/10.1007/978-3-031-21311-3_8.

[16] Russell, S. J., Norvig, P., Artificial Intelligence a Modern Approach, Pearson Education, Inc., New York, 2010.

[17] Breiman, L., Random forests, *Mach. Learn.*, 45 (2001), 5-32.

[18] Cover, T., Hart, P., Nearest neighbor pattern classification, *IEEE Trans. Inf. Theory*, 13 (1) (1967), 21-27.

[19] Li, X., Ye, N., Decision tree classifiers for computer intrusion detection, *In Real-Time System Security*, (2003), 77-93.

[20] Ozsari, S., Uguz, H., Hakli, H., Implementation of meta-heuristic optimization algorithms for interview problem in land consolidation: A case study in Konya/Turkey, *Land Use Policy*, 108 (2021), 105511.

[21] Google colab., (2023). Available: https://research.google.com/colaboratory/faq.html. [Accessed: May 2023].

[22] Ozsari, S., Yapicioglu, F. R., Yilmaz, D., Kamburoglu, K., Guzel, M. S., Bostanci, G. E., Acici, K., Asuroglu, T., Interpretation of magnetic resonance images of temporomandibular joint disorders by using deep learning, *IEEE Access*, 11 (2023), 49102-49113, https://doi.org/10.1109/ACCESS.2023.3277756.

[23] Kingma, D. P., Jimmy, Ba., Adam: a method for stochastic optimization, arXiv:1412.6980, 2014.

[24] Kennedy, J., Eberhart, R., Particle swarm optimization, *Proceedings of IEEE International Conference on Neural Networks*, 4 (1995), 1942-1948, https://doi.org/10.1109/ICNN.1995.488968.

[25] Karaboga, D., An idea based on honey bee swarm for numerical optimization, Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

http://communications.science.ankara.edu.tr

Commun.Fac.Sci.Univ.Ank.Series A2-A3
Volume 66, Number 1, Pages 45-63 (2024)
DOI: 10.33769/aupse.1301383
ISSN 1303-6009 E-ISSN 2618-6462

Research Article; Received: May 23, 2023; Accepted: September 08, 2023

A PERFORMANCE COMPARISON OF UTILITY FUNCTIONS FOR GAME THEORY BASED WEAPON-TARGET ASSIGNMENT

Oğuzkan AKBEL[1] and Aykut KALAYCIOĞLU[1]

[1]Electrical-Electronics Engineering, Ankara University, Ankara, TÜRKİYE

ABSTRACT. The weapon-target assignment problem has been considered as an essential issue for military applications to provide a protection for defended assets. The goal of a typical weapon-target assignment problem is to maximize the expected survivability of the valuable assets. In this study, defense of naval vessels that encounter aerial targets is considered. The vessels are assumed to have different types of weapons having various firepower and cost as well as the incoming targets may have different attack capabilities. In a typical scenario, in addition to protecting assets, it is also desirable to minimize the cost of weapons. Therefore, an asset-based static weapon-target assignment problem is considered in order to both maximize the expected survivability of the assets and minimize the weapon budget. Thus, a co-operative game theory based solution is proposed which relates the utilities of the individuals to the global utility and reach the Nash equilibrium.

1. Introduction

Weapon target assignment (WTA) has long been a fundamental issue in the military domain. Eliminating targets that attack valuable assets is a very complex problem and traditionally needs to be solved by command and control officers. However, due to the increasing number of threats and the complexity of the problem, an automated system is required to help the command and control officer make the right decision. WTA problem is closely related with the threat evaluation procedure that provides the information about the intents and capabilities of the incoming targets. The determination of which weapon to assign to incoming targets by the defense unit depends on knowledge such as the asset targeted by the threat and the destruction

Keywords. Game, weapon, assignment, optimization, utility.

o_akbel@windowslive.com-Corresponding author; 0009-0002-5289-1549
kalaycioglu@ankara.edu.tr; 0000-0001-8291-9958.

capability of the threat. Therefore, defense unit performs the WTA under the assumption of perfect information gathered by the threat evaluation [1].

Weapon assignment to incoming targets can be discussed in several aspects in military operations. In addition to the ground-based air defense including airbases, factories or valuable areas, maximizing the expected survivability of the vessels as a defended asset in a maritime environment is also considered in the literature [1]. Besides, static case considers the single assignment of weapons of the defending units to incoming threats, whereas a continuum of several stages situation, also called as shoot-look-shoot, is considered in the dynamic case. Moreover, target-based WTA models are also discussed as a special case of the asset-based models. A summary of several solutions for single or multiple objectives for the WTA problem is discussed in [2].

Employing game theoretical solution for WTA problem has several benefits. A game-theoretical vehicle-target assignment problem is discussed in [3] by aligning the individual utility functions to a global utility function with different aspects. Authors state that despite limited communication capabilities in an ambiguous environment, the vehicles act as individually logical units and can operate and decide by themselves.

WTA problem is discussed in the literature with different aspects. In [4], Kline et. al. discussed the evolution of WTA and analyzed and compared exact algorithms and heuristic algorithms such as very large-scale neighborhood search or genetic algorithm. In [5], a known solution of an optimization problem and a game theoretic approach have been compared and authors state that their game theory based solution have similar results with the first fit decreasing and best fit decreasing algorithms.
In [6], Karasakal considers the maximization of the defense of a valuable asset by a number of escorting vessels. In a recent study [7], authors discuss asset-based multi-objective WTA problem and show that multi-objective evolutionary algorithm based on decomposition has an effective performance. In [8], there is another research which uses the game theory and fuzzy logic as a hybrid system on decision making process. In the literature, there are also examples for multi-objective weapon target assignments which are solved by empirical approaches. As a comprehensive survey [9] explains several algorithms for WTA problem. In [10], Şahin and Leblebicioğlu present an approach as a fuzzy classification problem. They use a rule-based fuzzy classifier for weapon target assignment.

In this study, a static WTA problem for naval vessels is considered in terms of both maximizing the expected survivability of the defended assets and minimizing the cost of using weapons. All vessels are considered as valuable assets having different types of weapons with various firepower and cost. Besides, the capabilities of the attacking air threats are various. Therefore, an efficient weapon-target assignment is required for assets to protect both themselves and each other. The proposed solution relies on a co-operative game. Each vessel has its individual utility that needs to be maximized. Besides, these individual utility functions are linked with a global utility function by an alignment function and while each individual tries to increase its utility function, individuals also contribute to the global utility function. Therefore, Nash equilibrium is obtained when the individuals reach to the point where there is no improvement for their utility values, which is considered as the optimal assignment for each vessel. There are three different utility functions given in [3] on the performance of the game theoretical WTA assignments. As discussed in [3], vessel's utility functions are forced to align with the global utility function to reach an optimization at the global utility function. An alignment function is used to align the utilities. Since a dynamic range restriction approach is utilized in this study, a range restricted utility function is crucial. However, due to the overlapping regions of the weapons, the alignment cannot be obtained. We employ equally shared utility (ESU), wonderful life utility (WLU), and identical interest utility (IIU) functions along with the range restricted utility (RRU) function to compare the performance of the discussed utility function's combinations. The results show that not all combinations of the different utility functions yield an optimal solution.

In section 2, system model is given. In section 3, the theoretical background is explained. Simulation parameters are given in section 4. In section 5, the results of the simulations are shown. Finally, in section 6 the conclusions that we achieve are explained.

2. System Definition

In asset-based WTA problem, we assume that there are M weapons, N targets, and K assets. The probability p_{ij} is the probability that weapon i kills the target j whereas π_{jk} is the probability that target j completely destroys the asset k. The main target of the WTA problem is to increase the survivability probability of the assets. Also, we assign different values for each asset k, represented with ω_k. The objective function of the asset-based model to be maximized is as follows.

$$\sum_{k=1}^{K} \omega_k \prod_{j \in G_k} \left[1 - \pi_{jk} \prod_{i=1}^{M} \left(1 - p_{ij}\right)^{x_{ij}} \right] \tag{1}$$

Here, the set of targets that wants to destroy the asset k is represented with G_k and x_{ij} is a binary value which represents the assignment of weapon i to the target j. In our problem, we also consider the weapon costs and the value of the threat. Now, we expand the objective function given in (1) as follows.

$$\sum_{k=1}^{K}\omega_k\prod_{j\in G_k}\left[1-\pi_{jk}.\Omega_j\prod_{i=1}^{M}\left(1-p_{ij}\right)^{x_{ij}}-\log\varphi_i\right] \tag{2}$$

Now, in this new utility function Ω_j represents the value of the target j and φ_i is the cost of weapon i. Since WTA model aims to assign the most suitable weapon to the corresponding target, the discussed utility function utilizes the value of the target. In other words, while the utility function is increasing the probability of the survival of the assets, simultaneously the model considers the proper weapon assignment to the threats in terms of their costs.
As we mentioned our aim is to design an aerial defense system for a naval fleet. It is straightforward that the vessels on the fleet have various types of weapons and also there are various kinds of threats.

FIGURE 1 It shows a simplified naval environment having three assets and three incoming targets. The number of assets and targets may vary according to the scenario of the problem.

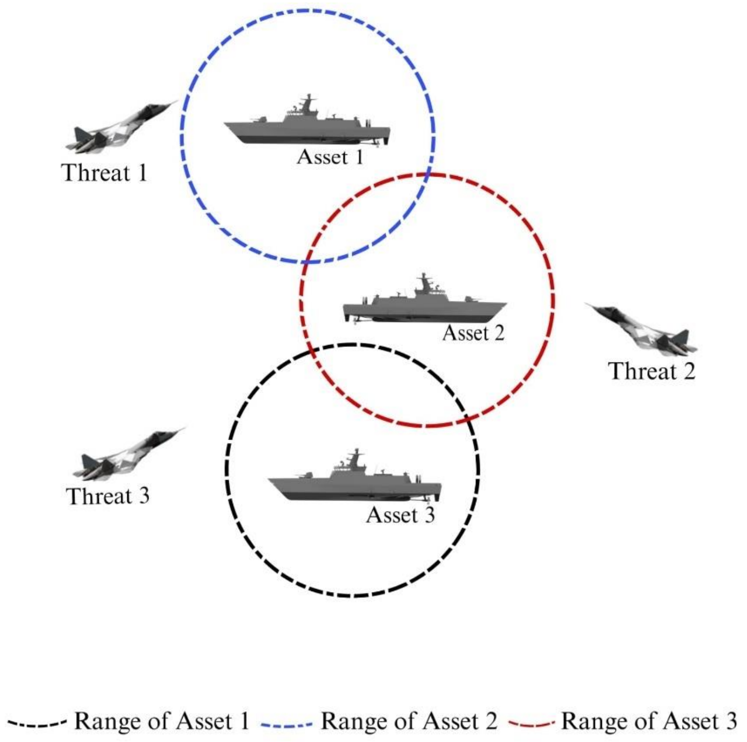


FIGURE 1 A simplified naval environment.

In Figure 1, circles define the range of weapons on each vessel. Our aim is to design an algorithm to make the choice of optimum weapon from the vessels' inventory while maximizing the survivability of assets and simultaneously minimizing the weapon costs.

3. Game Theoretical Solution

WTA problem can be considered as a M scalar optimization problem. There are solutions for this problem in the literature [1, 2, 6, 7, 11, 12]. In this study, we consider the problem in a game-theoretical approach where vessels are players and try to optimize their individual utilities. Using the game theory makes the algorithm more efficient. Game theoretical solutions will reduce communication and computational burdens, as each individual can make decisions with limited or no information from other individuals [3].

In a game-theoretical approach, each vessel is a player and tries to maximize its expected utility function. However, in an area defense scenario with more than one vessel, defending units are also expected to contribute to the global utility. For this reason, it is necessary to define a game where the vessels act in cooperation. Thus, any weapon of each vessel would be determined according to the common benefit or utility of the whole system.

Optimal assignment depends on the defensive and offensive characteristics of vessels and targets, respectively as well as the number of vessels and targets. It is straightforward that each vessel intends to optimize its self-utility function. However, alignment of vessel utilities should be taken into account to reach a maximal global utility [3]. On the other hand, alignment of individual utilities with a global utility is discussed in ordinal potential games.

Expected utility functions of the vessels are calculated according to the von Neumann-Morgenstern utility approach [13]. To align the utility functions of each vessel with the global utility function, we follow the definitions given in [14].

V_i and a_i represent the *i*-th vessel and the assignment of the corresponding vessel. The weapon-target assignment depends on whether the *j*-th target T_j is within the range of the i-th vessel. The set of vessels and assignments are shown as $V \coloneqq \{V_1, V_2, \dots, V_N\}$ and $a \coloneqq \{a_1, a_2, \dots, a_N\}$, respectively. Thus, each individual vessel V_i selects a proper assignment a_i to have a maximum utility function value $U_{V_i}(a)$. Players of the game decide to maximize their utility functions while providing a high global utility function, $U_g(a)$ [3].

In the range restricted utility function approach, the utility of a vessel and the global utility, $U_g(a)$ can be given as by (3).

$$U_{V_i}(a) = \sum_{T_j \in \mathcal{A}_i} U_{T_j}(a) \tag{3}$$

The basic problem with range restriction is that the restricted areas may overlap. For the overlapping regions another alignment function needed to be used in conjunction with the range restricted utility function.

The second utility function that has been defined in [3] is equally shared utility function. According to equally shared utility function, given in (4), the vessels share the utility value if they lock on the same target.

$$U_{V_i}(a) = \frac{U_{T_j}(a)}{n_{T_j}(a)}, \text{ if } a_i = \mathcal{T}_j \tag{4}$$

Equally shared utility function may not yield to optimum solution especially if one of the targets has much higher value than the others. The third utility function defined in [3] is wonderful life utility function. According to wonderful life utility function, given in (5), the vessels get as much utility as they contribute to the global utility.

$$U_{V_i}(a_i, a_{-i}) = U_{\mathcal{T}_j}(a_i, a_{-i}) - U_{\mathcal{T}_j}(\mathcal{T}_0, a_{-i}), \text{ if } a_i = \mathcal{T}_j \tag{5}$$

The last utility function to discuss is identical interest utility function. According to identical interest utility function all vessels' individual utilities equal to the global utility. In [3], authors state that identical interest utility function yields to optimum solution. On the other hand, with identical interest utility function, as shown in (6), every vessel needs to know each other's utility and the global utility. Thus, it increases communication and computational burden.

$$U_{V_i}(a) = U_g(a), \ \forall V_i \in V \tag{6}$$

In the proposed solution, if there are more than one threat for the assets, a two-step solution is considered. In the first step, the vessel will decide on which target they should be assigned and after that they decide the weapon of the corresponding vessel. If there are more than one vessel assigned to a single threat, they will play the game of the single threat case until the game reaches to the Nash equilibrium. The Nash equilibrium is the point of no regret, which is the optimum solution for the game.

To be able to have an improved computational burden, we limited the range. The range has been designed dynamically rather than a static one. On this matter, the "dualist game" has been the base of the algorithm. The dualist game is based on the dualist scenes in western movies. Two gunman approaches each other, and they try to shoot each other. The problem here is that to decide when they should fire. The solution to that game is that a gunman should fire his weapon if on the next step the other gunman's probability of shooting him is higher than his probability of shooting the other gunman on the current step.

If we apply the dualist game to our application, the vessels shall fire their weapons if there is a certain kind of threat level to the asset. The main reason for that is, when the target gets closer to the weapon, the uncertainty level begins to drop, and the probability of kill for the weapon gets higher. By limiting the range, the weapon systems do not need to check for assignment for every target, instead, they need to check for an assignment just for the targets which are in the range of them. In other words, the following equation 7 can be used to limit the range. For interested readers, computational burden of game theoretical algorithms is examined in [15]. Thus, when the situation in equation 7 happens, the vessel shall fire its weapon.

$$\prod_{k=1}^{K} \pi_{jk} > \prod_{i=1}^{M} p_{ij} \tag{7}$$

In addition to deciding the timing of the firing units, the equation 7 also provides a limit for the range. In [6], Karasakal shows a decision methodology for determining the vessel that fires first by estimating the trajectory of the target. Unlike that methodology, this study makes the decision of the vessels which are able to fire by employing adaptive limitation of the range.

By using equation 1 and 2 it is possible to calculate the utility for a single vessel. On the other hand, it is important to note that, every vessel will try to maximize its own utility and the result may not be aligned with the global utility. It is also possible that the outcome may not be optimum. To align the utilities of the vessels one shall use the potential functions described by equations 8, 9, and 10 as in [3].

In the following equations, V is the space that defines vessels, a is the assignment profile, $U_{V_i}(a)$ is the utility for vessel number i, $U_g(a)$ is the global utility, and a_{-i} is the assignment profile for all vessels except vessel number i. Then,

$$U_{V_i}(a_i', a_{-i}) - U_{V_i}(a_i'', a_{-i}) > 0 \leftrightarrow U_g(a_i', a_{-i}) - U_g(a_i'', a_{-i}) > 0 \tag{8}$$

If equation 8 is satisfied, then the utility for vessel is aligned with the global utility. Arslan et. al. defined the optimum assignment condition in [3] as,

$$a_{opt} \in argmax\ U_g(a), a \in A \tag{9}$$

To be able to align the global utility and individuals' utility, we employ potential games, also mentioned in [3] and [14]. Potential games are based on existence of a potential function. When a player changes its strategy, if the difference in the function is equal to the difference in the expected utility, then this function is a potential function for this game and the game is called as a potential game. [3] and [14] describe this type of game as,

$$U_{V_i}(a_i, a_{-i}) - U_{V_i}(a_i'', a_{-i}) = \Theta(a_i', a_{-i}) - \Theta(a_i'', a_{-i}) \qquad (10)$$

The function $\Theta(\mathrm{a})\colon \mathrm{A} \rightarrow \mathrm{R}$ is called as the potential function. There are two different types of potential functions, which are ordinal and cardinal potential functions. Ordinal potential functions are defined in [3] and [14] as,

$$U_{V_i}(a_i', a_{-i}) - U_{V_i}(a_i'', a_i) > 0 \leftrightarrow \Theta(a_i', a_{-i}) - \Theta(a_i'', a_{-i}) > 0 \qquad (11)$$

Here, the function $\Theta(\mathrm{a})\colon \mathrm{A} \rightarrow \mathrm{R}$ is called as the ordinal potential function.

This study takes advantage of the potential functions discussed in [3] when expected utility of the vessels and the global utility need to be increased at the same level. The first phase of the games that we have designed, as mentioned before, if there are multiple targets, then the vessels negotiate among each other by playing a game between them (with each other) to decide which vessel will be assigned to which target. In addition to that, if there are only one target and multiple vessels, the optimum vessel and the optimum weapon system to counter the attack will be decided. More on that, if there are multiple targets and only one vessel, the optimum weapon assignment to each target will be shown. If there is only one target and only one vessel, which is eligible to take the shot, then the optimum weapon system shall counter the attack. Therefore, the game is designed to find the optimum global utility, not the optimum utility for the vessels.

4. Simulation Parameters

Simulations parameters are discussed within this section. The simulation has run on a MATLAB environment. An example representation of the studied different simulation cases is given in
Figure *2*.

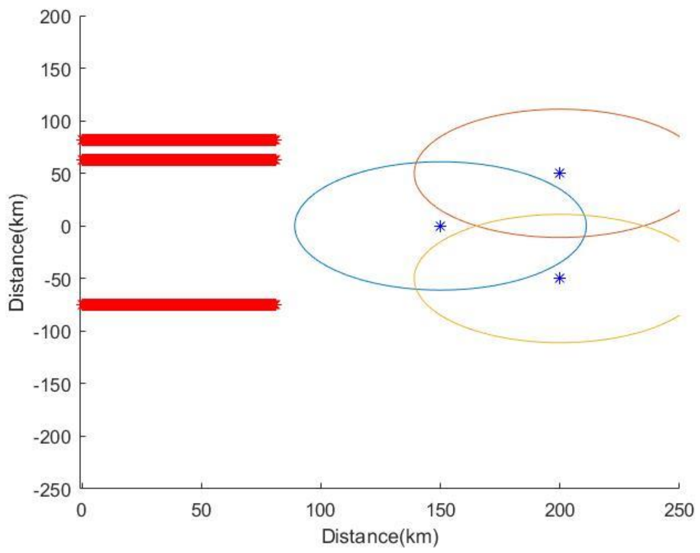


FIGURE 2 Simulation interface.

Here, the red lines represent the trajectory of the threats. Blue dots represent the vessels, the circles represent the range of the weapon systems. The threats have been distributed over the map randomly, and then they start to approach through the asset. The weapon systems counter the threats when they enter the area that can be seen on the interface.

In the simulation the vessels have multiple weapon systems. At the same time, there are many threats having different characteristics and weapon systems having different capabilities. There are multiple vessels to protect the assets.

The targets are assumed to be approaching from different directions, outside of the vessels range. Once they arrived at the range then the vessels shall react. One of the goals of the model is to cover all the targets. There shall not be unassigned target if there are enough weapons. Therefore, we used the alignment functions that we covered in previous section. Without the alignment functions, there is a possibility for the vessels to lock on the same target and there would be some targets that are unassigned to any weapons. This situation especially exists when utility of one of the targets is far larger than that of others. An alignment function aligns the individual utilities to the global utility. Therefore, the vessels have to try to increase the global utility. Once all the targets are covered, if there are still some weapons left, then they can lock on the targets that previously assigned to the other weapons.

Here, we assume that there are three weapons, three vessels, and three threats. Table 1 shows the probabilities that weapon *i* kills the threat *j*.

TABLE 1 Probability that weapon *i* kills the threat *j*, p_{ij}.

	All Vessels		
	Weapon 1	Weapon 2	Weapon 3
Threat 1	0.6	0.3	0.0001
Threat 2	0.8	0.5	0.3
Threat 3	0.9	0.6	0.5

The values in the simulation have been chosen to cover a wide range of weapons and threats. Threat #1 is clearly the strongest threat and hardest to kill. Table 1 also shows that weapon #1 is the strongest weapon and has the highest probability to kill. Weapon #3 and threat #3 added as weak weapon and threat. Weapon #2 and threat #2 are the mediocre ones. By this way, in the simulation in can be examined how the system reacts to weak and strong threats by having different weapons with different capabilities.

It is also mentioned that there are multiple objectives for the algorithm and one of them is to minimize the ammunition budget. This study assumes that the ammunitions for different weapon systems are not the same. Table 2 shows the ammunition budget of weapon *i*. It is assumed that the strongest weapon should have the highest price and the weakest should be the least expensive one.

TABLE 2 Ammunition budget of weapon i, φ_i.

Weapon 1	Weapon 2	Weapon 3
0.95 units	0.6 units	0.2 units

One of the parameters that we take into account is the probability that threat *j* destroys the asset k. Table 3 shows these probability values. It is assumed that the strongest threat should have the highest probability to destroy the asset and the weakest has the lowest probability.

TABLE 3 Probability that threat *j* destroys the asset k, π_{jk}.

Threat 1	Threat 2	Threat 3
1	0.5	0.3

Another parameter that we need to define the utility function is asset's value constant. The assets may have different importance level in a real combat zone. Therefore, we chose a most valuable asset, a least valuable asset and a mediocre one. Table 4 shows the asset values.

TABLE 4 Asset values, ω_k.

Asset 1	Asset 2	Asset 3
1	0.5	0.3

The last parameter that we need to define is a value constant for the threats. It is assumed that the strongest threat should be the most valuable one and, the weakest threat should be the least valuable one. The constant is given in Table 5.

TABLE 5 Threat values, Ω_j.

Threat 1	Threat 2	Threat 3
1	0.5	0.3

Using these values in equation 2, we are able to match these situations into numeric values. It also fits The Neumann-Morgenstern Theorem and its axioms.
A pseudo code for the simulation algorithm is given with Algorithm 1.

4. RESULTS

The results of our simulations using the weapon and threat types which have been mentioned in the previous section are given within this section. For the first trial we simulated a single threat situation. When the threat #1 gets into the range of a vessel, the utility function produces values for three weapons on it. For this trial, a single threat is assumed to be in the range. This scenario is based on the method of static games. Hence, the time is assumed to be frozen for a moment. The weapon system should fire the weapon, which has the highest utility value. According to the results given in Table 6, weapon #1 has the highest utility value, which is shown bold.

ALGORITHM 1 Pseudo code for the simulation algorithm.

```
1:  for (Each asset)
2:      Define range;
3:  end
4:  if (Any threat in any range of any asset)
5:      Check if it is in the range of single or multiple assets
6:      if (Single asset against single threat)
7:          Get utility values for the weapons;
8:          Assign the weapon with the highest utility;
9:      else if (Multiple assets against single threat OR multiple assets multiple
    threats)
10:         for(Each asset has a shared threat in their shared range)
11:             Get utility values for the weapons;
12:             Assign the weapon with the highest utility;
13:         end
14:         Prepare the game matrix;
15:         Apply the alignment function;
16:         Find the Nash Equilibrium;
17:         Assign the weapon;
18:     else if (Single asset against multiple threats)
19:         Get utility values for the weapons;
20:         Assign the weapon with the highest utility;
21:     end
22: end
```

TABLE 6 Simulation results of a single threat scenario for threat #1.

Weapon	Weapon #1	Weapon #2	Weapon #3
Maximum Utility Value	**0.5050**	0.2400	0.0000

We considered the simulation results for a single threat case for the threat #2 and #3 and the results are given in Table 7 and Table 8, respectively.

TABLE 7 Simulation results of a single threat scenario for threat #2.

Weapon	Weapon #1	Weapon #2	Weapon #3
Maximum Utility Value	**0.3050**	0.1900	0.1300

TABLE 8 Simulation results of a single threat scenario, for threat #3.

Weapon	Weapon #1	Weapon #2	Weapon #3
Maximum Utility Value	0.1600	0.1200	**0.1750**

For the scenarios above, the range is limited as well. The range limitations are made by using dualist game model. Equation 7 shows how range limitation is done by using the dualist game. For this situation, the weapon will be fired when the probability to kill for the threat of the asset is greater than the probability to kill for the weapon of the threat. For this reason, the cumulative distribution functions of the weapon and of the threat have been used.

For the scenarios that have multiple threats and multiple vessels to fire the target, it has been mentioned that an alignment to global utility is needed. In [3], range restricted utility function is shown as one of the candidates as a potential function. However, when the ranges of the vessels overlap, the RRU function does not yield the optimum results. Table 9 shows such scenario with threat #1 and threat #3.

TABLE 9 Simulation results of two threats and two vessels having overlapping ranges scenario, RRU function with no alignment function.

	Vessel #2		
Vessel #1		Threat #1	Threat #3
	Threat #1	**0.5050, 0.5050**	0.5050, 0.0000
	Threat #3	0.0000, 0.5050	0.0000, 0.0000

As one can see from Table 9, threat #1 strategy is strictly dominant strategy for vessel #1 (0.5050 >0, 0.5050>0). Thus, vessel #2, with the knowledge that vessel #1 is choosing the threat #1, must choose threat #1 as well (0.5050>0). Therefore, the Nash equilibrium is at [Threat #1, Threat #1]. As equation 3 shows that the global utility is some of the vessels' individual utilities. There is only one vessel can hit the target and therefore, the maximum global utility can be 0.5050 with this result. On the other hand, the maximum global utility could be sum of each target's utility. Thus, the maximum global utility should be 1.01 and this result shows that, without an additional alignment, when the ranges of two vessels are overlapped, there is not any alignment function anymore and as a result the global utility is not the optimum one.

For the overlapping area case, another alignment is needed. The equally shared utility function in [3] has been employed along with the range restricted utility function to overcome this problem. This function makes an equal distribution of utility if two or more vessels have the same target. In [3] it is mentioned that equally shared utility function may not be optimum for some situations for the alignment. Our simulations show that especially if one of the targets' utility is high enough, even though the utilities are shared, all of the weapons still try to lock on the high value target which yields a sub-optimum results. To show why equally shared utility function failed, we can show an example trial with two threats and two vessels with overlapping range areas. When the targets were on the overlapping area both weapon systems choose their optimum weapons to counter the threat, which were the ones that maximize their utility function. Here, the method is the same with one target and one vessel case. In other words, equation 3 calculates the utilities of the weapons and assign them to the targets. For this example, the difference is that there are multiple weapon systems and multiple targets, and the system needs to decide which weapon system will counter to the incoming targets. The game theory based solution gets the best weapon values for both weapon systems to counter the threats as inputs. The inputs produce a game matrix.

TABLE 10 Simulation results of two threats and two vessels having overlapping ranges scenario, ESU function with RRU function.

	Vessel #2		
Vessel #1		Threat #1	Threat #3
	Threat #1	**0.2525, 0.2525**	0.5050, 0.1750
	Threat #3	0.1750, 0.5050	0.0875, 0.0875

As one can see from Table 10, selecting threat #1 is the strictly dominant strategy for vessel #1 (0.2525>0.1750 and 0.5050>0.0875). From this, we know that vessel #1 cannot make a profitable deviation from threat #1 strategy. Thus, we know that vessel #1 must choose threat #1. So, for the situation that vessel #1 choosing threat #1 strategy for vessel #2 again threat #1 strategy is the dominant one (0.2525>0.1750). Therefore, the Nash equilibrium is at [Threat #1, Threat #1] point. For some situations like in Table 10, the most lethal threat has a very high utility value, the vessels are inclined to lock it even though they share the utility; it is still the highest utility for them individually. We can observe from Table 10 that with using the equally shared utility function, the system's global utility may not be at its optimum. Thus, we employ wonderful life utility function that has been mentioned in [3], instead of equally shared utility function. According to wonderful life utility function, a vessel only gets utility when it contributes to the global utility. This function along with range restricted utility function has increased the global utility and solved the problems that have been encountered during the scenarios with equally shared utility function. Equation 5 explains the use of wonderful life utility function with range restricted utility function. With this potential function the vessels do not only lock to the most important target, but some of them also choose different targets as well.

To show how wonderful life utility function changes the results, we observed the same scenario, discussed in Table 10, with wonderful life utility function instead of employing equally shared utility function. Simulation results are given in Table 11.

TABLE 11 Simulation results of two threats and two vessels having overlapping ranges scenario, WLU function with RRU function.

	Vessel #2		
Vessel #1		Threat #1	Threat #3
	Threat #1	0.5050, 0.0000	**0.5050, 0.1750**
	Threat #3	0.1750, 0.5050	0.0000, 0.1750

As one can see from Table 11, again selecting threat #1 is a strictly dominant strategy for vessel #1 (0.5050>0.1750 and 0.5050>0). Thus, vessel #2 must always choose threat #3. Therefore, the Nash equilibrium is at [Threat #1, Threat #3] point. This result shows that with wonderful life utility function we reached an alignment for the individual utilities to the global utility.

The last utility design method along with range restricted utility function that we will show in this study is identical interest utility function. This utility design method makes each vessel's utility equal to the global utility. We made experiment with this utility design method as well. We were able to reach the optimum Nash Equilibria by using this method. In fact, as mentioned in [3], the result must yield to the highest global utility, because the Nash equilibrium for the vessels is exactly that point. If we continue with the same example, Table 12 shows the simulation results when we employ identical interest utility function.

TABLE 12 Simulation results of two threats and two vessels having overlapping ranges scenario, IIU function with RRU.

	Vessel #2		
Vessel #1		Threat #1	Threat #3
	Threat #1	0.5050, 0.5050	**0.6800, 0.6800**
	Threat #3	**0.6800, 0.6800**	0.1750, 0.1750

As one can notice from Table 12, there are two Nash equilibria for this example. [Threat #1, Threat #3] and [Threat #3, Threat #1] are the Nash equilibria. As one can see from Table 11, a vessel cannot make a profitable deviation from these points. They are already most profitable strategies for both of them, and both of them makes the vessels aligned with the global utility.

TABLE 13 Results for Different Scenarios.

#	**Scenario**	**Maximum Global Utility**
1	3 threat #1 against 3 identical vessels RRU Function	1.5150
2	2 threat #1 and 1 threat #3 against 3 Identical vessels RRU Function	1.0100
3	3 threat #1 against 3 identical vessels RRU Function with ESU Function	1.5150
4	2 threat #1 and 1 Threat #3 against 3 identical vessels RRU Function with ESU Function	1.0100
5	2 Threat #1 and 1 Threat #3 against 3 identical vessels RRU function with WLU function	**1.185**
6	2 Threat #1 and 1 Threat #3 against 3 identical vessels RRU function with IIU function	**1.185**
7	1 Threat #1 and 2 Threat #3 against 3 identical vessels RRU function and ESU function	0.5050
8	1 Threat #1 and 2 Threat #3 against 3 identical vessels RRU function with WLU function	**0.8550**
9	1 Threat #1 and 2 Threat #3 against 3 identical vessels RRU function with IIU function	**0.8550**
10	3 Threat #1 against 3 identical vessels RRU function with WLU function	1.5150
11	3 Threat #1 against 3 identical vessels RRU function with IIU function	1.5150
12	3 Threat #1 against 5 identical vessels RRU Function with WLU Function	1.5150
13	3 Threat #1 against 5 identical vessels RRU function with IIU function	1.5150
14	3 Threat #1 against 2 identical vessels RRU function with WLU function	**1.0100**
15	3 Threat #1 against a single vessel RRU function with WLU function	0.5050

According to our results, wonderful life utility function is the optimum utility function for our application, which yields to optimum solution.

On the other hand, the identical interest utility function will ultimately be ineffective, as also mentioned in [3]. Every vessel in the game must know exactly what the global utility is. This adds another communication burden to the system. We also present several simulation results for various scenarios in Table 13.

The first scenario shows that if all of the threats are replicas of each other, then the range restricted utility is enough for a proper alignment even though if there are some overlapping areas. However, the effect of employing other utility functions appears when there exists threats having different properties. Scenarios 2, 4, 5 and 6 show that how the maximum global utility changes when RRU is employed alone, ESU with RRU, WLU with RRU, and IIU with RRU. These trials show that utilization of RRU alone and ESU with RRU have some drawbacks and do not approach to the optimum solution. On the other hand, utilization of WLU with RRU and IIU with RRU have yielded the optimum solution.

Scenarios 7, 8, and 9 show the results of ESU, WLU, and IIU with RRU for a different scenario. Again, this one show that ESU has a limited capacity; on the other hand, WLU and IIU led to an optimum solution.

For the scenarios 10 and 12 as well as 11 and 13 we used same utility functions, but we increased the number of vessels. The number of threats remain the same. Note that the utilities for 10 and 12, and 11 and 13 are the same, because even though the defenders are increased, all the threats are already covered with other weapons, so, they have no contribution to the global utility.

For the scenarios 14 and 15, the number of threats is higher than the number of defenders. For these scenarios even though there are threats that remain uncovered, the global utility is limited with the number of vessels and their contribution to the global utility. Therefore, when number of vessels decreases, the global utility decrease as well.

5. Conclusions

Simulation results show that the system will be stable at Nash equilibria including a pure one as long as it uses a utility function which leads to a potential function. We used a dynamic range limitation. It should be actually a natural result of the system, because if the threats are out of defined range, the utility value is usually lower than not firing the ammunition. Defining it in advance gives an advantage about the computational burden. The threats outside of the range are not considered during weapon assignment process and this leads a faster system, which is essential for this type of systems.

We used other potential functions alongside the range limitation and we were able to see which one is the best choice. According to the results, if we use range restricted utility on its own and if two or more vessels have overlapping areas on the range that they cover, it is not possible to align their utility with the global utility. Range limitation was an important aspect for this scenario. Thus, we decided to use other utility functions with it. Equally shared utility function is failed for some situations, especially if one of the targets has a utility value that is too high. Wonderful life utility function and identical interest utility function have the ability to get the system to the optimum point; however, identical interest utility function has additional communication burden. For this reason, the cumulative distribution functions of the weapon and of the threat function to be used with the RRU function should be wonderful life utility function. As one can see from the results section wonderful life utility function is one of the alignment functions that reached the maximum utility value for the global utility. It is also a better choice than the identical interest utility function which is another alignment function that leads the same result with wonderful life utility function, because, with identical interest utility function every vessel must have the information of the each other's utility value and the global utility value. Thus, wonderful life utility function has less computational burden. For these reasons, we point that wonderful life utility function along with range restricted utility function is the best choice according to our simulation results.

As a future work, the game theory based solution can be extended to a truly dynamic system, which is less discussed in the literature.

Author Contribution Statements The authors contributed equally to this article.

Declaration of Competing Interests The authors declare that they have no competing interest.

References

[1] Roux, J. and Vuuren, H. V., Threat evalution and weapon assignment decision Support: A review of the state of art, *Orion*, 23 (2) (2007), 151-187, https://doi.org/10.5784/23-2-54.

[2] Lötter, D. and Vuuren, J. V., Weapon assignment decision support in a surface based air defence environment, (2013), https://www.vuuren.co.za/papers/MORSPaper.pdf.

[3] Arslan, G., Marden, J. R. and Shamma, J. S., Autonomous vehicle-target assignment: A game-therotical formulation, *Trans. ASME,*129 (2007), 584-596.

[4] Kline, A., Ahner, D. and Hill, R., The weapon-target assignment problem, *Comput. Oper. Res.*, pp., 5 (2019), 226-236, https://doi.org/10.1016/j.cor.2018.10.015.

[5] Shi, Y., Xing, Y., Mou, C. and Kuang, Z., An optimization model based on game theory, *J. Multimed.*, 9 (4) (2014) 583-589, https://doi.org/10.4304/jmm.9.4.583-589.

[6] Karasakal, O., Air defense missile-target allocation models for a naval task group, *Comput. Oper. Res.*, 35 (2008), 1759-1770, https://doi.org/10.1016/j.cor.2006.09.011.

[7] Li, X., Zhou, D., Pan, Q., Tang Y. and Huang, J., Weapon-target asignment problem by multiobjective evolutionary algorithm based on decomposition, Complexity, 2018 (2018), 1-19, https://doi.org/10.1155/2018/8623051.

[8] Aplak, H. S. and Türkbey, O., Fuzzy logic based game theory applications in multi-criteria decision making process, *J. Intell. Fuzzy Syst.*, 25 (2) (2013), 359-371.

[9] Taghavi, R. and Ranjbar, M., Weapon scheduling in naval combat systems for maximization of defense capabilities, *Iran. J. Op. Res.*, 6 (2) (2015), 87-99.

[10] Șahin, M. A. and Leblebicioğlu, K., Approximating the optimal mapping for weapon target assignment by fuzzy reasoning, *Inf. Sci.*, 255 (2014), 30-44.

[11] Johansson, F. and Falkman, G., Real-time allocation of firing units to hostile targets, *J. Adv. Inf. Fusion*, 6 (2) (2011), 187-199.

[12] Paradis, S., Benaskeur, A. and Cutler, P., Threat evalution and weapon allocation in network-centric warefare, *Proceedings of the Seventh International Conferance on Information Fusion*, (2005).

[13] Ng, Y. K., Expected subjective utility: Is the neumann-morgenstern utility the same as neoclassical's?, *Soc. Choice Welf.*, 1 (1984), 177-186.

[14] Monderer, D. and Shapley, L. S., Potential games, *Games Econ. Behav.*, 14 (1996), 124-143, https://doi.org/10.1006/game.1996.0044.

[15] Roughtgarden, T., Algorithmic game theory, *Communications of the ACM*, 53 (7) (2010), 78-86.

[16] Yang, Y. and Wang, F. Y., Budget Constraints and Optimization in Sponsored Search Auctions, Academic Press, 2014.

[17] Wonnacott, W. M., *Modelling In The Design and Analysis of a Hit-To-Kill Rocket Guidance Kit, Thesis*, Monterey, California: Naval Postgraduate School, 1997.

[18] Washburn, A. R., Notes On Firing Theory, Naval Post Graduate School, Monterey, California, 2002.

[19] US. Defence Documentation Center, *Scientific and Technical Information*, Alexandria, Virginia, 1963.

[20] Glazebrook, K. and Washburn, A. R., Shoot-Look-Shoot: A review and extension, *Op. Res.*, 6 (2004), 454-463.

[21] Chatterjee, B., An optimization formulation to compute Nash equilibrium in finite games, 2009 Proceeding of International Conference on Methods and Models in Computer Science, (2009), https://doi.org/10.1109/ICM2CS.2009.5397970.

http://communications.science.ankara.edu.tr

Commun.Fac.Sci.Univ.Ank.Series A2-A3
Volume 66, Number 1, Pages 64-81 (2024)
DOI: 10.33769/aupse.1308165
ISSN 1303-6009 E-ISSN 2618-6462

COMMUNICATIONS SERIES A2-A3

Research Article; Received: June 14, 2023; Accepted: October 27, 2023

MODELLING INERTIAL MEASUREMENT UNIT ERROR PARAMETERS FOR AN UNMANNED AIR VEHICLE

Bağış ALTINÖZ[1,2], Hüsamettin EKEN[1], Anıl CÖNGER[1] and Sultan CAN[2]

[1]Roketsan, Ankara, TÜRKİYE
[2]Ankara University, Faculty of Engineering, Department of Electrical and Electronics Engineering, Ankara, TÜRKİYE

Abstract. This paper demonstrates a study that focuses on the modeling, design, and realization of an Inertial Measurement Unit (IMU) component for the use of Unmanned Aerial Vehicles (UAV). The experimental data is obtained by multiple flights conducted by the realized UAV (Teknofest–SEMRUK team UAV). The structure is remodeled for increasing the accuracy, and performance of the UAV after the conducted flights. Noise parameters are estimated throughout the Allan variance analysis. MEMS technology-based capacitive-type accelerometers and gyroscopes are preferred. This paper also discusses the error types and compares the real data with the modeled simulation data. Systematic errors of the inertial sensors are simulated according to their datasheet parameters. Sensor filters and noise are modeled and they are also implemented in the simulation. Simulation results and UAV measurements are compared to observe the efficiency of modeling. A complementary filter is presented and combined with a magnetometer, accelerometer, and gyroscope to obtain the ultimate design. The comparison showed a satisfactory agreement among the complementary filter measurements and UAV measurements in the stable position and the results presented.

1. Introduction

In the last decades, the increment of the demand for tools and equipment that do not involve humans in several application areas increased the studies about Unmanned Aerial Vehicles (UAVs). UAVs are briefly a kind of aircraft that does not have a pilot or passenger, carries equipment suitable for its intended use, such as a camera, GNSS, and laser scanning, and can perform its duty remotely and/or automatically.

Keywords. Complementary filter, inertial measurement unit, sensors, unmanned aerial vehicles.

bagis.altinoz@roketsan.com.tr; 0000-0003-2952-9437
husamettineee@gmail.com; 0009-0001-2693-4923
anilconger@gmail.com; 0009-0002-4922-5852
sultancan@ankara.edu.tr-Corresponding author; 0000-0002-9001-0506.

Within the need for automation, the oil and gas industry [1], inspection and control of wind turbines, agriculture, search and rescue, and firefighting applications [2], [3], aerial photography, cargo, and cartography [4], both civil and military [5] - [9] applications are requiring UAVs. Such systems attract a great deal of attention in terms of saving time and cost simultaneously with many tasks. Military, civil (hobby and commercial), and professional use of UAVs for scientific purposes are increasing rapidly in our country and all over the world [10]. Especially in military applications, mission completion in tactical critical military missions without endangering personnel is not only time and cost-saving but also the lifesaving advantage of the UAVs since UAVs find their place in many areas such as intelligence, reconnaissance, attack, early warning, air defense, cargo transportation.

UAV’s final design requires several items that should be evaluated individually. As one of the items on the flight control board, one of the most important components is a part of the UAV that can be defined as the brain of the overall system. The control board contains a gyroscope, accelerometer, and magnetometer sensors. The gyroscope sensor is used for measuring angular velocity, the magnetometer is used for direction finding, and the accelerometer sensor is used for acceleration measurement. According to the information received from these sensors, the card provides the desired movement and stabilization by changing the speed of the motors. A development board with an ARM-based processor is used as the control board in the UAV due to features such as power saving and high processing capability. Another crucial element is the engine. In the engine of UAV, brushless electric motors are generally used in RC vehicles because of the need for high KV (revolutions per voltage) and low friction. An electronic speed control card (ESC) is also needed to drive these brushless electric motors used. ESCs are the circuit that defines how fast the motors should spin. To control the UAV manually, at least a 4-channel control and a remote-control receiver that is compatible with this control are needed. Generally, the controls work at 2.4GHz frequency and their antenna lengths are short. For digital communication between the control and the receiver, it is first necessary to make an encrypted match between them. For the UAV to fly, sufficient thrust must be created in the engines. This force is provided by propellers. The propellers to be used in the UAV should be selected in the recommended diameter and structure according to the requirements of the engines. Li-Po batteries must be used to provide the necessary power to these electronic components used in the UAV. By calculating the total electrical power consumption of these electronic components, a battery with the appropriate capacity can be selected for the UAV. Another component is telemetry. With the telemetry module, the UAV can be monitored remotely, either wired or wirelessly. Finally, the GPS module used on the UAV provides the UAV's latitude, longitude, and altitude data according to sea level.

In this study, IMU design as the key component of the UAV is considered and modeled. After the realization of the UAV, the experimental data is obtained by proposed design flights and remodeled to increase the accuracy, and performance of the UAV. A complementary filter is presented and combined with a magnetometer, accelerometer, and gyroscope to obtain the ultimate design. MEMS technology-based capacitive-type accelerometers and gyroscopes are preferred.

2. Inertial Measurement Unit (IMU) in UAV Design

Inertial measurement units contain inertial sensors that are generally classified as accelerometers and gyroscopes. IMU integrates multi-axes, accelerometers, gyroscopes, and other sensors for estimating the location of the objects accurately.

Linear accelerations can be measured by accelerometers and angular velocity measurements are made by gyroscopes. Accelerometers are categorized into four different classes in technology viewpoint such as piezoelectric, piezoresistive, capacitive, and quartz accelerometers. Gyroscope technologies are categorized into three branches mechanical, optical, and capacitive. The output of the gyroscopes usually has data in terms of degrees/second. Inertial measurement units containing three accelerometers and three gyroscopes measured with six degrees of freedom. In inertial measurement units, in addition to accelerometers and gyroscopes, there are converters to meet the required power requirement, a processor with which calibration coefficients are loaded, and an interface that provides communication.

Although IMUs are commonly used in navigation the accumulated error is an important drawback that the designers have to cope with since any measurement errors, however small, are accumulated over time. Error is categorized into systematic errors and random errors.

2.1. **Systematic Errors**. Systematic errors can be defined as errors on inertial sensors they mathematically modeled, calibrated, and generally temperature dependent. Systematic errors occur when the behavior of a component of an inertial sensor changes with temperature. Therefore, the temperature is the primary concern for systematic errors. Decreasing the effect of systematic errors mostly depends on a precise calibration process. Thus, the calibration process must include temperature stabilization. Systematic errors are divided into three different categories: bias, scale factor, and misalignment.

2.2. **Bias**. Bias, which is also called offset or drift, is one of the systematic error components for inertial sensors. Sources of bias error can be from changes in vibration frequency to different movement electrons in sensor electronics. Bias causes drift in the inertial sensor signal even if a sensor is located at zero input position or rotation. The unit of bias error is deg/h for gyroscopes mg (milli-g) for accelerometers.

Bias errors in the accelerometer, and gyroscope cause position errors as explained below with Equations (1), (2), and (3):

$$p_a = \int_0^t \int_0^t b_a \, dt \tag{1}$$

$$p_a = \frac{1}{2} b_a t^2 \tag{2}$$

$$p_g = \int_0^t \frac{1}{2} g b_w t^2 \, dt = \frac{1}{6} g b_w t^3 \tag{3}$$

where b_a is standing for bias error in the accelerometer, b_g is for bias error in the gyroscope, g, and t are the gravity and time, respectively. p_a is the position error due to the accelerometer bias and p_g is the position error due to the gyroscope bias. An illustration regarding the bias error representation for inertial sensor data is presented in Figure 1.

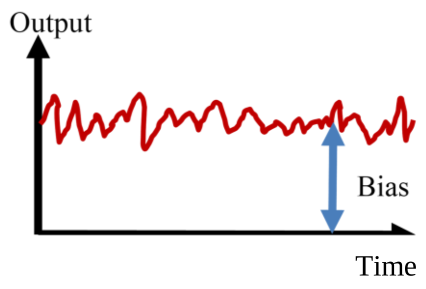


FIGURE 1. Bias Error representation for an inertial sensor data.

2.3. **Scale Factor.** Ideally, the proportion between input and output is one for the inertial sensors. This ratio is not achievable in most cases. Scale factor error is an essential component in the high acceleration or rotation environment. Because it affects the system in a higher magnitude than bias error. Figure 2 shows three different cases of scale factor error.

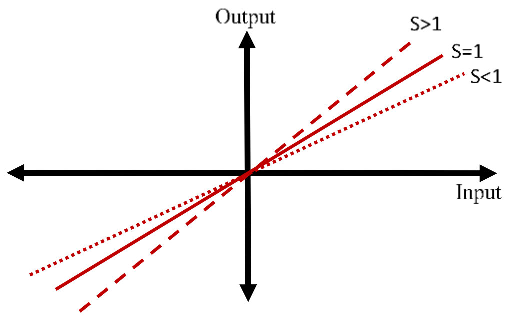


FIGURE 2. Scale Factor error cases for input and input of an inertial sensor.

2.3. **Misalignment.** Sensor alignments are orthogonal in ideal conditions as illustrated in Figure 3. However, integration imperfections, sensor production errors, and coupling between input and output axes are inevitable. Thus, the misalignment error arises.

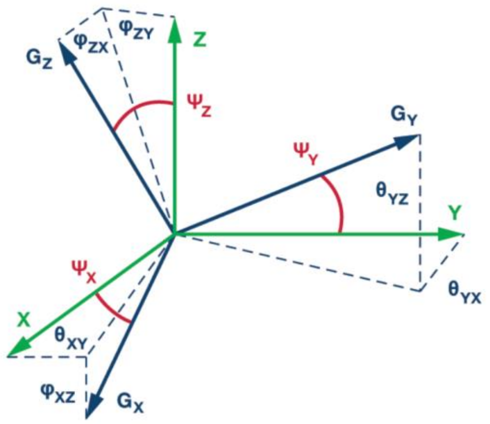


FIGURE 3. Misalignment error between three orthogonal axes.

2.5. **Random Errors**. Inertial sensor measurements include noise as with everything in nature. Random errors can be thought of as the components of noise part of the inertial sensor signals. Random errors differ from each other according to their frequency components. Velocity random walk and angular random walk are white noises for accelerometers and gyroscopes respectively. Additionally, bias instability, flicker noise, and rate random walk are also random errors.

Determining random errors is straightforward because there are two golden methods available in the literature. These methods are power spectral density and Allan variance. The former is generally used in signal and noise analysis, especially for white noise in signal processing applications. The latter is a special tool to estimate random errors on atomic clocks but the same noise parameters affect the inertial sensors. Thus, the same method is used to extract random errors.

In the following sections, angular/velocity random walk and bias instability (or flicker noise) are described.

2.6. **Angular/Velocity Random Walk.** The most dominant random error parameter is angular and velocity random walk in the inertial sensors. As described before, gyroscope measurement consists of angular random walk, and accelerometer measurements include velocity random walk error. Thermomechanical and electronic noises cause angular/velocity random walk errors. These random errors influence all of the frequency domains. Therefore, these errors are categorized as white noise.

Angular and velocity random walk parameters are estimated by Allan variance analysis. In this study, at least one-hour long data was collected from the inertial sensors to obtain Allan variance graphics. The unit of velocity random walk error is or $\frac{mg}{\sqrt{Hz}}$ and the unit of angular velocity walk is $\frac{\deg}{\sqrt{h}}$ or $\frac{\deg}{s\sqrt{Hz}}$.

2.7. **Bias Instability.** common random error parameter because it has low-frequency characteristics. It means that if the effect of bias instability is to be observed, a certain amount of time has to be elapsed after turning on the inertial sensors.

The unit of bias instability for the accelerometer is $\frac{meter}{\sec^2}$ or mg, and the unit of bias instability for the gyroscope is $\frac{\deg}{h}$ or $\frac{\deg}{s\sqrt{Hz}}$. Bias instability error is not modeled in this study because it is not an effective random error parameter for UAV applications.

2.8. **Proposed UAV Design and Realization.** The mathematical model of the proposed UAV is illustrated in Figure 4 in which there are 6 axes that the UAV can move along.

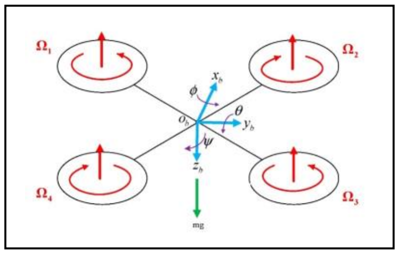


FIGURE 4. UAV configuration for the mathematical model.

$$\mathrm{U1} = \mathrm{b}({\Omega_1}^2 + {\Omega_2}^2 + {\Omega_3}^2 + {\Omega_4}^2) \tag{4}$$

$$\mathrm{U2} = b\sin\left(\frac{pi}{4}\right)({\Omega_1}^{\,2} - {\Omega_2}^{\,2} - {\Omega_3}^{\,2} + {\Omega_4}^{\,2}) \tag{5}$$

$$\mathrm{U3} = b\sin\left(\frac{pi}{4}\right)({\Omega_1}^{\,2} + {\Omega_2}^{\,2} - {\Omega_3}^{\,2} - {\Omega_4}^{\,2}) \tag{6}$$

$$\mathrm{U4} = \mathrm{d}({\Omega_1}^2 - {\Omega_2}^2 + {\Omega_3}^2 - {\Omega_4}^2) \tag{7}$$

The acceleration of yaw-, pitch, and roll angles by combining the engine's moment of inertia and rotor inertia with the equations given above. Obtaining these accelerations is given in the following equations.

$$\ddot{\phi} = \frac{\dot{\theta}\dot{\psi}\left(I_{yy} - I_{zz}\right) + J_r\dot{\theta}\Omega_r + l(U2)}{I_{xx}} \tag{8}$$

$$\ddot{\theta} = \frac{\dot{\phi}\dot{\psi}(I_{zz} - I_{xx}) - J_r\dot{\phi}\Omega_r + l(U3)}{I_{yy}} \tag{9}$$

$$\ddot{\Psi} = \frac{\dot{\theta}\phi(I_{xx} - I_{yy}) + (U4)}{I_{zz}} \tag{10}$$

Likewise, the equations required to obtain the accelerations in the x, y, and z axes are given below.

$$\ddot{X} = \frac{(\sin\Psi\sin\phi - \cos\Psi\sin\theta\cos\phi)U1 - A_x\dot{X}}{m} \tag{11}$$

$$\ddot{Y} = \frac{(\cos\psi\sin\phi + \sin\psi\sin\theta\cos\phi)U1 - A_y\dot{Y}}{m} \tag{12}$$

$$\ddot{Z} = \frac{mg - (\cos\theta\cos\phi)U1 - A_z\dot{Z}}{m} \tag{13}$$

Thus, with these equations given above, it will be possible to simulate a UAV that will fly with command control. The variables required for the equations are given in the table below

TABLE 1. Variable list for the thrust equations.

Symbol	**Description**	**Unit**
I_{xx}, I_{yy}, I_{zz}	Inertia moments	Kgm^2
J_r	Rotor inertia	Kgm^2
l	Rotor axis to quadcopter center distance	m
b	Thrust coefficient	$\mathrm{N/s}^2$
d	Drag coefficient	$\mathrm{Nm/s}^2$
m	Mass of UAV	Kg
A_x, A_y, A_z	Air resistance in each axis	Kg/s

It is also necessary to calculate the total speed of the rotors required for the equations. This total is formed by the difference in the rotation directions of the rotors. In quadcopters, since the first and third motors rotate in the same direction, the second and fourth motors rotate in the same direction, and the total speed is found as follows.

$$\Omega_r = \Omega_1 - \Omega_2 + \Omega_3 - \Omega_4 \tag{14}$$

Concerning those calculations and design parameters, the final design is conducted and it is presented in Figure 5. The battery connection, propeller connection, and top view of the design are shown in Figure 5(a), Figure 5(b), and Figure 5(c), respectively.

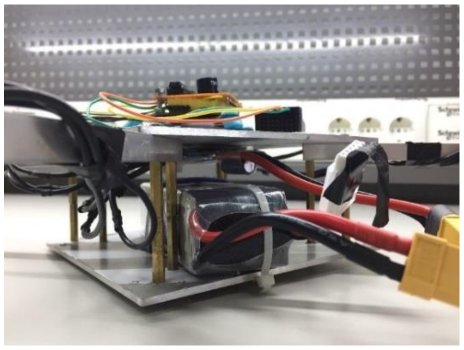
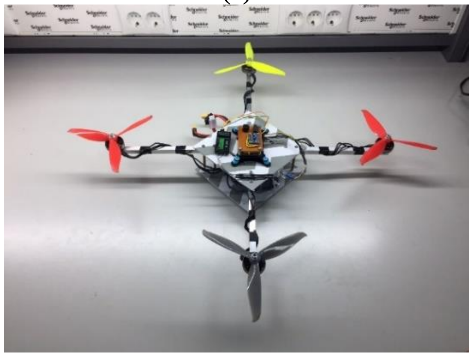
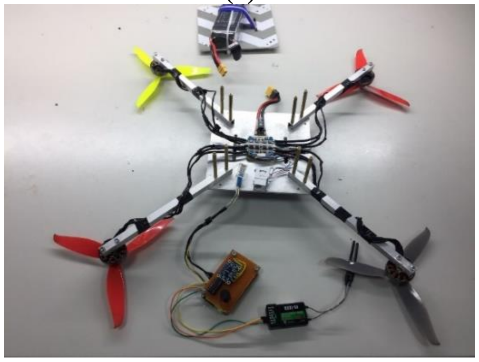

(a)

(b)

(c)

FIGURE 5. Design a) battery connection b) propeller connection c) top view of the final version.

3. Modelling an IMU

To calculate the random walk parameter, it is necessary to retrieve data from inertial sensors at constant temperatures for a long time. After the data are collected, the data is analyzed with the Allan variance method. The Allan variance method examines the data at short time intervals for the first step and then at longer time intervals as the analysis progresses.

The calculation of Allan variance is given below as the averaged data with length y_i:τ, N: data length, and the total number of data with M: τ length.

$$\sigma_{AV}^2 = \frac{1}{2(M-1)} \sum_{i=1}^{M-1} [y_{i+1} - y_i]^2 \qquad (15)$$
$$M = \frac{N}{\tau}$$

As a result of the Allan variance calculation, a log-log graph is drawn for different τ lengths as in Figure 6.

Since the most important error parameters in MEMS accelerometers and gyroscopes are the angle and velocity random walk parameters this study focuses on those parameters.

The Angle/Speed random walk parameter is calculated by the intersection of the Allan variance graph and the line with a slope of -1/2. Data were collected for 1 hour from the accelerometers and gyroscopes used within the scope of the project. The resulting Allan variance plots for the gyroscope and accelerometer are shown in Figure 7 and Figure 8, respectively.

Angle and velocity random walking parameters obtained as a result of the tests are given in the table below.

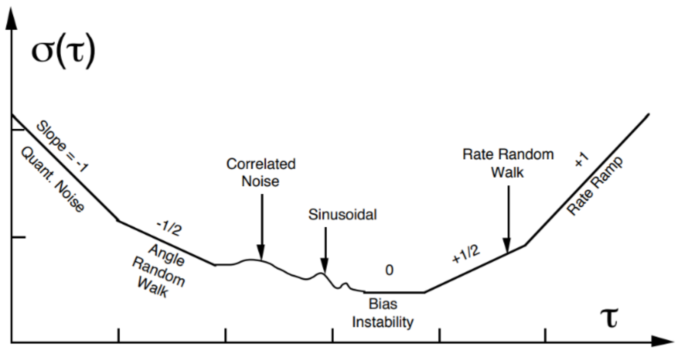


FIGURE 6. Representation of the Allan variance plot for different τ values 0.

TABLE 2. Angle and Speed Random Walk Parameters

Axes	**Gyroscope (*deg/√hr*)**	**Accelerometer (m/s^2/√Hz)**
X-Axis	1.9368	0.0012
Y-Axis	1.239	0.0012
Z-Axis	1.239	0.0014

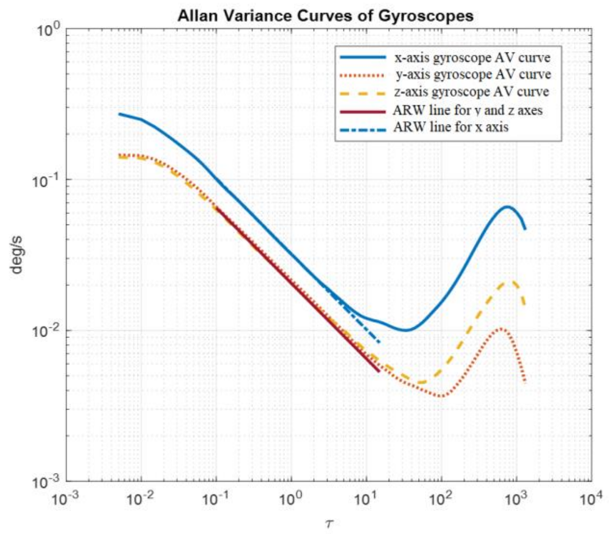


FIGURE 7. Gyroscope Allan variance plot.

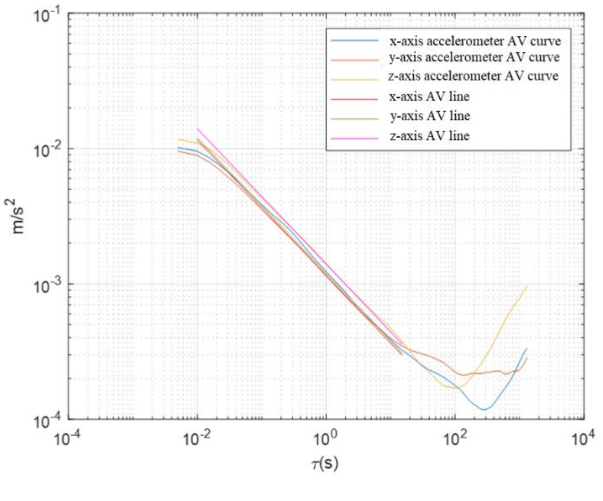


FIGURE 8. Accelerometer Allan variance plot.

While the sensors are being modeled, the filters must also be added to the models simultaneously. To reduce the noise power of the inertial measurement unit, a second-order Butterworth filter with a cutoff frequency of 30 Hz was used. While designing the sensor filter, the values shown in the data file of the sensors were adhered to. The following flow was applied while performing the modeling.

$$\mathrm{y[k]} = \frac{\mathrm{N}}{\sqrt{\Delta t}} w[k] \tag{16}$$

where N is the angle or noise random walk parameter, y[k] is angle-velocity random walk data, w[k] is white noise, Δt is sampling time. The noise values of the magnetometers used in the inertial measurement unit are taken as a reference. In the fixed position data, the y-axis magnetometer measured 200 microTesla more than the x-axis. While modeling, the relevant value was added directly as a total. A filter with a cutoff frequency of 30 Hz was applied to the magnetometer data.

$$x_m[\mathrm{k}] = \mathrm{A}w[k] \text{and } y_m[\mathrm{k}] = \mathrm{A}w[k] + 200uT \tag{17}$$

y_m[k] is standing for magnetometer data y-axis, x_m[k] is magnetometer data x-axis, w[k] is the white noise and A is the magnetometer noise parameter.

3.1. **Complementary Filter.** The complementary filter method is used widely in the literature, especially in inertial sensor fusion applications. This article uses a basic approach of the complementary filter to obtain the roll, pitch, and yaw angles. Accelerometer and gyroscope measurements are used in roll and pitch measurements. Gyroscope and magnetometer measurements are used in yaw measurements. The weight of the gyroscope measurement to calculate ϕ, θ, and ψ angles is 0.97. Additionally, the weight of accelerometer calculations to ϕ, θ angles is 0.03. Besides, the weight of magnetometer measurements to calculate ψ angle is 0.03. The mathematical equation of the basic complementary filter is given below and the block diagram of the sensor output is shown in Figure 9.

$$\mathrm{o[k]} = \alpha(\mathrm{a[k]}) + (1 - \alpha)(\mathrm{b[k]}) \tag{18}$$

o[k] shows the output of the complementary filter. α is the filter coefficient for sensor a. 1- α is also a coefficient for sensor b. Thus, a basic type of complementary filter is formed.

Comparing the mathematical model output and UAV measurements in the stable position (in zero rotation rate, gravity only affects the z-axis accelerometer) shows that these two measurements fit each other as illustrated from Figure 10 to Figure 15. Figures 10, 11, and 12 illustrate the comparison of UAV measurement and simulation results of the gyroscope for the *x*-, *y*-, and *z–axis*, respectively. The corresponding comparisons of the accelerometer are shown in Figures 13,14 and 15 for the *x*-, *y*-, and *z* –axis, respectively.

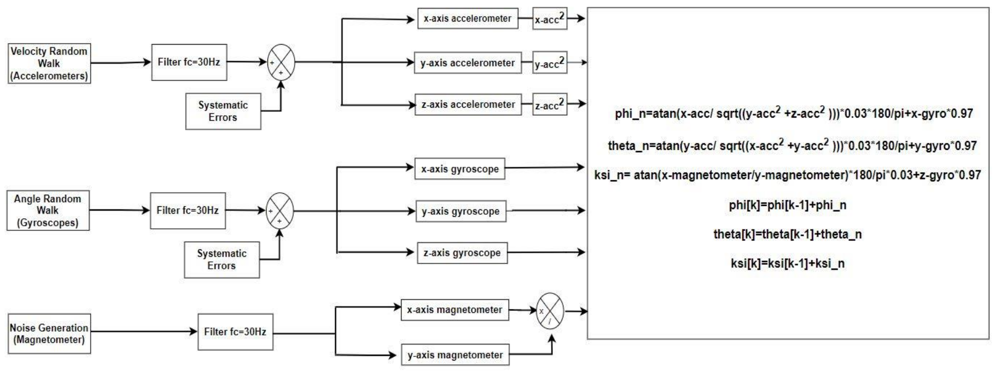

Velocity Random Walk (Accelerometers)
Filter fc=30Hz
Systematic Errors
x-axis accelerometer
y-axis accelerometer
z-axis accelerometer
x-acc²
y-acc²
z-acc²
Angle Random Walk (Gyroscopes)
Filter fc=30Hz
Systematic Errors
x-axis gyroscope
y-axis gyroscope
z-axis gyroscope
Noise Generation (Magnetometer)
Filter fc=30Hz
x-axis magnetometer
y-axis magnetometer
phi_n=atan(x-acc/ sqrt((y-acc² +z-acc²)))*0.03*180/pi+x-gyro*0.97
theta_n=atan(y-acc/ sqrt((x-acc² +y-acc²)))*0.03*180/pi+y-gyro*0.97
ksi_n= atan(x-magnetometer/y-magnetometer)*180/pi*0.03+z-gyro*0.97
phi[k]=phi[k-1]+phi_n
theta[k]=theta[k-1]+theta_n
ksi[k]=ksi[k-1]+ksi_n

FIGURE 9. Simulink Block for the ultimate design.

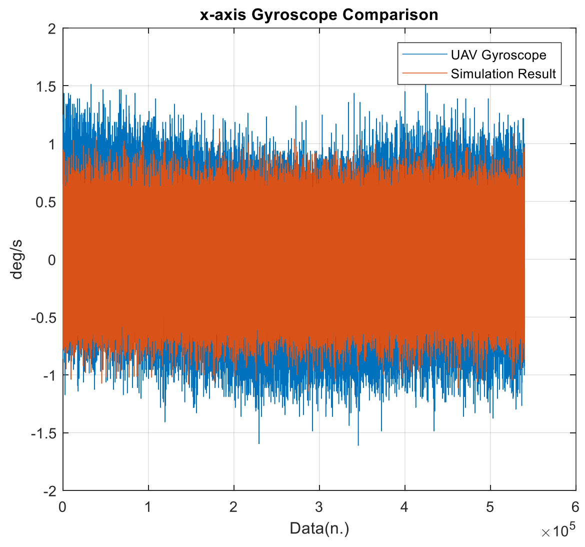


FIGURE 10. Comparison of UAV measurement and simulation result for x-axis gyroscope.

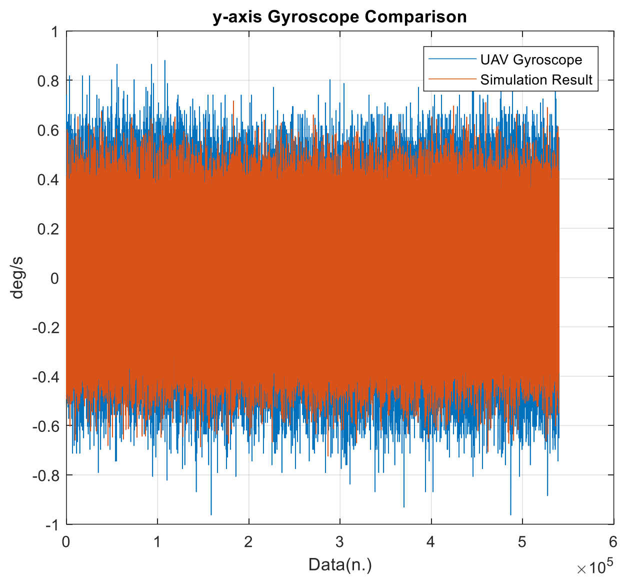


FIGURE 11. Comparison of UAV measurement and simulation result for y-axis gyroscope.

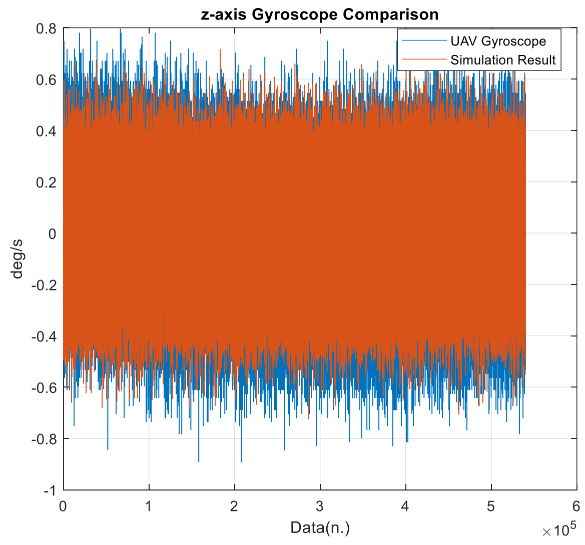


FIGURE 12. Comparison of UAV measurement and simulation result for z-axis gyroscope.

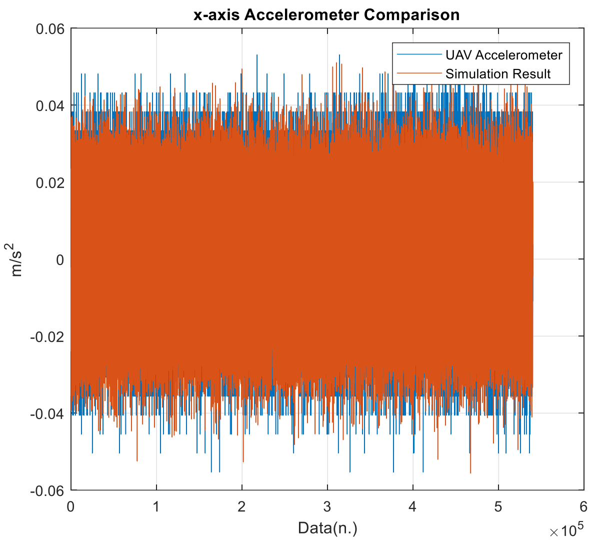


FIGURE 13. Comparison of UAV measurement and simulation result for x-axis accelerometer.

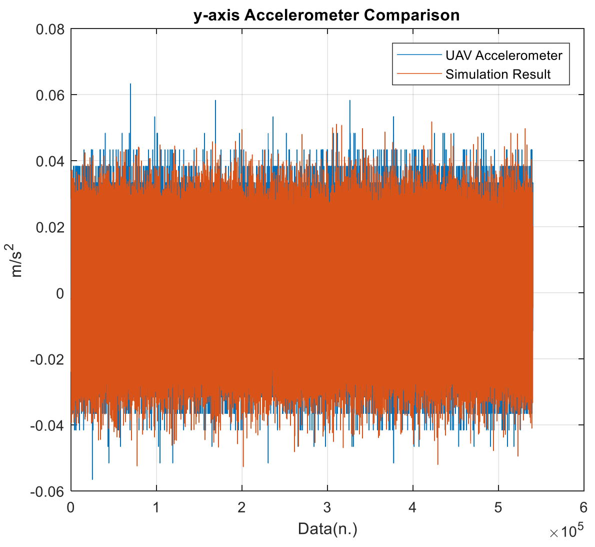


FIGURE 14. Comparison of UAV measurement and simulation result for y-axis accelerometer.

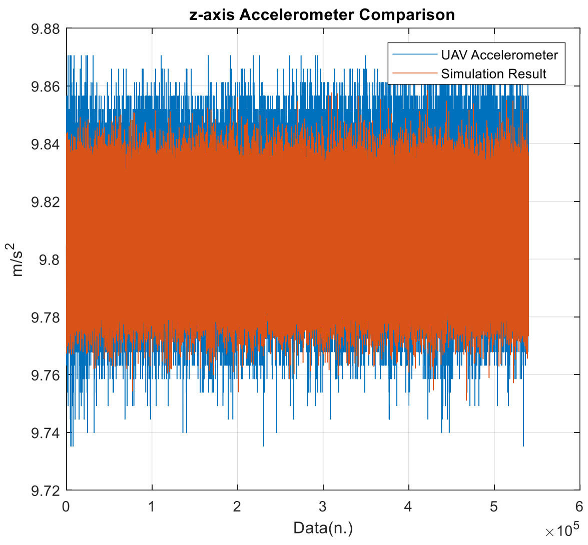


FIGURE 15. Comparison of UAV measurement and simulation result for z-axis accelerometer.

4. Results and Conclusion

This paper has demonstrated the modeling, design, and realization of IMU, which uses MEMS technology-based capacitive-type accelerometers and gyroscopes. The study also discussed the error types by comparing the real data with the simulated one. Remodeling has been performed to increase the accuracy, and the performance, and for this purpose, a complementary filter has been presented and combined with a magnetometer, accelerometer, and gyroscope for obtaining the ultimate design. As a result, in this study;

- Mathematical model of inertial measurement units has been formed and the model has been applied to the UAVs' inertial sensor measurements.
- Noise parameters have been estimated throughout the Allan variance analysis.
- Systematic errors of the inertial sensors have been simulated according to their datasheet parameters.
- Sensor filters and noise have been modeled and they have been also implemented in the simulation. Simulation results and UAV measurements have been compared to observe the efficiency of modeling.

• Magnetic sensors have also been taken into consideration to calculate the heading (yaw angle) of the UAV. Therefore, noise parameters and offset values of magnetic sensors are provided for simulation. Comparing the simulation results and

UAV heading measurements reveals that the suggested model fits the required modeling aspects.

This study can be extended with future works regarding the studies based on different types of gyroscopes and accelerometers instead of MEMS technology-based capacitive-type accelerometers and gyroscopes. Besides, as a future study, the proposed model can be implemented to the low budget, low-accuracy air platforms.

Author Contribution Statements
Bağış Altınöz: Validation, Investigation, Writing-review & editing, Visualization.
Anıl Cönger, Hüsamettin Eken: Investigation, Methodology
Sultan Can: Conceptualization, Writing-original draft, supervision.

Declaration of Competing Interests The authors declare that none of the work reported in this study could have been influenced by any known competing financial interests or personal relationships.

Acknowledgement We gratefully acknowledge the financial support by Scientific Research Projects of Ankara University (BAP) under Grant no 21Ö0443002. We would also like to thank SEMRUK team members and Dr. Tolga İnal for their support and contribution to the Teknofest UAV competition and TUSAŞ lift-up projects.

References

[1] Di Felice, F., Mazzini, A., Di Stefano, G., Romeo, G., Drone high-resolution infrared imaging of the Lusi mud eruption, *Mar. Pet. Geo.*, 90 (2018), 38-51, https://doi.org/10.1016/j.marpetgeo.2017.10.025.

[2] Aydin, B., Selvi, E., Tao, J., Starek, M. J., Use of fire-extinguishing balls for a conceptual system of drone-assisted wildfire fighting, *Drones*, 3 (1) (2019), 17, https://doi.org/10.3390/drones3010017.

[3] Bodnar, L., Restas, A., Qiang, X., Conceptual approach of measuring the professional and economic effectiveness of drone applications supporting forest fire management, Procedia Eng., 211 (2018), 8-17, https://doi.org/10.1016/j.proeng.2017.12.132.

[4] Restas, A., Drone applications for supporting disaster management, *World J. Eng. Technol.*, 3 (2015), 316-321, https://doi.org/10.4236/wjet.2015.33C047.

[5] Rao Mogili, U. M., Deepak, V. L., Review on application of drone systems in precision agriculture, *Procedia Comp. Sci.*, 133 (2018), 502-509, https://doi.org/10.1016/j.procs.2018.07.063.

[6] Marinello, F., Pezzuolo, A., Chiumenti, A., Sartori, L., Technical analysis of unmanned aerial vehicles (drones) for agricultural applications, *Eng. Rural Develop.*, (2016), 15.

[7] Morey, N. S., Mehere, P. N., Hedaoo, K., Agriculture drone for fertilizers and pesticides spraying, *Int. J. Eng. App. Technol.*, 3 (5) (2017).

[8] Shaw, I., History of U. S. drones, *Thinking (In) Security, Political Philosophy, and Robots,* (2014), https://understandingempire.wordpress.com/2-0-a-brief-history-of-u-s-drones/(2014).

[9] Stamp, J., Unmanned drones have been around since world war I [Online]. Retrieved from: http://www.smithsonianmag.com/arts-culture/unmanned-drones-have-been around- since-world-war-i-16055939/#ZOkewSDbAgEoRHhA.99.

[10] Luppicini, R., So, A., A technological review of commercial drone use in the context of governance, ethics, and privacy, *Technol. Soc.*, 46 (2016), 109-119, https://doi.org/10.1016/j.techsoc.2016.03.003.

[11] DigiKey, (2011). Available at: https://www.digikey.com/en/articles/using-an-accelero meter-for-inclination-sensing. [Accessed May 2023].

[12] Allan variance: noise analysis for gyroscopes, (2015). Available at: https://tele sens.co/wp-content/uploads/2017/05/AllanVariance5087-1.pdf. [Accessed May 2023].

[13] Qi, G., Ma, S., Guo, X., Li, X., Guo, J., High-order differential feedback control for quadrotor UAV: theory and experimentation, Electronics, 9 (12) (2020), 2001, https://doi.org/10.3390/electronics9122001.

[14] De Pasquale, G., Soma, A., Reliability testing procedure for MEMS IMUs applied to vibrating environments, Sensors, 10 (1) (2010), 456-474, https://doi.org/10.3390/s100100456.

[15] IEEE 952, IEEE standard specification format guide and test procedure for single-axis interferometric fiber optic gyros, (1997), https://doi.org/110.1109/IEEESTD. 1998.86153.

http://communications.science.ankara.edu.tr

Commun.Fac.Sci.Univ.Ank.Series A2-A3
Volume 66, Number 1, Pages 82-94 (2024)
DOI: 10.33769/aupse.1361266
ISSN 1303-6009 E-ISSN 2618-6462

Research Article; Received: September 15, 2023; Accepted: October 30, 2023

A MACHINE LEARNING-BASED FRAMEWORK USING THE PARTICLE SWARM OPTIMIZATION ALGORITHM FOR CREDIT CARD FRAUD DETECTION

Abdullah Asım YILMAZ[1]

[1]Department of Computer Engineering, Atılım University, Ankara, TÜRKİYE

Abstract. The detection of fraudulent activities in credit cards transactions presents a significant challenge due to the constantly changing and unpredictable tactics used by fraudsters, who take advantage of technological advancements to evade security measures and cause substantial financial harm. In this paper, we suggested a machine learning based methodology to detect fraud in credit cards. The suggested method contains four key phases, including data normalization, data preprocessing, feature selection, classification. For classification artificial neural network, decision tree, logistic regression, naive bayes, random forest while for feature selection particle swarm optimization is employed. With the use of a dataset created from European cardholders, the suggested method was tested. The experimental results show that the suggested method beats the other machine learning techniques and can successfully classify frauds with a high detection rate.

1. Introduction

Since the inception of credit cards and online payment systems, numerous individuals have discovered ways to deceive and unlawfully obtain credit card details in order to make unauthorized purchases. Consequently, a significant volume of fraudulent transactions occurs on a daily basis. In response, banks and e-commerce platforms are actively working to detect and prevent such fraudulent activities. They are leveraging deep learning (DL) and machine learning (ML) techniques to detect and halt fraudulent transactions before they are approved [1]. With the advancement of cutting-edge technology and global communication, fraudulent activities have been on the rise at an alarming rate [2]. As indicated by

Keywords. Credit card fraud detection, machine learning, particle swarm optimization.
abdullah.yilmaz@atilim.edu.tr-Corresponding author; 000-0002-3014-609X.

the Global Payments Report of 2015, credit cards emerged as the most widely utilized payment method worldwide in 2014 when compared to alternatives like e-wallets and bank transfers [3]. To conduct fraudulent actions using credit card services, cybercriminals usually target large-scale transactional services. Fraud using credit cards refers to transactions performed on atypical transaction patterns, inactive card, or unauthorized card use. [4]. Broadly speaking, credit card fraud can be classified into three main categories. These are conventional frauds (such as fake and stolen cards), merchant-related frauds (including merchant triangulation and collusion) and online frauds (involving counterfeit merchant websites) [5].

ML, a subset of Artificial Intelligence, has emerged as a prominent and widely discussed field in recent years. It has attracted significant attention, and numerous companies are now actively considering investments in machine learning to enhance their services. Machine learning involves employing a range of computer algorithms and statistical modeling techniques to enable computers to perform tasks without relying on explicit programming instructions [1]. Data mining refers to the procedure of extracting meaningful and insightful patterns from extensive collections of data, with the goal of uncovering descriptive, predictive, and valuable models [6]. By employing statistical and mathematical techniques, data mining techniques have the capacity to extract valuable information from large datasets. In the context of credit card fraud detection (CCFD), these techniques can be utilized to differentiate between normal and suspicious credit card transactions by identifying distinct characteristics [7]. On the other hand, ML is centered around learning and developing models to classify, cluster, or perform other tasks, rather than solely discovering valuable information like data mining [6]. Machine learning techniques have found extensive application across various domains in computer science. These domains include spam filtering, credit scoring, web search algorithms, recommendation systems, targeted advertising, fraud detection, classification problems and numerous other areas [8-12]. Machine learning classifiers function by constructing models based on sample inputs and utilizing them to make predictions or decisions, as opposed to relying solely on fixed program instructions. A wide range of machine learning approaches exists, each designed to address diverse and heterogeneous problems [13]. The model obtained would acquire knowledge from the "training data" and utilize that experiential knowledge to make predictions or carry out actions. DLs, which are a branch of ML, involve the use of artificial neural networks. Various methods such as restricted Boltzmann machines, recurrent neural networks, deep belief networks, generative adversarial networks, long short-term memory networks and convolutional neural networks are employed. A well-trained neural network would possess the ability to capture distinct relationships throughout the entire dataset [1].

2. Related Work

Awoyemi et al. [14] conducted a comparative analysis of different ML methods on credit card fraud dataset of European cardholders. In this study, it is used a hybrid sampling technique to solve the dataset's imbalance. The authors of study considered Naive Bayes (NB), K-Nearest Neighbors (KNN), and Logistic Regression (LR) ML methods and the implementation of the study was performed using the Python scripting language. Here, accuracy is used as the main metric value to measure how well each machine learning approach performs. According to the obtained empirical testing results, it was observed that LR, KNN, and NB each achieved accuracy levels of 54.86%, 97.69%, and 97.92%. In spite of the relatively strong performance of the KNN and NB, the authors did not take into account the possibility of utilizing a feature selection method.

Pumsirirat and Yan [15] proposed deep learning model to detect credit card fraud. The authors' goal is to concentrate on fraud cases that can't be identified using supervised learning. In this paper, proposed model was created using restricted Boltzmann machine and auto-encoder. In this study, the authors used tensor flow library from Google to implement their deep learning model include AE and RBM. According to the experimental results obtained on the datasets, suggested model produce high accuracy and Area Under Curve (AUC) score for huge fraud datasets. Sahin et al. [16] proposed a new cost-sensitive decision tree method to construct an fraud detection system for credit card transactions. This method uses support vector machines (SVM) and decision trees (DT). In this study, proposed method is compared with traditional classification models on a real world credit card dataset. The accuracy and true positive rate of the results show that the proposed method works better than other well-known methods.

Varmedja et al. [17] suggested CCFD approach utilizing ML on a credit card fraud dataset [18]. In order to solve the problem of imbalance of classes on CCFD dataset, the authors used synthetic minority oversampling technique (SMOTE). Multilayer perceptron, NB and RF ML techniques were employed to to assess the performance of suggested approach. According to the empirical results, the RF algorithm achieved the best fraud detection rate with 99.96% accuracy compared to MLP and NB ML methods.

3. Proposed Model for Credit Card Fraud Detection

This section presents the suggested framework for fraud detection. The suggested model for detecting fraud is ML-based and contains an optimized feature selection. This feature selection process provides with particle swarm optimization (PSO) algorithms. The suggested system's methodology, as shown in Figure 1, include four primary stages, namely, data normalization, data preprocessing, feature selection, and classification. First, data normalization is accomplished for to normalize the data in the training dataset. Second, the fraud detection data is pre-processed. SMOTE is used in this stage so as to sort out the class imbalance problem on the fraud detection data [19]. Third, feature selection operation is performed using PSO in order to get more accurate results in the classification process. Lastly, ML-based algorithms are used in order to carried out the classification processes.

For classification DT, RF, LR, Artificial Neural Network (ANN), and NB while for feature selection GA and PSO is employed in proposed method. The proposed method was tested using a dataset generated from European cardholders. The empirical results demonstrate that the suggested method beats the other machine learning techniques and can successfully classify frauds with a high detection rate.
An overview of the suggested model and the literature consulted for the suggested method are the two subsections that make up the remaining portion of this section. The model overview section first provides a detailed description of the proposed fraud detection framework for detecting fraud. Second, the literature consulted for the suggested method explain used feature selection technique and machine learning techniques that were employed in the proposed method.

3.1. **Overview of proposed model for fraud detection.** Figure 1 shows the architecture of the suggested methodology. In the first step, minimum-maximum scaling algorithm is utilized to normalize the fraud detection dataset in normalized data block (NDB) [20]. The mathematical formulation of minimum-maximum scaling algorithm is shown in Equation (1). In order to provide that each of the input values fall inside a predetermined range, scaling operation is performed. Second, NDB's normalized data is used to implement the PSO algorithm in PSO feature selection block (FSB). PSO creates candidate feature vector (FCV) an at each iteration of the PSO FSB. This FCV an is then used to test the trained models and train the models.

$$f_s = \frac{f - \min(f)}{\max(f - \min(f))} \tag{1}$$

3.2. **Literature Consulted for the Suggested Model.** The literature that was consulted in order to create the suggested fraud detection method is reviewed in this part, which is two subparts. First part provides a detailed explanation of the feature selection technique, and five well-known machine learning-based algorithms are described in second part.

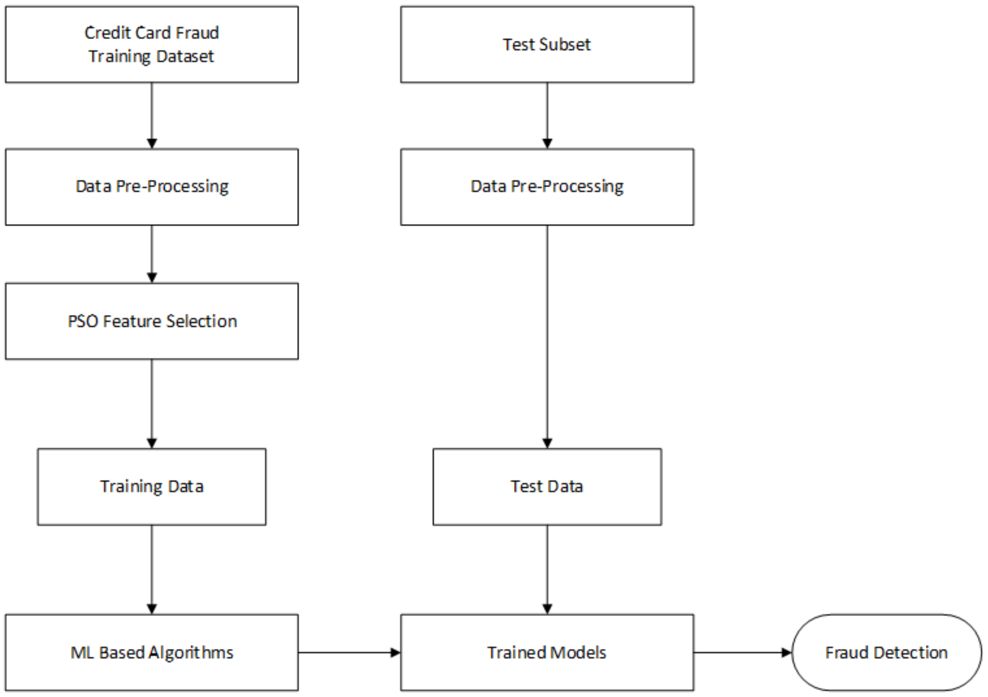


FIGURE 1. Architecture of the proposed framework.

3.2.1. *Feature Selection Methods.* PSO is an optimization method that draws inspiration from the group behavior of fish schools and flocks of birds. It aims to find ideal answer to issue by improving a group of potential solutions called particles through their cooperation and communication [21]. In PSO, particles represent potential solutions and move through a multi-dimensional search space. While particle's velocity dictates the magnitude and direction of motion, its position represents answer. According to their personal experience and the best experience of the entire group, particles modify their placements and velocities as they move through the search space [22]. The algorithm, whose flowchart is shown in the Figure 2, begins by initializing particles with random positions and velocities. Particles change their positions and velocities during each iteration using both the best position discovered by all other particles in the group (global best) and their individual best position (local best). This update is influenced by individual

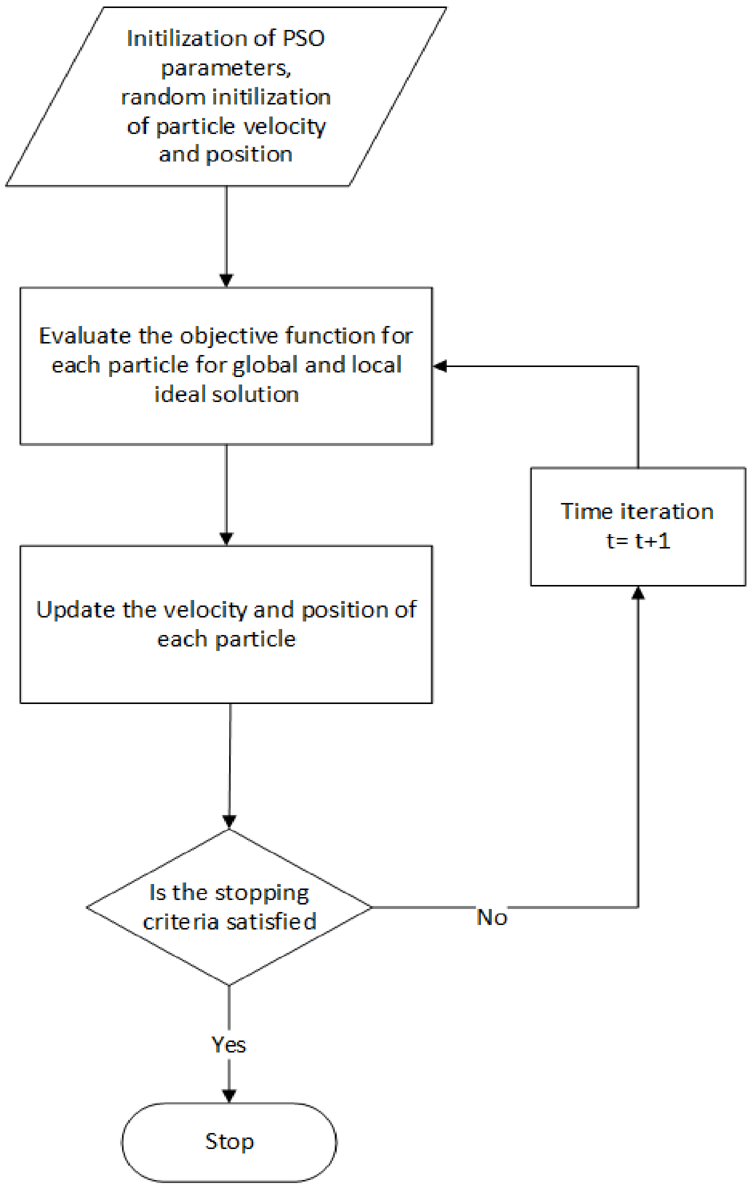

Initilization of PSO parameters, random initilization of particle velocity and position
Evaluate the objective function for each particle for global and local ideal solution
Update the velocity and position of each particle
Is the stopping criteria satisfied
No
Time iteration t= t+1
Yes
Stop

FIGURE 2. Flowchart of basic PSO algorithm.

experiences and the collective behavior of the group [23]. Social component and cognitive component are the two primary parts of the velocity update equation in PSO. The social component pulls particles in the direction of the ideal position discovered by any particle in the group, while the cognitive component directs particles toward their ideal position. These components balance exploration and exploitation, enabling efficient search in the solution space [24]. PSO iterates till a criterion for termination is satisfied, such as the required fitness value has been attained, and reaching a maximum number of iterations. The final positions of particles represent optimized solutions, or the best solution found [25]. PSO has been successfully used in various optimization problems [11], including function optimization, parameter tuning, neural network training, and feature selection.

3.2.2. *Machine Learning-Based Algorithms.* Despite its name, the Generalized Linear Models approach known as Logistic Regression is often referred to as Maximum Entropy. In this approach, a logistic function is used to characterize the probabilities that describe the potential outcomes of a single experiment. When there are one or more arguments, the output or result is determined using the logistic regression approach. The binary form of the output value is [26], which is either 0 or 1.

The data is divided using a condition in the Decision Tree classification. Data that meet the requirement are put in one class, while the rest are put in the other class. This procedure is iterative. There are various techniques for separation. These include similarity-based multi-attribute splitting, which compares terms in the document with predetermined words, and single-attribute splitting, which looks for the presence or absence of particular words for classification [27].

A learning technique for classification and regression is the Random Forest classification algorithm. Numerous decision trees are built throughout the training stage. To classify the fresh incoming state, the new state is sent to each of the trees. Each tree does categorization, and as a result, outputs a class. Based on majority vote, the output class is selected while taking into account the maximum number of related classes that the different trees can produce. The Random Forest approach is simple to understand and apply for both experts and laypeople, requiring little research and programming. Even those with little experience in statistics can use it with ease [28].

To extract features from a linear combination of data is the basic goal of the artificial neural networks technique and then model this obtained information as a nonlinear function of the features. Neural networks appear as a network diagram in which nodes are connected to each other in certain ways. Nodes are arranged in a layer. Architecturally, neural networks consist of three layers: hidden, ouput and input layer. Neural networks come in two varieties, namely, feedforward and feedback. Since the nodes are connected in only one direction in feedforward neural networks, this type of neural network is more suitable for sentiment analysis studies. Each link between nodes has a weight value that was determined by using the gradient descent approach to minimize the error function. A mathematical model that provides a value in two steps makes up a neuron. The weighted sum of the neuron input is determined in the first phase, and then an activation function is applied to this sum to produce output. With the use of input data from the complete network, the activation function, which is inherently nonlinear, can anticipate a previously learnt nonlinear function [29].

The NB approach is a classification algorithm based upon theorem of Bayes that has been utilized often in recent sentiment analysis studies. The assumptions made by naive Bayes classifiers are that the components (properties) of a given class are unrelated to one another in affinity. When attempting to divide the text into more than one class, the Naive Bayes approach is frequently utilized. [30].

4. Experimental Results and Discussions

The datasets used, the experimental findings, a review of the suggested model, and implementation specifics are all presented in this section. Python programming was used to carry out the proposed methodology's implementation. We used a personal PC with a 4.2 GHz Intel Core i5 11400H processor and 64 GB of RAM to carry out our studies. Additionally, a Linux system served as the setting for our tests.

4.1. **Dataset.** In our experimental studies, we utilize a fraud detection dataset, which consists credit card operations made by European cardholders. The total number of transactions in this dataset is 284807, with 0.172% of those transactions being fraudulent. This dataset contains 30 features, include (V1,..,V28), amount, and time. The dataset's attributes are all quantitative in type. The class (type of transaction) is represented by the last column. Here, zero value denotes non- fraudulent transaction while one value denotes fraudulent transaction. For purposes of data integrity and security, the features V1 through V28 are not named.

4.2. **Results and discussions.** Understanding the yield and performance of machine learning techniques requires an understanding of the assessment criteria used for classification processes. Evaluation metrics distinguish between model outcomes and explain how well the classification model performs [31]. Therefore, the suggested method's classification performance was indicated using accuracy, f-score, specificity, sensitivity metrics. The formulas in Table 1 were used to calculate these evaluation indicators. True negative, true positive, false negative, and false positive are represented in this table by the letters TN, TP, FN, and FP, respectively.

The experiments were run on a dataset of fraud [18]. FV = a1, a2, a3 was used in the classification procedure. The DT, RF, LR, ANN, and NB algorithms were trained and tested for each feature vector in FV. Tables 2, 3, and 4 provide the results. Both RF and ANN algorithms achieved the best test accuracy of 99.89% using a1, as shown in Table 2. However, in terms of precision, the RF approach produced the best results. Table 3's findings from the a2 test show that the RF method, with an accuracy of 99.88%, is the best model. The results that were attained when employing a3 are shown in Table 4. RF obtained 94.35% precision, 82.63 f-score and 99.92% accuracy rate for fraud detection in this case. a3 achieved the best outcomes when compared to a1, a2, and a3 results. Furthermore, the NB showed poorer performance concerning f1-score, precision and recall when compared to the results shown in Tables 2, 3, and 4.

To assess the effectiveness of the suggested paradigm, a comparison with other methodologies that are currently in use was also made. The accuracy values for both the proposed network and the other existing approaches are shown in Table 5. This table demonstrates with accuracy metric values that proposed method given and the majority of the suggested ML approaches that were applied outperformed suggested current methods in [33, 32, 17, 14]. Furthermore, the RF (implemented with v5) is the most accurate classifier in terms of classification. With an impressive accuracy of 99.92%, this model was able to detect credit card fraud.

TABLE 1. Assessment metrics formulations.

Evaluation metric	**Formula**
Sensitivity	TP / (TP+FN)
Accuracy	(TP+TN) / (TP+TN+FP+FN)
F-score	2*TP / (2*TP+FP+FN)
Specificity	TN / (TN+FP)

TABLE 2. Results of classification for feature vector a1.

Model	**Accuracy (%)**	**Recall (%)**	**Precision (%)**	**F-score (%)**
ANN	99.89	76.24	81.27	80.24
DT	99.87	74.12	74.34	68.66
NB	97.08	86.58	8.96	16.74
RF	99.89	75.56	88.49	82.57
LR	99.88	62.36	79.47	58.12

TABLE 3. Results of classification for feature vector a2.

Model	Accuracy (%)	Recall (%)	Precision (%)	F-score (%)
ANN	99.85	63.67	73.86	70.72
DT	99.81	66.28	61.52	57.45
NB	97.56	78.97	10.66	19.44
RF	99.88	74.26	83.73	79.36
LR	99.79	51.04	76.97	51.36

TABLE 4. Results of classification for feature vector a3.

Model	Accuracy (%)	Recall (%)	Precision (%)	F-score (%)
ANN	99.12	78.52	17.72	23.30
DT	99.81	73.34	68.65	69.51
NB	99.55	61.85	20.58	29.75
RF	99.92	73.36	94.35	82.63
LR	99.74	53.66	41.27	46.96

TABLE 5. Comparison with existing methods.

Model	Accuracy (%)
IF [14]	58.83
DT [32]	95.50
DT [34]	97.08
LR [33]	97.18
SVM [32]	97.50
LR [32]	97.70
NB [17]	99.23
PSO -NB (Proposed a3)	99.55
PSO-DT (Proposed a1)	99.87
PSO-LR (Proposed a1)	99.88
PSO-RF (Proposed a3)	**99.92**

5. CONCLUSION

ML-based technique for detecting credit card fraud was put out in this study. Using machine learning approaches, we provide a framework for fraud detection that incorporates improved feature selection. With PSO algorithms, this feature selection procedure is provided. Four primary processes make up the approach of the suggested system: data normalization, data preprocessing, feature selection, and classification. In order to normalize the data in the training dataset, data normalization is first completed. The data from the fraud detection procedure is then pre-processed. Third, feature selection operations are carried out utilizing PSO to produce more accurate classification results. In the output phase, a classifier based on ML (DT, RF, LR, ANN, NB) is used to perform the classification operations.

Three optimal feature vectors were produced when our suggested model was applied to the dataset of credit card transactions made by European cardholders. The experimental findings showed that the GA-RF (using v3) obtained an overall ideal accuracy of 99.92% using the PSO-selected features. Additionally, utilizing v1, other classifiers like the GA-DT were able to attain an astounding accuracy of 99.87%. Results from this study were better than those from earlier studies using similar techniques. In later studies, we intend to compare the results of more models and employ other global and metaheuristic search techniques for feature selection. We also intend to use several datasets to test the suggested strategy.

Declaration of Competing Interests The authors declare no conflict of interest.

References

[1] Raghavan, P., El Gayar, N., Fraud detection using machine learning and deep learning, *Int. Conf. on Comput. Intelligence and Knowledge Economy (ICCIKE)*, (2019), 334-339, https://doi.org/10.1109/ICCIKE47802.2019.9004231.

[2] Sisodia, D. S., Reddy, N. K., Bhandari, S., Performance evaluation of class balancing techniques for credit card fraud detection, *IEEE Int. Conf. on Power, Control, Signals and Instrumentation Engineering (ICPCSI)*, (2017), 2747-2752, https://doi.org/10.1109/ICPCSI.2017.8392219.

[3] WorldPay, Global payments report preview: The guide to the world of online payments, (2015). Available at: http://offers.worldpayglobal.com/rs/850-JOA856/images/Global PaymentsReportNov2015.pdf. [Accessed August 2023].

[4] Federal Trade Commission, Consumer sentinel network - data book for January, (2022). Available at: https://www.ftc.gov/. [Accessed August 2023].

[5] Bhatla, T. P., Prabhu, V., Dua, A., Understanding credit card frauds, *Cards Business Rev.*, 6 (2003), 1-15.

[6] Sahin, Y., Duman, E., Detecting credit card fraud by decision trees and support vector machines, *Int. MultiConf. of Engineers and Computer Scientists (IMECS)*, (2011), 1-5.

[7] Elkan, C., Magical thinking in data mining: Lessons from COIL challenge 2000, *ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, (2001), 426-431, https://doi.org/10.1145/502512.502576.

[8] Yilmaz, A. A., Guzel, M. S., Bostanci, E., Askerzade, I., A novel action recognition framework based on deep-learning and genetic algorithms. *IEEE Access*, 8 (2020), 100631-100644, https://doi.org/10.1109/ACCESS.2020.2997962.

[9] Aslan, Ö., Yilmaz, A. A., A new malware classification framework based on deep learning algorithms, *IEEE Access*, 8 (2021), 87936-87951, https://doi.org/10.1109/ACCESS.2021.3089586.

[10] Yilmaz, A. A., Guzel, M. S., Bostanci, E., Askerzade, I., A vehicle detection approach using deep learning methodologies, *Int. Conf. on Theoretical and Applied Computer Science and Engineering (ICTACSE)*, (2018), 64-71.

[11] Yilmaz, A. A., A novel hyperparameter optimization aided hand gesture recognition framework based on deep learning algorithms, *Trait. Du Signal*, 39 (3) (2022), 823-833, https://doi.org/10.18280/ts.390307.

[12] Yilmaz, A. A., Intrusion detection in computer networks using optimized machine learning algorithms, *Int. Informatics and Software Engineering Conf. (IISEC)*, (2022), 1-5, https://doi.org/10.1109/IISEC56263.2022.9998258.

[13] Yee, O. S., Sagadevan, S., Ahamed Hassain Malim, N. H., Credit card fraud detection using machine learning as data mining technique, *JTEC*, 10 (1-4) (2018), 23-27.

[14] Awoyemi, J. O., Adetunmbi, A. O., Oluwadare, S. A., Credit card fraud detection using machine learning techniques: A comparative analysis, *Int. Conf. on Computer Networks and Information (ICCNI)*, (2017), 1-9, https://doi.org/10.1109/ICCNI.2017.8123782.

[15] Pumsirirat, A., Liu, Y, Credit card fraud detection using deep learning based on auto-encoder and restricted Boltzmann machine, IJACSA, 9 (1) (2018), 18-25, https://doi.org/10.14569/IJACSA.2018.090103.

[16] Sahin, Y., Bulkan, S., Duman, E., A cost-sensitive decision tree approach for fraud detection, *Expert Syst. Appl.*, 40 (15) (2013), 5916-5923, https://doi.org/10.1016/j.eswa.2013.05.021.

[17] Varmedja, D., Karanovic, M., Sladojevic, S., Arsenovic, M., Anderla, A., Credit card fraud detection-machine learning methods, *Int. Sympos. INFOTEH-JAHORINA*, (2019), 1-5, https://doi.org/10.1109/INFOTEH.2019.8717766.

[18] Kaggle Datasets: The credit card fraud detection dataset, (2013). Available at: https://www.kaggle.com/mlg-ulb/creditcardfraud. [Accessed August 2023].

[19] Guo, S., Liu, Y., Chen, R., Sun, X., Wang, X. X., Improved SMOTE algorithm to deal with imbalanced activity classes in smart homes, *Neural Process. Lett.*, 50 (2) *(2019)*, 1503-1526, https://doi.org/10.1007/s11063-018-9940-3.

[20] Jain, A., Nandakumar, K., Ross, A., Score normalization in multimodal biometric systems, *Pattern Recognit.*, 38 (12) (2005), 2270-2285, https://doi.org/10.1016/j.patcog.2005.01.012.

[21] Khan, S. U., Yang, S., Wang, L., L. Liu., A modified particle swarm optimization algorithm for global optimizations of inverse problems, *IEEE Trans. Magn.*, 52 (3) (2016), 1-4, https://doi.org/10.1109/TMAG.2015.2487678.

[22] Selvi, V., Umarani, R., Comparative analysis of ant colony and particle swarm optimization techniques, *IJCA*, 5 (4) (2010), 1-6, https://doi.org/10.5120/908-1286.

[23] Shi, Y., Eberhart, R. C., Empirical study of particle swarm optimization, *Proceedings of the 1999 Congress on Evolutionary Computation (CEC99)*, (1999), 1945-1950, https://doi.org/10.1109/CEC.1999.785511.

[24] Krohling, R. A., Gaussian swarm: A novel particle swarm optimization algorithm, *IEEE Conf. on Cybernetics and Intelligent Systems*, (2004), 372-376, https://doi.org/10.1109/ICCIS.2004.1460443.

[25] Bai, Q., Analysis of particle swarm optimization algorithm, *Comput. Inf. Sci.*, 3 (1) (2010), 180-184, https://doi.org/10.5539/cis.v3n1p180.

[26] Tyagi, A., Sharma, N., Sentiment Analysis using logistic regression and effective word score heuristic, *IJET*, 7 (2) (2018), 20-23, https://doi.org/10.14419/ijet.v7i2.24.11991.

[27] Kaur, H., Mangat V., A survey of sentiment analysis techniques, *Int. Conf. on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)*, (2017), 921-925, https://doi.org/10.1109/I-SMAC.2017.8058315.

[28] Mamtesh, M., Mehla, S., Sentiment analysis of movie reviews using machine learning classifiers, *IJCA*, 182 (50) (2019), 25-28, https://doi.org/10.5120/ijca2019918756.

[29] Hemmatian, F., Sohrabi, M. K., A survey on classification techniques for opinion mining and sentiment analysis, *Artif. Intell. Rev.*, 52 (3) (2019), 1495-1545, https://doi.org/10.1007/s10462-017-9599-6.

[30] Alsaeedi, A., Khan, M. Z., A study on sentiment analysis techniques of twitter data, *IJACSA*, 10 (2) (2019), 361-374, https://doi.org/10.14569/IJACSA.2019.0100248.

[31] AnalyticsVidhya: Important model evaluation error metrics, (2019). Available at: https://www.analyticsvidhya.com/blog/2019/08/11important-model-evaluation-error-metrics. [Accessed August 2023].

[32] Khare, N., Sait, S. Y., Credit card fraud detection using machine learning models and collating machine learning models, *IJPAM*, 118 (20) (2018), 825-838.

[33] Seera, M., Lim, C. P., Kumar, A., Dhamotharan, L., Tan, K. H., An intelligent payment card fraud detection system, *Ann. Oper. Res.*, 8 (2021), 1-23, https://doi.org/10.1007/s10479-021-04149-2.

[34] Dornadula, V. N., Geetha, S., Credit card fraud detection using machine learning algorithms, *Procedia Comput. Sci.*, 165 (2019), 631-641, https://doi.org/10.1016/j.procs.2020.01.057.

http://communications.science.ankara.edu.tr

Commun.Fac.Sci.Univ.Ank.Ser. A2-A3
Volume 66, Number 1, Pages 95–117 (2024)
DOI:10.33769/aupse.1356138
ISSN 1303–6009 E-ISSN 2618–6462

COMMUNICATIONS SERIES A2-A3

Research Article; Received: September 06, 2023; Accepted: November 17, 2023

WHICH POOLING METHOD IS BETTER: MAX, AVG, OR CONCAT (MAX, AVG)

Yahya DOGAN[1]
[1]Computer Engineering Department, Faculty of Engineering,
Siirt University, Siirt, TÜRKİYE

Abstract. Pooling is a non-linear operation that aggregates the results of a given region to a single value. This method effectively removes extraneous details in feature maps while keeping the overall information. As a result, the size of feature maps is reduced, which decreases computing costs and prevents overfitting by eliminating irrelevant data. In CNN models, the max pooling and average pooling methods are commonly utilized. The max pooling selects the highest value within the pooling area and aids in preserving essential features of the image. However, it ignores the other values inside the pooling region, resulting in a significant loss of information. The average pooling computes the average values within the pooling area, which reduces data loss. However, by failing to emphasize critical pixels in the image, it may result in the loss of significant features. To examine the performance of pooling methods, this study comprised the experimental analysis of multiple models, i.e. shallow and deep, datasets, i.e. Cifar10, Cifar100, and SVHN, and pool sizes, e.g. $2x2$, $3x3$, $10x10$. Furthermore, the study investigated the effectiveness of combining two approaches, namely Concat (Max, Avg), to minimize information loss. The findings of this work provide an important guideline for selecting pooling methods in the design of CNNs. The experimental results demonstrate that pooling methods have a considerable impact on model performance. Moreover, there are variances based on the model and pool size.

1. Introduction

Deep learning has achieved remarkable results in a variety of tasks [1–3]. There are various successful architectures in this field [4–6], and convolutional neural networks (CNNs) are widely utilized, particularly in image classification and object recognition. CNN architectures typically consist of convolutional, pooling, and fully connected layers. Convolutional layers perform calculations by sliding learnable filters over the data to extract diverse features. Each filter is adjusted to recognize a specific feature and is applied to the entire dataset, allowing feature maps to be

Keywords. Pooling, deep learning, convolutional neural network.

yahyadogan@siirt.edu.tr-Corresponding author; 0000-0003-1529-6118.

created. Pooling layers reduce the size of feature maps by performing subsampling in the relevant region, thereby diminishing the network's computational cost. Fully-connected layers are used to generate the network's output. The feature maps acquired from the convolution layers are vectorized and then passed through one or more fully-connected layers to yield the network output.

Pooling layers are crucial in CNN architecture. These layers reduce the dimensionality of a feature map by obtaining a summary of a given region, but it results in information loss. As a result, selecting the proper method for pooling is critical for model performance. In CNNs, max pooling and average pooling methods are extensively utilized, each having its advantages and disadvantages. The max pooling takes the maximum activation to represent the pooling region of interest. This approach eliminates other features by focusing on the most important elements, resulting in a more specific feature map. It is very sensitive to the direction, size, and position of items in a given feature map. The average pooling, on the other hand, uses the average of all features to represent the region of interest in the feature map, allowing for the creation of a more generic feature map.

It is unclear which pooling method performs best under different conditions. In this study, different model architectures, i.e. shallow and deeper, different pooling sizes, e.g. $2x2$, $3x3$, ..., $10x10$, and different datasets, i.e. Cifar10 [7], Cifar100 [7], SVHN [8], were compared to evaluate the performance of the methods. Furthermore, the effect of concatenating these approaches to capture both significant features in images and overall patterns in data was experimentally studied.

The rest of the article is organized as follows. Section 2 is a brief review of previous research on pooling layers. Section 3 discusses the materials and processes in detail. Section 4 describes experimental studies and results. The article concludes with future directions.

2. Related Works

In general, two common pooling methods are utilized for reducing the size of feature maps: local pooling and global pooling. In local pooling, to minimize the dimensionality of the feature map, inferences are drawn from small neighboring regions within the feature map, e.g., $3x3$. In contrast, global pooling generates a single scalar value that represents the entire feature map. This research focuses on local pooling methods. Numerous studies have been conducted in this area since local pooling methods have a substantial impact on the success of CNNs. The studies can be categorized into four primary categories: value-based, probability-based, rank-based, and transformed-based methods [9].

In value-based pooling methods [10–12, 14–16], a value selection is determined based on a criterion among the pooling region's values. Mixed pooling [10] adds a parameter to choose between maximum and average pooling. Detail preservation pooling [12] is an adaptive pooling method that uses an inverse bilateral filter to amplify local spatial changes while retaining key structural details. It includes a learnable parameter for controlling the feature map's downsampling. Spatial pyramid pooling [13] creates fixed-length outputs regardless of input size and reduces information loss due to cropping. LEAP pooling [14] employs a shared linear filter for each feature channel to combine features in the pooling region, resulting in a reduction in the number of parameters and training errors. Dynamic correlation pooling [15] introduces a correlation pooling technique that relies on the Mahalanobis distance between adjacent pixels in an image. The output is dynamically determined by assessing the relationship between the Mahalanobis distance and a predefined threshold distance. The avg-topk pooling [16] method takes the average of the top-k activations in the pooling area, assisting in the preservation of significant features and addressing the problem caused by outliers and noises.

Probability-based pooling methods [17–21] calculate the probability of trading off between max and average pooling, thereby reducing error rates and preventing overfitting. Lp pooling [17] determines the pooling type based on a probability value P. $P = 1$ corresponds to Gaussian mean, while $P = \infty$ corresponds to maximum pooling. Stochastic pooling [18] substitutes a stochastic procedure for deterministic pooling operations. Within this approach, activations within the pooling region undergo normalization and are randomly selected through the utilization of a multinomial distribution. The max pooling dropout [19] combines max pooling and dropout techniques, and it has been experimentally shown to outperform maximum and scaled maximum probabilities. Song et al. (2018) [20] propose a sparsity-based stochastic pooling method that balances the advantages of max and average pooling by utilizing the sparsity level and control function of activations to obtain a feature representation. Hybrid pooling [21] combines both the max and average pooling methods by calculating maximum and average pooling values for the given pooling region. This combination is performed using a predefined probability.

Rank-based pooling methods [22–24] rank the activations within a specified pooling region and produce a pooled output based on weighted activation sums. During training, the weights are often learned via the back-propagation approach. This strategy overcomes the scale issues encountered by value-based pooling methods, allowing the model to capture critical activations and perform better. These methods are categorized into three groups based on weighting mechanisms: rank-based average pooling (RAP), rank-based weighted pooling (RWP), and rank-based stochastic pooling (RSP). The RAP approach considers the greatest activations in the pooling region, ignoring the rest, and then computes the average of these activations. RAP has a superior discriminative ability and provides a balanced approach between maximum and average pooling. The RWP strategy recognizes that each region is not equally significant. It takes the weighted average of each activation in the specified pooling region multiplied by a suitable coefficient. In RWP, reasonable weights are ascribed to activations based on their magnitudes, with the largest activation receiving the highest weight and the smallest activation receiving the least weight. The RSP strategy substitutes traditional pooling operations with a stochastic procedure in which activations are chosen based on probabilities derived from a multinomial distribution. RSP, as opposed to value-based stochastic pooling, computes probability based on activation order rather than activation value. The principal advantage of this strategy is the high degree of randomness in activation selection.

Domain-based pooling methods [25–27] use different domains to reduce spectral variance in feature maps, such as time, space, frequency, and wavelet domains. These studies often concentrate on the frequency domain and aim to filter out higher frequencies by removing low-frequency components. This transformation is achieved using various transformations such as Discrete Fourier Transform, Fast Fourier Transform, Hartley Transform, and Discrete Cosine Transform. These pooling methods can be sensitive to noise and perform filtering, but the computational cost can be high.

When various pooling approaches are compared across categories, it is clear that popular CNN models, e.g., VGG16 [28], ResNet [29], and EfficientNet [30], prefer the usage of max and average pooling methods due to their computational efficiency and lack of additional parameters. Therefore, in this study, the effects of these methods on multiple datasets, CNN models, pool sizes, and the combination of these methods have been investigated through experimental studies.

3. Material and Methods

3.1. **Pooling Methods.** In popular CNN models, average and max pooling methods are frequently utilized. In the average pooling, feature maps are divided into discrete rectangular regions, and sampling is performed by calculating the average activation value of each region. Mathematically, the expression for average pooling is as follows [31]:

$$f_{average} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{1}$$

Where x_i represents each activation value in the pooling area, and N denotes the number of activation values in that area. In max pooling, the activation within a pooling region with the highest value is chosen. This method is extensively used in CNN architectures and is reported to perform better in sparser encoding and simpler linear classifiers. As a consequence, its prominence has increased over the past few years. Mathematically, the expression for max pooling is as follows [34]:

$$f_{max}(x) = max_i(x_i) \tag{2}$$

Figure 1 depicts the (b) maximum pooling, (c) average pooling, and (d) Concat(Avg, Max) pooling outputs for a $4x4$ feature map (a) with a pool size of $2x2$.

Examining the efficacy of concatenating the two methods, i.e. ConCat (Avg, Max), is one of the primary contributions of this study. Extensive experimental

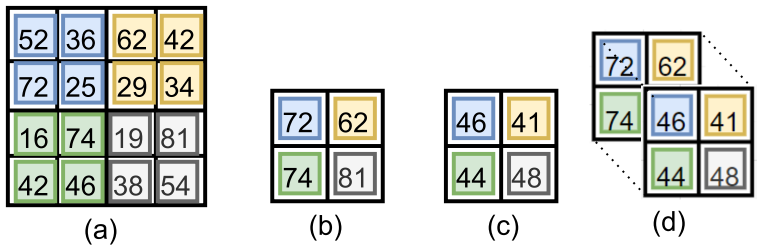


Figure 1. Illustration of (b) max/, (c) average pooling, and (d) Concat(Avg, Max) with a pooling area of size $2x2$ and stride of 2.

studies have been performed in this context to determine whether this method is required for small pooling sizes and whether it prevents information loss for large pooling sizes.

3.2. **Datasets.** Experimental studies were conducted using three benchmark datasets, namely CIFAR10, CIFAR100, and SVHN, to evaluate the performance of pooling methods. These datasets are commonly used to compare CNNs in the literature [9]. The CIFAR-10 dataset [7] is comprised of a total of $60,000$ RGB images with a resolution of $32x32$ and ten categories of labeling. The dataset consists of $60,000$ images divided into two sets: $50,000$ images for training and $10,000$ images for testing. Each class in the dataset has an equal number of examples, resulting in $6,000$ images per class. There exists a complete distinction between the classes. This particular dataset is frequently cited in the scientific literature for proposing new methodological approaches. The CIFAR100 dataset, which was introduced by Krizhevsky et al. (2014) [7], consists of 100 classes with 600 images in each class, totaling $60,000$ images. The images per class are separated into 500 training images and 100 test images. The resolution of the image is identical to CIFAR10, i.e., $32x32$. The dataset also has 20 superclasses in addition to the 100 classes. Consequently, each image has a ”fine” label that corresponds to its class and a ”coarse” label that corresponds to its superclass. The Street View House Numbers (SVHN) dataset [8] is a real-world image dataset commonly used to develop deep learning algorithms with minimal preprocessing and formatting requirements. It contains $600,000$ $32x32$ RGB images of printed numbers ranging from 0 to 9, serving as a number classification benchmark dataset. It is similar to MNIST, e.g., images composed of small cropped digits, but includes additional labeled data and tackles the more difficult, unsolved real-world problem of recognizing numbers and digits in images of natural scenes. The cropped images contain the digit of interest as well as adjacent digits and other distracting objects. Figure 2 depicted some examples taken from these datasets.

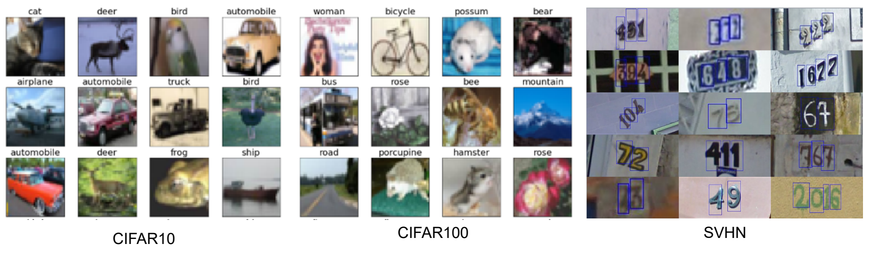


FIGURE 2. Random samples and classes from CIFAR10 (left), CIFAR100 (center), and SVHN (right).

3.3. **Models.** To assess the performance of pooling methods, two distinct models were utilized. Firstly, a shallow model based on the LeNet-5 [31] architecture was constructed to compare pooling methods. The architecture in question comprises a pair of convolutional layers, followed by two pooling layers, and finally, three fully connected (FC) layers. To accelerate convergence, the ReLU [32] activation function was used as opposed to the tanh activation function in the original LeNet-5. The first convolutional layer employed six learnable filters, while the second convolutional layer employed sixteen filters, both with a filter size of $5x5$. To preserve the resolution of the feature maps, a padding value of 2 was utilized in the convolutional layers. The stride value in the pooling layers was set equal to the kernel size to avoid overlapping. Figure 3 illustrates the LeNet-5-based architecture.

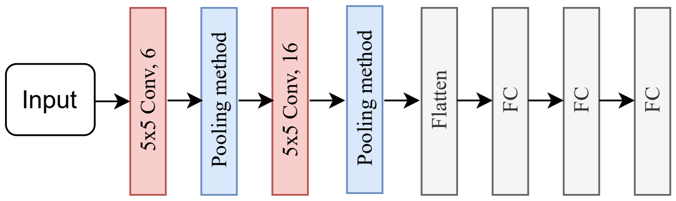


FIGURE 3. LeNet-5 based model.

For every method of pooling, the models were trained from scratch for 30 epochs. In other words, pre-trained weights were not used. As the loss function, cross-entropy was used, and stochastic gradient descent was employed as the optimization algorithm. [33]. The learning rate was set to $1e-3$, momentum was set to 0.9, and the batch size to 16.

Secondly, to compare pooling methods in a different and deeper model, a model based on ResNet-9 [29] was used. The model in question comprises eight convolutional layers, followed by four pooling layers, and finally, one FC layer. All convolutional layers utilized $3x3$ filters. After the convolutional layers, batch normalization layers were included to avoid the common vanishing gradient issue in deep learning models. ReLU activation layers were applied after the respective layers. Two residual networks were utilized to facilitate the training of the deep model. A residual network is defined as a kind of neural network that includes skipping connections, which perform identity mappings and integrate layer outputs with the input via element-wise addition. Figure 4 depicts the specifics of the model developed based on ResNet-9. During the training phase, the LeNet-5 model's hyperparameters were utilized.

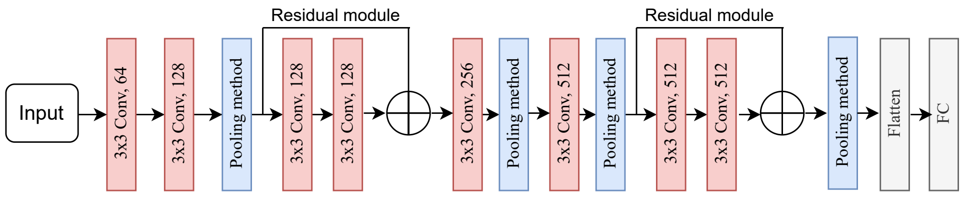


FIGURE 4. ResNet-9 based model.

3.4. **Metrics.** In this study, classification-based models were created and the performance of pooling methods was compared. A set of metrics known as the confusion matrix is utilized to assess the efficacy of classification models. The confusion matrix comprises four different concepts: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). A confusion matrix is typically represented in a table format as follows:

TABLE 1. Confusion matrix. TP: The number of in which the true class is positive and the model predicts it to be positive, FP: The number of in which the true class is negative but the model predicts it to be positive. TN: The number of instances in which the true class is negative and the model predicts it to be negative. FN: The number of instances in which the true class is positive but the model predicts it to be negative.

	Actually Positive	Actually Negative
Predicted Positive	TN	FP
Predicted Negative	FN	TP

To compare the performance of methods using the confusion matrix, 4 metrics were used: accuracy, precision, recall, and F1 score. The accuracy metric quantifies the proportion of instances correctly predicted by a model. This metric is used to evaluate a classification model's overall efficacy.

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{3}$$

The precision metric indicates how many of the values predicted as positive by a model are actually positive. This metric limits the number of false positive predictions made by a classification model.

$$Precision = \frac{TP}{(TP + FP)} \quad (4)$$

The recall metric indicates how many values that should be classified as positive are predicted as positive by a model. This metric is used to limit a classification model's number of false negative predictions.

$$Recall = \frac{TP}{(TP + FN)} \quad (5)$$

The F1 score metric is the harmonic mean of the metrics for precision and recall. This metric takes false positive and false negative predictions made by a classification model into account.

$$F1score = 2 * \frac{(Precision * Recall)}{(Precision + Recall)} \quad (6)$$

4. Experiment and Results

In this section, the performance of pooling methods was compared for different models (Lenet-5 and Resnet-9), different datasets (Cifar10, Cifar100, and SVHN), and different pool sizes ($2x2$, $3x3$, ..., $10x10$). In this context, firstly, the performance of methods for different pool sizes was examined using the Lenet-5 model and the Cifar10 dataset. For each pooling method, the Lenet-5 model was trained from scratch for 30 epochs. Figure 5 shows the training accuracy achieved for 4 different pool sizes, i.e., $2x2$, $5x5$, $7x7$, and $10x10$. Analyzing the graphs reveals that the average pooling begins with low accuracy for all pool sizes. In later epochs, it is observed that the average pooling achieves higher accuracy values for small pool sizes, e.g. $2x2$ and $5x5$, but falls behind other methods for larger pool sizes, such as $7x7$ and $10x10$.

Table 2 compares pooling methods quantitatively using the Lenet-5 model and the Cifar10 dataset. Examining Table 2 reveals that the average pooling method is more effective for small pool sizes, such as $2x2$, and $3x3$. Due to the fact that the stride value is equal to the kernel size, there is no overlap. In other words, as the size of the pool increases, there is an expected increase in information loss, resulting in a decrease in the accuracy of all models. As the pool size increases, it can be observed that the efficacy of the average pooling degrades more rapidly than other methods. Depending on the pool size, the max pooling outperforms the average pooling after a certain point, such as $6x6$. Within the scope of this study, the proposed Concat(Avg, Max) method demonstrated superiority over the other two methods as the pool size increased.

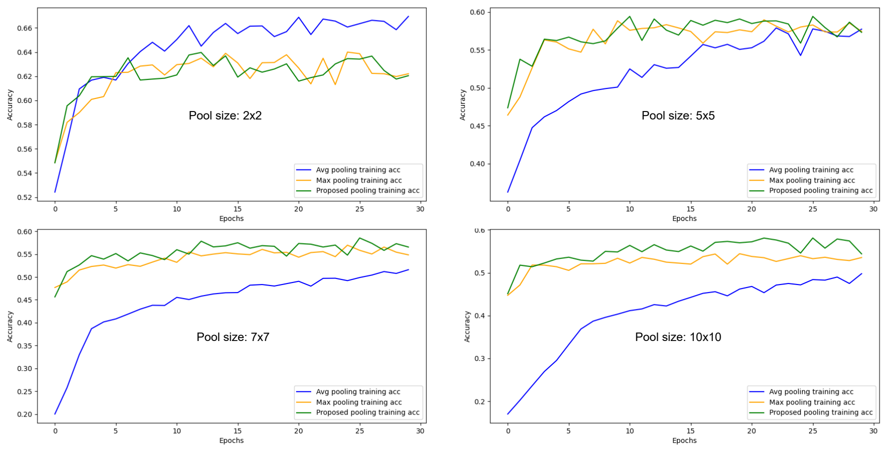


FIGURE 5. Training graphs of pooling methods for $2x2$, $5x5$, $7x7$, and $10x10$ pool sizes using the Lenet-5 model and the Cifar10 dataset.

In Figure 6, the performance of the aforementioned pooling methods for different pool sizes is presented graphically using the F1 score metric. As observed, the average pooling has a significant advantage over the other methods for a $2x2$ pool size. Interestingly, for a $5x5$ pool, all methods yield virtually identical results. It is evident that the Concat(Avg, Max) method is more successful for large pools ($\geq 6x6$). Since the average pooling takes the average of values in the pooling area, the influence of high activations diminishes as the pool size increases, resulting in a general performance decline.

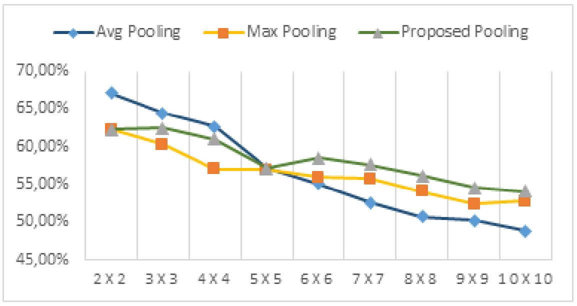


FIGURE 6. F1 score performances of pooling methods for varying pool sizes using the Lenet-5 model and the Cifar10 dataset.

Secondly, experimental studies were conducted to evaluate the performance of the methods on a more challenging dataset. In this context, the Lenet-5 model

Table 2. Compare pooling methods for Model: LeNet-5 Dataset: Cifar10

Pool size	Method	Accuracy	Precision	Recall	F1 score
2x2	Avg pooling	**0.6695**	**0.6763**	**0.6695**	**0.6707**
	Max pooling	0.6221	0.6304	0.6221	0.6230
	Concat (Avg-Max)	0.6205	0.6298	0.6205	0.6225
3x3	Avg pooling	**0.6454**	**0.6495**	**0.6454**	**0.6447**
	Max pooling	0.6087	0.6168	0.6087	0.6030
	Concat (Avg-Max)	0.6218	0.6366	0.6218	0.6250
4x4	Avg pooling	**0.6224**	**0.6382**	**0.6224**	**0.6267**
	Max pooling	0.5781	0.5874	0.5781	0.5705
	Concat (Avg-Max)	0.6151	0.6171	0.6151	0.6098
5x5	Avg pooling	**0.5773**	0.5749	**0.5773**	0.5710
	Max pooling	0.5743	0.5966	0.5743	0.5694
	Concat (Avg-Max)	0.5731	**0.5983**	0.5731	**0.5715**
6x6	Avg pooling	0.5600	0.5546	0.5600	0.5506
	Max pooling	0.5664	0.5857	0.5664	0.5594
	Concat (Avg-Max)	**0.5862**	**0.6065**	**0.5862**	**0.5848**
7x7	Avg pooling	0.5370	0.5360	0.5370	0.5254
	Max pooling	0.5614	0.5774	0.5614	0.5573
	Concat (Avg-Max)	**0.5786**	**0.5956**	**0.5786**	**0.5766**
8x8	Avg pooling	0.5160	0.5129	0.5160	0.5069
	Max pooling	0.5484	0.5670	0.5484	0.5409
	Concat (Avg-Max)	**0.5655**	**0.5814**	**0.5655**	**0.5615**
9x9	Avg pooling	0.5111	0.5063	0.5111	0.5023
	Max pooling	0.5360	0.5483	0.5360	0.5240
	Concat (Avg-Max)	**0.5495**	**0.5754**	**0.5495**	**0.5453**
10x10	Avg pooling	0.4979	0.4919	0.4979	0.4880
	Max pooling	0.5356	0.5446	0.5356	0.5281
	Concat (Avg-Max)	**0.5441**	**0.5732**	**0.5441**	**0.5409**

and the Cifar100 dataset were used to evaluate various pool sizes. Figure 7 shows the training accuracy achieved by the methods for $2x2$, $5x5$, $7x7$, and $10x10$ pool sizes. Similar to the Cifar10 dataset, it can be observed that the average pooling has lower initial accuracy than other methods for all pool sizes during the initial epochs. In later epochs, the average pooling performs better than other methods for small pool sizes, but it lags for large pool sizes. In contrast to the Cifar10 dataset, the methods in this dataset attain a high accuracy value for the $2x2$ pool size at the $5th$ epoch and then decline. This observation indicates that it is preferable to use the model from the epoch with the highest accuracy rather than the model obtained after the training process.

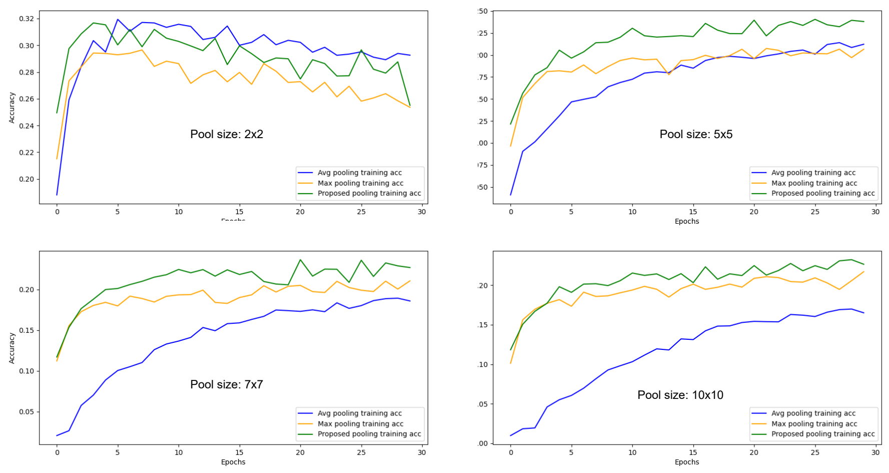


FIGURE 7. Training graphs of pooling methods for $2x2$, $5x5$, $7x7$, and $10x10$ pool sizes using the Lenet-5 model and the Cifar100 dataset.

Using the LeNet-5 model and the Cifar100 dataset, Table 3 illustrates quantitatively the efficacy of various pooling methods. Similar to the Cifar10 dataset, it is observed that the average pooling performs better for smaller pool sizes. Specifically, for a $2x2$ pool size, the closest performing method, i.e., max pooling, exhibits a 3.41% increase in performance. As the pool size increases, the efficacy of the average pooling diminishes relative to that of other methods. The Concat(Avg, Max) method has a 6.28% higher success rate than the average pooling when evaluating the $10x10$ pool size.

In Figure 8, the performance of different pooling methods for various pool sizes is graphically presented using the LeNet-5 model and the Cifar100 dataset. It can be observed that the average pooling yields the highest performance for a $3x3$ pool. In the Cifar10 dataset, the highest score was obtained for a $2x2$ pool size. This suggests that the performance of a model can vary based on the size of the data pool, even when using the same model and different datasets. Similar to the Cifar10 dataset, when the average pooling method is used and the pool size is increased, the model's success significantly decreases. Concat(Avg, Max) obtains the highest performance for larger pool sizes, while max pooling also achieves relatively close scores.

Cifar10 and Cifar100 datasets are partially similar datasets. In the continuation of the study, experimental work was conducted using the SVHN dataset, which contains more diverse images, and a Lenet-5-based model. Table 4 provides quantitative performance results for varied pool sizes. Similar to other datasets, the average pooling exhibited higher performance for smaller pool sizes. As the

Table 3. Compare pooling methods for Model: Lenet Dataset: Cifar100

Pool size	**Method**	**Accuracy**	**Precision**	**Recall**	**F1 score**
2x2	Avg pooling	**0.2926**	**0.2957**	**0.2926**	**0.2878**
	Max pooling	0.2536	0.2676	0.2536	0.2522
	Concat (Avg-Max)	0.2551	0.2676	0.2551	0.2537
3x3	Avg pooling	**0.3314**	0.3355	**0.3314**	**0.3228**
	Max pooling	0.2703	0.2804	0.2703	0.2588
	Concat (Avg-Max)	0.3026	**0.3382**	0.3026	0.2935
4x4	Avg pooling	**0.2949**	0.3052	0.2949	**0.2817**
	Max pooling	0.2594	0.2717	**0.2594**	0.2451
	Concat (Avg-Max)	0.2930	**0.3154**	0.2930	0.2762
5x5	Avg pooling	0.2264	0.2235	0.2264	0.2048
	Max pooling	0.2130	0.2188	0.2130	0.1903
	Concat (Avg-Max)	**0.2352**	**0.2670**	**0.2352**	**0.2167**
6x6	Avg pooling	0.2122	0.2040	0.2122	0.1901
	Max pooling	0.2065	0.2134	0.2065	0.1853
	Concat (Avg-Max)	**0.2380**	**0.2702**	**0.2380**	**0.2170**
7x7	Avg pooling	0.2042	0.1902	0.2042	0.1797
	Max pooling	0.2068	0.2071	0.2068	0.1845
	Concat (Avg-Max)	**0.2323**	**0.2597**	**0.2323**	**0.2105**
8x8	Avg pooling	0.1858	0.1841	0.1858	0.1636
	Max pooling	0.2105	0.2133	0.2105	0.1887
	Concat (Avg-Max)	**0.2268**	**0.2494**	**0.2268**	**0.2082**
9x9	Avg pooling	0.1836	0.1803	0.1836	0.1593
	Max pooling	0.2166	0.2185	0.2166	0.1944
	Concat (Avg-Max)	**0.2321**	**0.2428**	**0.2321**	**0.2089**
10x10	Avg pooling	0.1651	0.1654	0.1651	0.1424
	Max pooling	0.2171	0.2138	0.2171	0.1923
	Concat (Avg-Max)	**0.2265**	**0.2340**	**0.2265**	**0.2052**

pool size increased, the efficacy of the average method decreased more compared to other methods. The average method demonstrated roughly half the efficacy of the Concat(Avg, Max) method when evaluating the $10x10$ pool size. In general, for a shallow model, the average pooling performed better for smaller pool sizes, whereas max pooling, particularly Concat(Avg, Max), stood out as the pool size increased. Figure 9 illustrates the performance of methods for the Lenet-5 model and the SVHN dataset. The performance of the average pooling was comparable for $2x2$, $3x3$, and $4x4$ pool sizes. As with other datasets, its performance decreased substantially as the size of the pool grew. The Concat(Avg, Max) method obtained an F1 score of 72.61% for the $10x10$ pool size, while the average pooling achieved a performance of 29.41%. In general evaluation of the Lenet-5 model, or in other

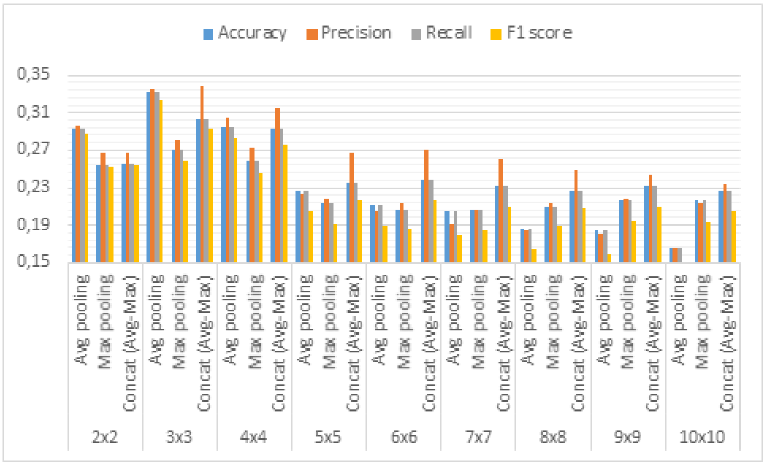


FIGURE 8. Accuracy, precision, recall, and F1 score results of pooling methods with various pool sizes in the LeNet model using the Cifar100 dataset.

words, a shallow model, the selection of pooling method is important depending on the pool size used. If the pool size is small, the average pooling should be preferred, whereas if the pool size is large, the maximum pooling method should be chosen. If the number of model parameters increase is not a concern, the Concat(Avg, Max) method can be applied to larger pool sizes.

In the continuation of the study, the performance of pooling methods was compared using ResNet-9 which has a deeper and different architecture. In this context, since the relevant model contained a greater number of pooling layers, the images were resized to a resolution of $224x224$. Figure 10 depicts the accuracy graphs acquired during the training phase for various pool sizes and each method. The shallow Lenet-5 model performed better with smaller pool sizes, such as $2x2$ and $3x3$, when the average pooling was applied to all the datasets used in the study. Figure 11 illustrates the efficacy of methods for various pool sizes utilizing the ResNet-9 model and the Cifar10 dataset. In addition, Table 5 quantifies all of the experimental results. In contrast to the Lenet-5 model, the average pooling in ResNet-9 trails behind other methods for all pool sizes. In addition, as the pool size increases, the performance of the average pooling method diminishes in both the shallow and deep models. While the max pooling and the Concat(Avg, Max) method attain comparable performance, the Concat(Avg, Max) method is observed to perform better for larger pool sizes.

Examined quantitatively, the max pooling for a $2x2$ pool attained a 13.12% higher F1 score than the average pooling. The difference between Concat(Avg, Max) and max pooling was less than 1%, and the max pooling was found to be superior. Considering a larger pool size of $5x5$, the Concat(Avg, Max) obtained the

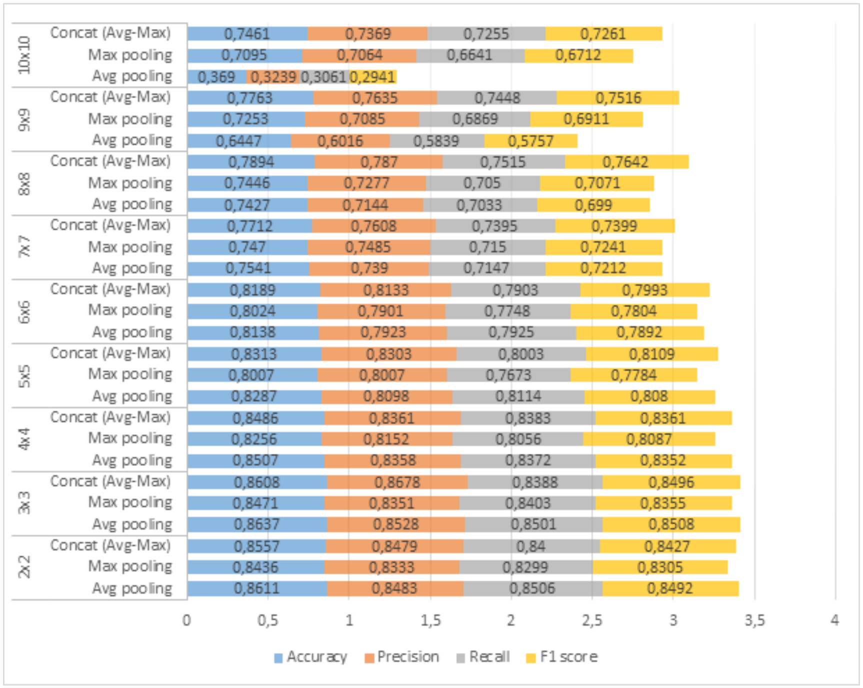


FIGURE 9. Accuracy, precision, recall, and F1 score results of pooling methods with various pool sizes in the LeNet-5 model using the SVHN dataset.

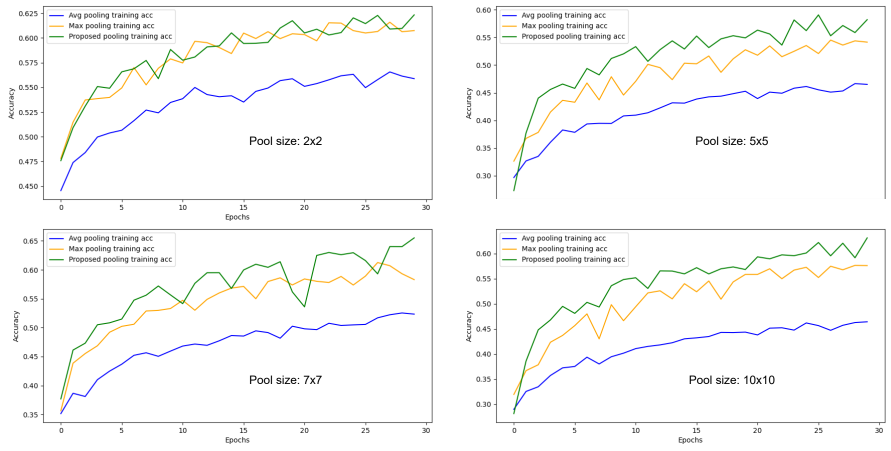


FIGURE 10. Training graphs of pooling methods for $2x2$, $5x5$, $7x7$, and $10x10$ pool sizes using the Resnet-9 model and the Cifar10 dataset.

best performance. It achieved a 21.3% improvement in the F1 score over the average pooling and a 7.44% improvement over the maximum pooling for this specific pool size. Notably, as the pool size increases, the methods' efficacy typically decreases due to information loss. Concat(Avg, Max) performed better than the $2x2$ pool size for $7x7$ and $8x8$ pool sizes. Due to the increased number of pooling layers in this experimental setting, the image resolution became inadequate; therefore, the stride value was fixed at 6 after a pool size of $6x6$.

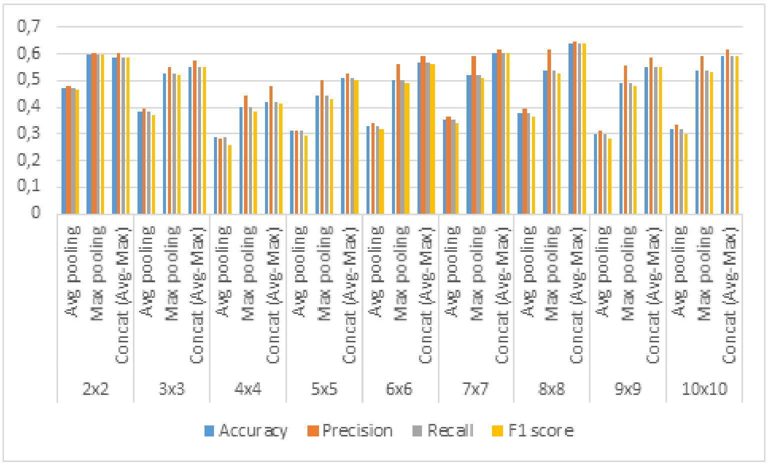


FIGURE 11. Accuracy, precision, recall, and F1 score results of pooling methods with various pool sizes in the ResNet-9 model using the Cifar10 dataset.

Using the ResNet-9 model and the Cifar100 dataset, experimental analysis was conducted to assess the performance of the methods on a similar but more difficult dataset. Table 6 provides a quantitative presentation of the results obtained for the respective experimental setting. In addition, to facilitate the comparison of methods for various pool sizes, the scores are presented in Figure 12 as a stacked bar graph. The graph reveals a situation comparable to the Cifar10 dataset. While the max pooling outperforms other methods for small pools, it is clear that the Concat(Avg, Max) method performs better as the pool size increases. Similar to the previous situation, $8x8$ and $9x9$ pool sizes received higher scores than $2x2$.

Finally, the performance of the methods was compared using the ResNet-9 model and the SVHN dataset. The quantitative results obtained for this experimental setup are provided in Table 7. Additionally, the F1 score of the methods for each pool size is given in Figure 13. In contrast to other experimental results, it was observed that even for the smallest pool size, the Concat(Avg, Max) method yielded higher scores. Notable is the fact that, for a $2x2$ pool size, the Concat(Avg, Max) method produced F1 score enhancements of 2.84% and 12.29%, compared to the maximum and average methods, respectively.

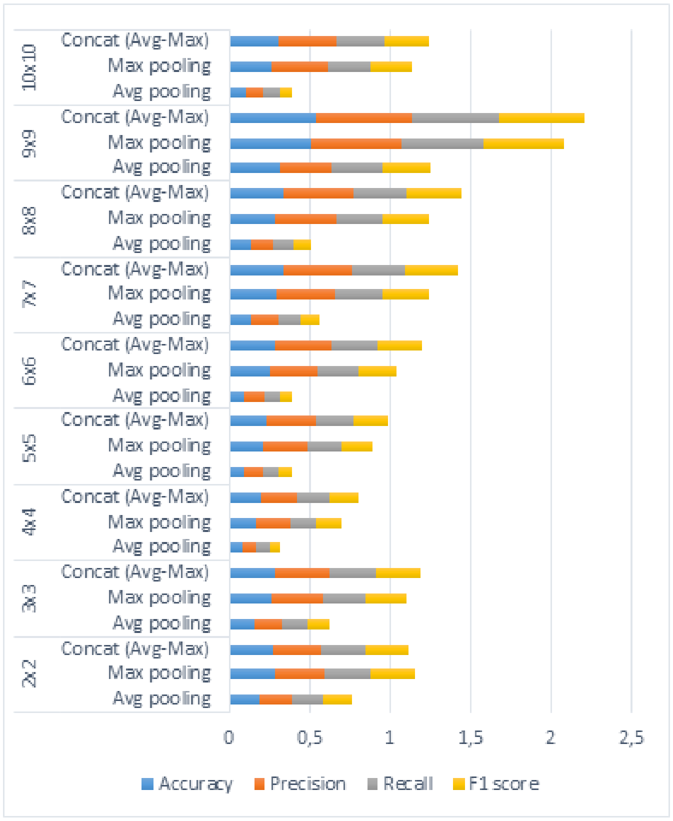


FIGURE 12. Accuracy, precision, recall, and F1 score results of pooling methods with various pool sizes in the ResNet-9 model using the Cifar100 dataset.

5. Conclusion

In this study, the success of pooling methods, which play a crucial role in the success of CNNs and are frequently preferred, was compared experimentally for various models, datasets, and pool sizes. In addition, the experimental performance of concatenating maximum and average pooling methods to reduce information loss under similar conditions was examined. The concept underlying this method is to incorporate the strengths of both pooling methods. While maximum pooling is effective at preserving the input's most prominent characteristics, average pooling is effective at capturing the data's general tendencies. It was hypothesized that the final feature map derived by combining the feature maps generated by these two methods would provide a more accurate representation of the input by incorporating both the most prominent features and general trends. The experimental

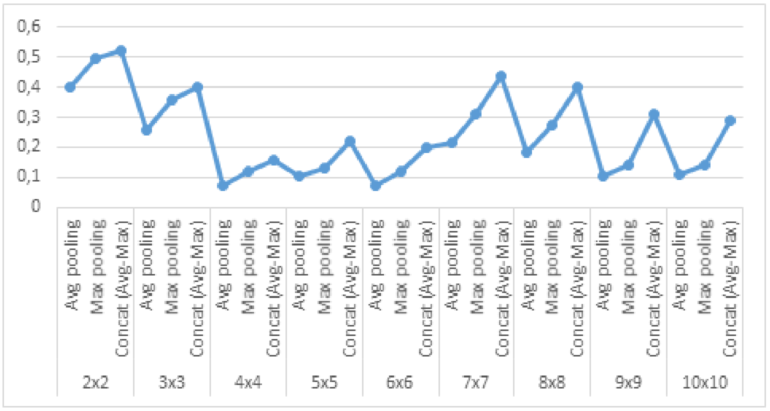


FIGURE 13. F1 score results of pooling methods with various pool sizes in the ResNet-9 model using the SVHN dataset.

results revealed that the superiority of a particular pooling method varied depending on the application scenario. In the shallow model, i.e., LeNet-5, the average pooling outperformed all other methods for small pool sizes across all datasets used in the study. As the pool size increased, the efficacy of the average pooling method deteriorated and fell behind that of the maximum pooling. For large pool sizes, i.e. $> 5x5$, the Concat(Avg, Max) outperformed the other two algorithms. In the ResNet-9 deep model, the max pooling performed better than the other methods for small pool sizes. In this model, the average pooling lagged behind the other methods for all pool sizes. As the size of the pool increased, the Concat(Avg, Max) method provided a more accurate representation and obtained better results. For the SVHN dataset, this method yielded the highest scores for all pool sizes. This study guides selecting pooling methods depending on the model and pool size. The experimental results demonstrated that the pooling method has a significant effect on model performance. Moreover, there were model and pool size-dependent variations among different pooling methods. Future research will investigate the impact of using multiple pooling methods at various levels of deep CNN models.

Declaration of Competing Interests The author declares that there is no competing interest regarding the publication of this paper.

Acknowledgement The numerical calculations reported in this paper were partially performed at TUBITAK ULAKBIM, High Performance and Grid Computing Center (TRUBA resources).

References

[1] Ataş, I., Human gender prediction based on deep transfer learning from panoramic dental radiograph images, *Trait. du Signal*, 39 (5) (2022), 1585, http://dx.doi.org/10.18280/ts.390515.

[2] Ataş, M., Özdemir, C., Ataş, İ., Ak, B., Özeroğlu, E, Biometric identification using panoramic dental radiographic images withfew-shot learning, *Turk. J. Electr. Eng.*, 30 (3) (2022), 1115-1126, http://dx.doi.org/10.55730/1300-0632.3830.

[3] Ozdemir, C., Gedik, M. A., Kaya, Y., Age estimation from left-hand radiographs with deep learning methods, *Trait. du Signal*, 38 (6) (2021), http://dx.doi.org/10.18280/ts.380601.

[4] Krizhevsky, A., Sutskever, I., Hinton, G. E., Imagenet classification with deep convolutional neural networks, *Commun. ACM*, 60 (6) (2017), 84-90, http://dx.doi.org/10.1145/3065386.

[5] Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Dosovitskiy, A., Mlp-mixer: An all-mlp architecture for vision, *Adv. Neural Inf. Process. Syst.*, 34 (2021), 24261-24272, https://arxiv.org/abs/2105.01601.

[6] Meng, L., Li, H., Chen, B. C., Lan, S., Wu, Z., Jiang, Y. G., Lim, S. N., Adavit: Adaptive vision transformers for efficient image recognition, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (2022), 12309-12318, http://dx.doi.org/10.1109/cvpr52688.2022.01199.

[7] Krizhevsky, A., Nair, V., rey Hinton, G., CIFAR-10 dataset, (2014), Available at: https://www.cs.toronto.edu/ kriz/cifar.html.

[8] Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A., The street view house numbers (SVHN) dataset, (2016). Available at: https://www.kaggle.com/datasets/stanfordu/street-view-house-numbers.

[9] Akhtar, N., Ragavendran, U., Interpretation of intelligence in cnn-pooling processes: a methodological survey, *Neural. Comput. Appl.*, 32 (3) (2020), 879-898, http://dx.doi.org/10.1007/s00521-019-04296-5.

[10] Yu, D., Wang, H., Chen, P., Wei, Z., Mixed pooling for convolutional neural networks, *International Conference on Rough Sets and Knowledge Technology*, (2014), 364-375, http://dx.doi.org/10.1007/978-3-319-11740-9_34.

[11] Dogan Y., A new global pooling method for deep neural networks: Global average of top-k max-pooling, *Trait. du Signal*, 40 (2) (2023), 577-587, http://dx.doi.org/10.18280/ts.400216.

[12] Saeedan, F., Weber, N., Goesele, M., Roth, S., Detail-preserving pooling in deep networks, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2018), 9108-9116, http://dx.doi.org/10.1109/cvpr.2018.00949.

[13] He, K., Zhang, X., Ren, S., Sun, J., Spatial pyramid pooling in deep convolutional networks for visual recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*,37 (9) (2015), 1904-1916, http://dx.doi.org/10.1109/tpami.2015.2389824.

[14] Sun, M., Song, Z., Jiang, X., Pan, J., Pang, Y. Learning pooling for convolutional neural network, *Neurocomputing*, 224, (2017), 96-104, http://dx.doi.org/10.1016/j.neucom.2016.10.049.

[15] Wang, F., Huang, S., Shi, L., Fan, W., The application of series multi-pooling convolutional neural networks for medical image segmentation, *Int. J. Distrib. Sens. Netw.*, 13 (12) (2017), http://dx.doi.org/10.1177/1550147717748899.

[16] Özdemir, C., Avg-topk: A new pooling method for convolutional neural networks, *Expert Syst. Appl.*, (2023), 119892, http://dx.doi.org/10.1016/j.eswa.2023.119892.

[17] Sermanet, P., Chintala, S., LeCun, Y., Convolutional neural networks applied to house numbers digit classification, *Proceedings of the 21st International Conference on Pattern Recognition*, (2012), 3288-3291, https://doi.org/10.48550/arXiv.1204.3968.

[18] Fei, J., Fang, H., Yin, Q., Yang, C., Wang, D., Restricted stochastic pooling for convolutional neural network, *Proceedings of the 10th International Conference on Internet Multimedia Computing and Service*, (2018), 1-4, http://dx.doi.org/10.1145/3240876.3240919.

[19] Wu, H., Gu, X., Max-pooling dropout for regularization of convolutional neural networks, *International Conference on Neural Information Processing*, (2015), 46-54, http://dx.doi.org/10.1007/978-3-319-26532-2_6.

[20] Song, Z., Liu, Y., Song, R., Chen, Z., Yang, J., Zhang, C., Jiang, Q., A sparsitybased stochastic pooling mechanism for deep convolutional neural networks, *Neural Netw.*, 105 (2018), 340-345, http://dx.doi.org/10.1016/j.neunet.2018.05.015.

[21] Tong, Z., Aihara, K., Tanaka, G., A hybrid pooling method for convolutional neural networks, *International Conference on Neural Information Processing*, (2016), 454-461, http://dx.doi.org/10.1007/978-3-319-46672-9_51.

[22] Shahriari, A., Porikli, F., Multipartite pooling for deep convolutional neural networks, arXiv:1710.07435, (2017), http://arxiv.org/abs/1710.07435.

[23] Kumar, A., Ordinal pooling networks: for preserving information over shrinking feature maps, arXiv:1804.02702, (2018), http://arxiv.org/abs/1804.02702.

[24] Kolesnikov, A., Lampert, C. H. Seed, Expand and constrain: three principles for weakly-supervised image segmentation, *European Conference on Computer Vision*, (2016), 695-711, http://dx.doi.org/10.1007/978-3-319-46493-0_42.

[25] Williams, T., Li, R., Wavelet pooling for convolutional neural networks, *International Conference on Learning Representations*, (2018).

[26] Rippel, O., Snoek, J., Adams, R. P., Spectral representations for convolutional neural networks, *Adv. Neural Inf. Process. Syst.*, (2015), 28, https://doi.org/10.48550/arXiv.1506.03767.

[27] Wang, Z., Lan, Q., Huang, D., Wen, M., Combining fft and spectral-pooling for efficient convolution neural network model, *2016 2nd International Conference on Artificial Intelligence and Industrial Engineering (AIIE)*, (2016), 203-206, http://dx.doi.org/10.2991/aiie-16.2016.47.

[28] Simonyan, K., Zisserman, A., Very deep convolutional networks for large-scale image recognition, arXiv:1409.1556, (2014), https://doi.org/10.48550/arXiv.1409.1556.

[29] He, K., Zhang, X., Ren, S., Sun, J., Deep residual learning for image recognition, *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (2016), 770-778, http://dx.doi.org/10.1109/cvpr.2016.90.

[30] Tan, M., Le, Q., Efficientnet: Rethinking model scaling for convolutional neural networks, *International conference on machine learning (ICML)*, (2019), 6105-6114, https://doi.org/10.48550/arXiv.1905.11946.

[31] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., Gradient-based learning applied to document recognition, *Proc. IEEE*, 86 (1998), 2278–2324, http://dx.doi.org/10.1109/5.726791.

[32] Nair, V., Hinton, G. E., Rectified linear units improve restricted boltzmann machines, *Proceedings of the 27th international conference on machine learning (ICML)*, (2010), 807-814.

[33] Bottou, L., Stochastic gradient descent tricks, Neural Networks: Tricks of the Trade: Second Edition, (2012), 421-436, http://dx.doi.org/10.1007/978-3-642-35289-8_25.

[34] Boureau Y.-L., Le Roux N., Bach F., Ponce J., LeCun Y., Ask the locals: multiway local pooling for image recognition, in Computer Vision, *IEEE International Conference*, (2011), 2651-2658, http://dx.doi.org/10.1109/iccv.2011.6126555.

Appendix

Table 4. Compare pooling methods for Model: Lenet Dataset: SVHN

Pool size	**Method**	**Accuracy**	**Precision**	**Recall**	**F1 score**
2x2	Avg pooling	**0.8611**	**0.8483**	**0.8506**	**0.8492**
	Max pooling	0.8436	0.8333	0.8299	0.8305
	Concat (Avg-Max)	0.8557	0.8479	0.8400	0.8427
3x3	Avg pooling	**0.8637**	0.8528	**0.8501**	**0.8508**
	Max pooling	0.8471	0.8351	0.8403	0.8355
	Concat (Avg-Max)	0.8608	**0.8678**	0.8388	0.8496
4x4	Avg pooling	**0.8507**	0.8358	0.8372	0.8352
	Max pooling	0.8256	0.8152	0.8056	0.8087
	Concat (Avg-Max)	0.8486	**0.8361**	**0.8383**	**0.8361**
5x5	Avg pooling	0.8287	0.8098	**0.8114**	0.8080
	Max pooling	0.8007	0.8007	0.7673	0.7784
	Concat (Avg-Max)	**0.8313**	**0.8303**	0.8003	**0.8109**
6x6	Avg pooling	0.8138	0.7923	**0.7925**	0.7892
	Max pooling	0.8024	0.7901	0.7748	0.7804
	Concat (Avg-Max)	**0.8189**	**0.8133**	0.7903	**0.7993**
7x7	Avg pooling	0.7541	0.7390	0.7147	0.7212
	Max pooling	0.7470	0.7485	0.7150	0.7241
	Concat (Avg-Max)	**0.7712**	**0.7608**	**0.7395**	**0.7399**
8x8	Avg pooling	0.7427	0.7144	0.7033	0.6990
	Max pooling	0.7446	0.7277	0.7050	0.7071
	Concat (Avg-Max)	**0.7894**	**0.7870**	**0.7515**	**0.7642**
9x9	Avg pooling	0.6447	0.6016	0.5839	0.5757
	Max pooling	0.7253	0.7085	0.6869	0.6911
	Concat (Avg-Max)	**0.7763**	**0.7635**	**0.7448**	**0.7516**
10x10	Avg pooling	0.3690	0.3239	0.3061	0.2941
	Max pooling	0.7095	0.7064	0.6641	0.6712
	Concat (Avg-Max)	**0.7461**	**0.7369**	**0.7255**	**0.7261**

TABLE 5. Compare pooling methods for Model: ResNet-9 Dataset: Cifar10

Pool size	**Method**	**Accuracy**	**Precision**	**Recall**	**F1 score**
2x2	Avg pooling	0.4689	0.4764	0.4689	0.4646
	Max pooling	**0.5984**	**0.6031**	**0.5984**	**0.5958**
	Concat (Avg-Max)	0.5845	0.6016	0.5845	0.5859
3x3	Avg pooling	0.3825	0.3971	0.3825	0.3730
	Max pooling	0.5272	0.5498	0.5272	0.5221
	Concat (Avg-Max)	**0.5476**	**0.5730**	**0.5476**	**0.5465**
4x4	Avg pooling	0.2870	0.2822	0.2870	0.2584
	Max pooling	0.4014	0.4392	0.4014	0.3812
	Concat (Avg-Max)	**0.4209**	**0.4782**	**0.4209**	**0.4134**
5x5	Avg pooling	0.3118	0.3098	0.3118	0.2904
	Max pooling	0.4439	0.5032	0.4439	0.4290
	Concat (Avg-Max)	**0.5045**	**0.5261**	**0.5045**	**0.5034**
6x6	Avg pooling	0.3266	0.3391	0.3266	0.3154
	Max pooling	0.5034	0.5621	0.5034	0.4923
	Concat (Avg-Max)	**0.5675**	**0.5907**	**0.5675**	**0.5636**
7x7	Avg pooling	0.3507	0.3660	0.3507	0.3411
	Max pooling	0.5193	0.5892	0.5193	0.5093
	Concat (Avg-Max)	**0.6044**	**0.6175**	**0.6044**	**0.6018**
8x8	Avg pooling	0.3749	0.3930	0.3749	0.3668
	Max pooling	0.5352	0.6164	0.5352	0.5264
	Concat (Avg-Max)	**0.6413**	**0.6444**	**0.6413**	**0.6400**
9x9	Avg pooling	0.3009	0.3090	0.3009	0.2805
	Max pooling	0.4882	0.5558	0.4882	0.4778
	Concat (Avg-Max)	**0.5520**	**0.5830**	**0.5520**	**0.5504**
10x10	Avg pooling	0.3173	0.3375	0.3173	0.2982
	Max pooling	0.5357	0.5911	0.5357	0.5291
	Concat (Avg-Max)	**0.5926**	**0.6139**	**0.5926**	**0.5887**

Table 6. Compare pooling methods for Model: ResNet-9 Dataset: Cifar100

Pool size	**Method**	**Accuracy**	**Precision**	**Recall**	**F1 score**
2x2	Avg pooling	0.1893	0.2055	0.1893	0.1779
	Max pooling	**0.2841**	**0.3104**	**0.2841**	**0.2809**
	Concat (Avg-Max)	0.2714	0.3031	0.2714	0.2659
3x3	Avg pooling	0.1567	0.1701	0.1567	0.1383
	Max pooling	0.2617	0.3237	0.2617	0.2549
	Concat (Avg-Max)	**0.2884**	**0.3369**	**0.2884**	**0.2766**
4x4	Avg pooling	0.082	0.0899	0.082	0.064
	Max pooling	0.1658	0.2103	0.1658	0.1527
	Concat (Avg-Max)	**0.1954**	**0.2315**	**0.1954**	**0.1789**
5x5	Avg pooling	0.097	0.1138	0.0972	0.0783
	Max pooling	0.2050	0.2845	0.2050	0.1929
	Concat (Avg-Max)	**0.2334**	**0.3034**	**0.2334**	**0.2150**
6x6	Avg pooling	0.0937	0.1247	0.0937	0.0767
	Max pooling	0.2476	0.3048	0.2476	0.2350
	Concat (Avg-Max)	**0.2880**	**0.3448**	**0.2880**	**0.2751**
7x7	Avg pooling	0.1383	0.1625	0.1383	0.1223
	Max pooling	0.2916	0.3696	0.2916	0.2873
	Concat (Avg-Max)	**0.3323**	**0.4265**	**0.3323**	**0.3296**
8x8	Avg pooling	0.1305	0.1389	0.1305	0.1111
	Max pooling	0.2886	0.3755	0.2886	0.2867
	Concat (Avg-Max)	**0.3382**	**0.4307**	**0.3382**	**0.3309**
9x9	Avg pooling	0.3186	0.3212	0.3186	0.2969
	Max pooling	0.5047	0.5709	0.5047	0.5032
	Concat (Avg-Max)	**0.5418**	**0.5936**	**0.5418**	**0.5355**
10x10	Avg pooling	0.1001	0.1139	0.1001	0.076
	Max pooling	0.2674	0.3421	0.2674	0.2552
	Concat (Avg-Max)	**0.2997**	**0.3616**	**0.2997**	**0.2840**

TABLE 7. Compare pooling methods for Model: ResNet-9 Dataset: SVHN

Pool size	**Method**	**Accuracy**	**Precision**	**Recall**	**F1 score**
2x2	Avg pooling	0.4552	0.4219	0.4044	0.4014
	Max pooling	0.5512	0.5250	0.4914	0.4959
	Concat (Avg-Max)	**0.5754**	**0.5680**	**0.5104**	**0.5243**
3x3	Avg pooling	0.3539	0.3074	0.2704	0.2591
	Max pooling	0.4305	0.3911	0.3585	0.3559
	Concat (Avg-Max)	**0.4706**	**0.4427**	**0.4016**	**0.3988**
4x4	Avg pooling	0.1903	0.0921	0.1140	0.0745
	Max pooling	0.2392	0.1759	0.1551	0.1200
	Concat (Avg-Max)	**0.2718**	**0.2278**	**0.1833**	**0.1598**
5x5	Avg pooling	0.2004	0.1565	0.1392	0.1066
	Max pooling	0.2741	0.1871	0.1791	0.1292
	Concat (Avg-Max)	**0.3168**	**0.2451**	**0.2358**	**0.2182**
6x6	Avg pooling	0.1979	0.1028	0.1163	0.075
	Max pooling	0.2374	0.1228	0.1645	0.1216
	Concat (Avg-Max)	**0.3178**	**0.2419**	**0.2278**	**0.1982**
7x7	Avg pooling	0.2974	0.2744	0.2307	0.2166
	Max pooling	0.3953	0.3726	0.3170	0.3130
	Concat (Avg-Max)	**0.4897**	**0.4928**	**0.4359**	**0.4364**
8x8	Avg pooling	0.2400	0.2424	0.1983	0.1819
	Max pooling	0.3657	0.3459	0.2862	0.2709
	Concat (Avg-Max)	**0.4671**	**0.4681**	**0.3995**	**0.4028**
9x9	Avg pooling	0.1851	0.1374	0.1426	0.1041
	Max pooling	0.2769	0.1997	0.1856	0.1439
	Concat (Avg-Max)	**0.3929**	**0.3665**	**0.3222**	**0.3093**
10x10	Avg pooling	0.2017	0.1523	0.1477	0.1098
	Max pooling	0.2680	0.2503	0.1823	0.1393
	Concat (Avg-Max)	**0.3754**	**0.3875**	**0.3057**	**0.2871**

http://communications.science.ankara.edu.tr

Commun.Fac.Sci.Univ.Ank.Series A2-A3
Volume 66, Number 1, Pages 118-131 (2024)
DOI: 10.33769/aupse.1378578
ISSN 1303-6009 E-ISSN 2618-6462

Research Article; Received: October 19, 2023; Accepted: January 23, 2024

HEATMAP CREATION WITH YOLO-DEEP SORT SYSTEM CUSTOMIZED FOR IN-STORE CUSTOMER BEHAVIOR ANALYSIS

Murat ŞİMŞEK[1] and Mehmet Kemal TEKBAŞ[2]

[1]Department of Artificial Intelligence Engineering, Ostim Technical University, Ankara, TÜRKİYE
[2]Department of Electrical and Electronics Engineering, Ankara University, Ankara, TÜRKİYE

Abstract. Due to the limitations of the hardware system, analysis of retail stores has caused problems such as excessive workload, incomplete analysis, slow analysis speed, difficult data collection, non-real-time data collection, passenger flow statistics, and density analysis. However, heatmaps are a viable solution to these problems and provide adaptable and effective analysis. In this paper, we propose to use the deep sequence tracking algorithm together with the YOLO object recognition algorithm to create heatmap visualizations. We will present key innovations of our customized YOLO-Deep SORT system to solve some fundamental problems in in-store customer behavior analysis. These innovations include our use of footpad targeting to make bounding boxes more precise and less noisy. Finally, we made a comprehensive evaluation and comparison to determine the success rate of our system and found that the success rate was higher than the systems we compared in the literature. The results show that our heatmap visualization enables accurate, timely, and detailed analysis.

1. Introduction

Heatmap is a data visualization technique that uses color-coded representations to present patterns and variations in data [1]. Heatmap analysis provides an effective tool for representing and interpreting data, allowing data analysts and researchers to

Keywords. Heatmap visualization, retail analysis, YOLO, object detection, object tracking, machine learning, deep learning.
murat.simsek@ostimteknik.edu.tr-Corresponding author; 0000-0002-8648-3693
kemaltekbas@gmail.com; 0009-0008-9877-6021.

gain insight, make informed decisions, and communicate findings more efficiently [2, 3]. It is widely used to analyze and display large data sets, especially in the fields of statistics, business intelligence, science, and image processing [4]. The idea of retail analysis is an indispensable need for the economy [5]. The idea of increasing sales and optimizing in-store business activities by understanding human behavior and the need to optimize business processes using data from in-store traffic and interactions has been a great need in the retail world [6]. With the innovations that came with the increase in machine learning, these ideas began to develop, and better research began to be conducted to solve these problems. According to our studies, the idea of heatmap analysis in retail stores is a suitable solution to our problems [7]. The analysis method we developed reduces the time spent on customer analysis without using excessive cameras and labor to obtain data at a specific point. In this article, we present real-time analytics on retail stores to detect hot spots. In this research, we used the deep sorting algorithm along with the YOLO algorithm to create our heatmap visualization, and these algorithms are suitable for our work. YOLO (You Only Look Once) was chosen for its single-stage detector capabilities [8], allowing it to be significantly faster on object detection tasks. Additionally, YOLO offers the advantage of readily available pre-trained models, making it easy to start a project quickly. The smaller size of the model suite also contributes to its suitability, especially for projects with limited budgets or resource-constrained environments [9]. Moreover, the combination of YOLO with the Deep SORT algorithm allowed us to not only detect objects efficiently but also track them accurately over time [10]. This capability is essential for analyzing customer behavior in retail environments where individuals' movements and interactions must be comprehensively monitored, providing a high success rate even when objects are viewed from different angles [11]. Leveraging these algorithms, we were able to create precise and informative heatmap visualizations that play an important role in understanding and optimizing customer experiences in retail spaces [12]. It is also worth noting that there is still a large gap between theoretical research and real-world applications for reliably detecting people for heatmap visualization. We examined the information in the literature and the areas where heatmaps are most commonly used [13]. After our tests on the model determined to be used in our in-store analyses, we will explain in detail the mathematical and theoretical explanations of our effective and appropriate approaches to the techniques and methods used in the algorithms. Our goal is to make this process faster with higher accuracy and to detect and track multiple objects simultaneously by getting more accurate data from the real-time model.

2. Model Developments: Mathematical and Theoretical Overview

Before delving into the intricacies of our configuration, it's crucial to establish a solid foundation by elucidating the core principles underpinning key techniques like NMS (Non-Maximum Suppression), Gaussian blur, and BBOX (Bounding Box). A bounding box is a rectangular region that encloses an object of interest detected by the YOLO model during object detection [14]. We want to avoid loss of perspective and adjust where the tracked objects stand without visual confusion, we need to make the bounding box function unique and effective for our purpose.

Gaussian blur is a preprocessing step applied to the input image of the YOLO model. This operation refers to a filtering operation used to soften or blur the input image. We must create an adaptive structure that will work in any situation to enable the analysis of the collected data [15]. It should ensure that the data collected and the heatmap to be superimposed on it create a clear picture of the participation rates in any case. For our heatmap to work properly in crowded and complex areas, we can reduce the noise by smoothing the pixels.

Non-maximum suppression (NMS) is a post-processing technique used to filter out redundant object detections and improve the accuracy of the final output [16]. NMS is applied to the bounding box predictions generated by the YOLO model. When YOLO performs object detection, it divides the input image into a grid and predicts bounding boxes and class probabilities for each grid cell. However, multiple bounding boxes may overlap and detect the same object. NMS helps eliminate duplicate detections by selecting the most confident and accurate bounding boxes while discarding the rest. Due to our needs, we need to make the adjustments.

2.1. **Human-based foot bounding box method.** Instead of using the given Yolo bounding boxes in the literature, we made improvements to our case and started to detect objects from the base point. The function converts these bounding box coordinates to a different representation known as "xywh" format, where x, y represent the center of the bounding box and w, h represent the width and height of the bounding box.

A 2D bounding box, defined as a set B with xb , yb coordinates. Where {(xmin, ymin), (xmin, ymax), (xmax, ymin), (xmax, ymax)} are the four corners of the bounding box for our object [17].

$$B = \{ (x_c , y_c) \in R^2 \mid x_{min} < x_b < x_{max} , y_{min} < y_b < y_{max} \} \quad (1)$$

$$\text{Bounding Box Weight} = | x_{min} - x_{max} | = w \quad (2)$$

$$\text{Bounding Box Height} = | y_{min} - y_{max} | = h \quad (3)$$

$$x_c = (y_{min} + w / 2) \quad (4)$$

$$y_c = (y_{max} - h / 8) \quad (5)$$

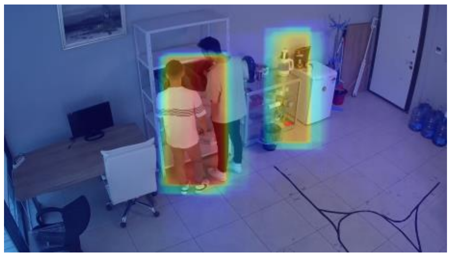


FIGURE 1. Existing bounding box (demo test environment).

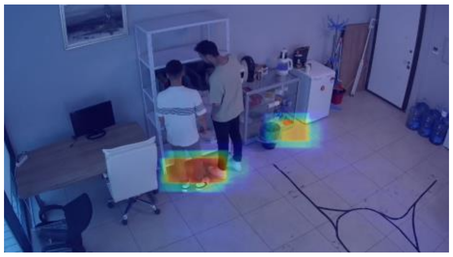


FIGURE 2. Human-based foot bounding box method (demo test environment).

2.2. **Enhancing object detection: NMS hyperparameter optimization.** There are two useful hyperparameters we should mention in this technique reduce the noise in the configuration.

The Intersection over Union (IoU) is a crucial metric used in assessing the performance of object detection and classification algorithms. It quantifies the similarity between a predicted bounding box and a ground truth (target) bounding box. The formula computes the ratio of the area of overlap (intersection) between these two boxes to the area of their union. In the formula, 'Target ∩ Prediction' represents the intersection, which is the shared region between the target and prediction boxes, while 'Target U Prediction' signifies the union, representing the total area covered by both boxes. IoU yields a value between 0 and 1, with higher values indicating better detection accuracy. Typically, a predefined threshold is used, and predictions exceeding this threshold are considered correct detections [18].

$$(\text{IoU}) = (\text{Target} \cap \text{Prediction}) / (\text{Target U Prediction}) \quad (6)$$

Another crucial aspect of object detection and classification is the confidence threshold. Each bounding box prediction generated by the model is accompanied by a confidence score, which signifies the model's level of confidence in that specific detection. During the post-processing step, known as Non-Maximum Suppression (NMS), bounding boxes with confidence scores falling below the predetermined threshold are disregarded. To be considered a valid prediction, the confidence score for a given bounding box must be greater than or equal to the threshold value. In such cases, the model's output, which may include class labels and bounding box coordinates, is accepted as a valid detection. This confidence threshold is an essential tool for filtering out low-confidence predictions, ensuring that only high-confidence detections contribute to the final results, and enhancing the precision of object detection.

There is a need to protect values below the threshold. Especially in crowded environments, it is desirable to consider objects below a certain threshold. This ensures that some items that have low scores but could be important are not eliminated. Thanks to these ideologies, the threshold parameters have been adjusted to ignore the complexity and get a clean analysis of the areas of interest.

Due to our experiments in a crowded place like a retail store, we decided to take the intersection over the union threshold as 0.5 and the confidence threshold as 0.4, which would indicate that there is a 40% chance that the object exists in that box.

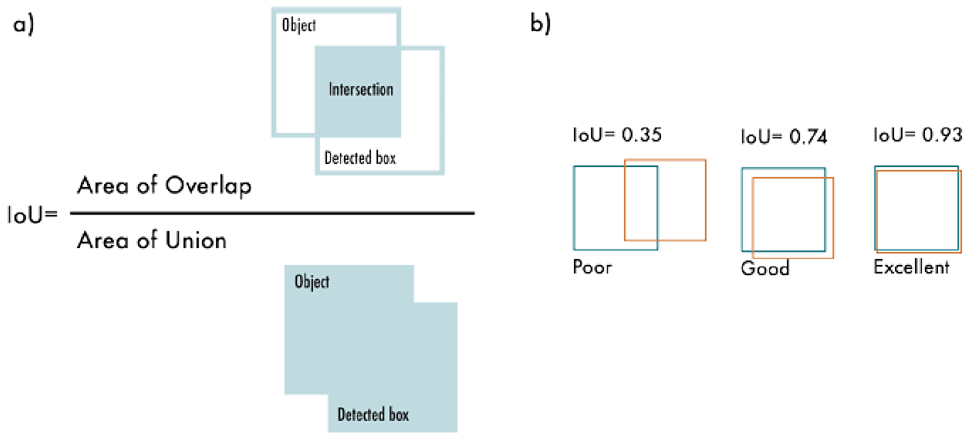


FIGURE 3. Intersection over Union (IoU). a) The IoU is calculated by dividing the intersection of the two boxes by the union of the boxes b) examples of three different IoU values for different box locations [19].

2.3. **Hyperparameter optimization for enhanced Gaussian blur.** Creating Gaussian data is a critical element to increase the sensitivity of the results obtained when creating heatmaps and to provide a more reliable analysis. Components of Gaussian data allow object details and tracking to be more accurate, as well as reduce noise in the data.

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{7}$$

Where x is the distance from the origin on the horizontal axis, y is the distance from the origin on the vertical axis, and σ is the standard deviation of the Gaussian distribution. It is important to note that the origin of these axes is at the center (0, 0). When applied in two dimensions, this formula produces a surface whose contours are concentric circles with a Gaussian distribution from the center point [20].

In our configuration (13, 13), specifies the size of the Gaussian kernel. The numbers (13, 13) indicate that the kernel will have a size of 13x13 pixels. 10 is the standard deviation of the Gaussian kernel.

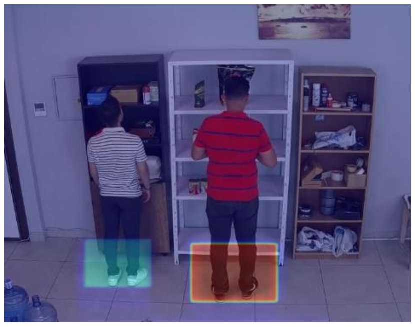


FIGURE 4. Demo test environment.

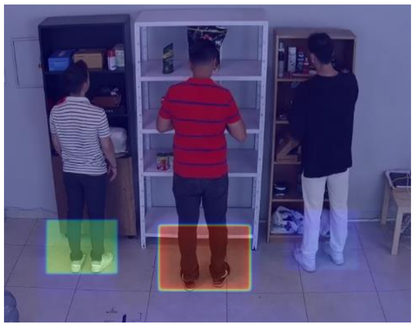


FIGURE 5. Demo test environment.

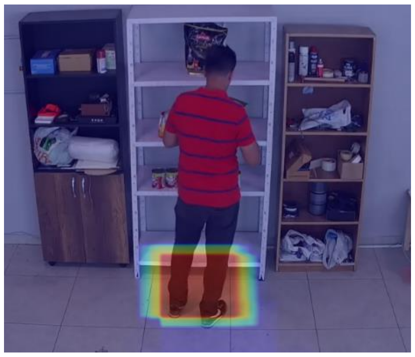


FIGURE 6. Demo test environment.

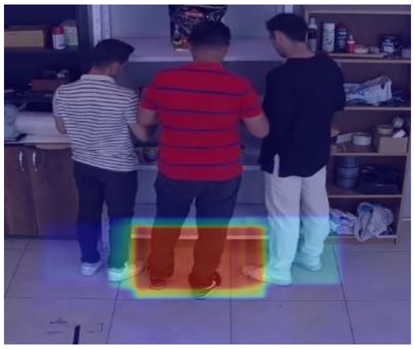


FIGURE 7. Demo test environment.

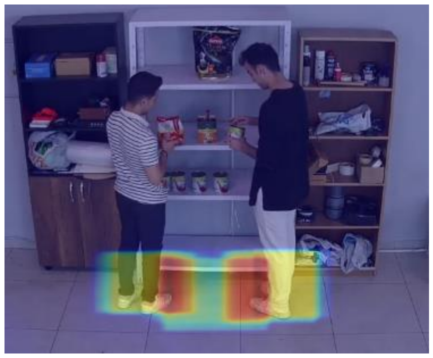


FIGURE 8. Demo test environment.

The processes outlined above have been developed in our demo test environment through the optimization of hyperparameters and the application of enhanced techniques. These advancements reflect a comprehensive approach to refining the methodologies, ensuring both the efficiency and efficacy of the results.

3. Experimental Results

In this section, we delve into the experimental results, highlighting the comprehensive examination of our study through rigorous empirical investigations. The experimental endeavors undertaken in this research are pivotal in unraveling the intricacies of the proposed methodology. We meticulously employed a set of well-defined performance parameters to assess the efficacy and robustness of our approach. These carefully selected metrics not only serve as benchmarks for evaluating the experimental outcomes but also contribute to the elucidation of the underlying dynamics of the phenomena under investigation. Through a systematic and methodical analysis of these performance parameters, we aim to provide a nuanced understanding of the outcomes derived from our experimental trials, thereby contributing to the advancement of knowledge in our field of inquiry.

$$\text{Precision} = TP / (TP + FP) \tag{8}$$

TP (true positive): the number of samples that actually belong to that class and are correctly predicted as that class by the model.

FP (false positive): The number of samples that do not belong to this class but are incorrectly predicted by the model as belonging to this class.

Precision: It is the percentage of correctly predicted positive samples out of the total positive predicted samples. In other words, it shows how accurate the model is when predicting whether an object belongs to a certain class.

$$\text{Recall} = TP / (TP + FN) \tag{9}$$

FN (false negatives): The number of samples that actually belong to that class but are incorrectly predicted by the model as another class.

Recall: It is the total sample replacement rate of correctly predicted positive samples belonging to that classified class. In other words, it shows the performance of the model in detecting a particular class.

$$\text{F1-score} = 2\ (\text{Precision x Recall}) / (\text{Precision} + \text{Recall}) \tag{10}$$

Harmonics of precision and Recall values are determined. So, there is a balanced distribution between precision and recall. While it is desired to increase both values, sometimes increasing one causes the other to decrease. F1-score combines these two metrics to provide an overall performance specification [21].

$$AP = (x + a)^n = \sum_{k=0}^{k=n-1} \binom{n}{k} x^k a^{n-k} \tag{11}$$

AP: average precision. Represents the area under the precision-recall curve for a class.

Recall(k): k. It is the sensitivity (recall) value calculated for the threshold value.

Recall(k+1): k+1. It is the sensitivity (recall) value calculated for the threshold value.

Precision(k): k. It is the precision value calculated for the threshold value.

[Recall(k)−Recall(k+1)]: This expression represents the difference between two consecutive sensitivity values. This represents a horizontal slice on the precision-recall curve [22].

The formula creates a precision-recall curve with sensitivity and sensitivity values calculated against different threshold values. The AP value is equal to the size of the area under this curve. This formula sums the horizontal slices for each of the threshold values specified to calculate this area [23]. In this article, we discussed heatmap visualization to better understand and analyze customer behavior and movements in retail stores. In this context, while examining the techniques we developed by customizing the YOLOv7 model, we also compared this model with itself and its upper version, YOLOv8, available in the literature. Before we get into performance comparisons, it's important to make a more fundamental comparison between YOLOv7 and YOLOv8. YOLOv7 and YOLOv8 are among the advanced object detection models of the YOLO series. It has an architecture based on YOLOv7, YOLOv4, YOLO-R and Scaled YOLOv4 and increases network efficiency by using Extended Efficient Layer Aggregation Network (E-ELAN) [24]. This model improves the accuracy and results of the model by considering various factors such as the number of layers, input image size, and channel width with compound scaling methods. On the other hand, YOLOv8 offers a more complex structure with the combination of FPN and PAN modules and detects objects of different sizes and shapes more effectively using advanced techniques such as Soft-

NMS [25]. In terms of performance, both models offer high accuracy, but YOLOv8 achieves a slightly higher mAP score on the COCO dataset and shows a slight reduction in the number of parameters despite more FLOPs, offering more complex calculations. These features highlight the key differences between the simplicity and efficiency of YOLOv7 and the advanced architecture and accuracy of YOLOv8 [26]. We chose the YOLOv7 model to perform fast and efficient analysis, especially in in-store systems. The size of the model and its more compact structure make it an ideal option for such applications. In addition, thanks to its compatibility, it provides the advantage of easily integrating old systems and applications into the new model. As a result of the bounding box techniques and tests we use, our approaches to hyperparameters increase the desired performances and show that our system is more effective in this area. Performance metrics on our system are calculated using an NVIDIA GeForce GTX 1650 TI, an Intel (R) CoreTM i5-10300H CPU at 2.50 GHz, and 8 GB of RAM.

TABLE 1. Evaluation of the system with image input size of 640.

PERFORMANCE METRİC	**YOLOv7_tiny**	**YOLOv8_s**	**Our system**
Precision	0.9000	0.9722	0.9250
Recall	0.8372	0.8140	0.8605
F1 Score	0.8675	0.8861	0.8916
AP	0.7535	0.7913	0.7959

As a result of our different approaches from the literature, we can see that our system stands out as an option that offers balanced performance and gives the highest result in terms of F1 score. Although the Yolov8 model has a higher ability to detect objects accurately, our system can help achieve better results in crowded and noisy environments by balancing its abilities to accurately detect and capture objects.

4. CONCLUSION

Our approach creates a heatmap visualization driven by a YOLO object detection model. Before developing this approach, we reviewed similar approaches available in the literature and adjusted our parameters and techniques to further improve the analysis of detected objects and tracked images. The main challenge we faced was analyzing hot spots in crowded or small rooms. We have identified a potential problem in these cases where the customer is closer to the security cameras due to the illusion of perspective blurring a key point rather than just showing the customer's point of interest. Our experiments continued with the aim of more effectively detecting and tracking objects in our retail analysis. Through our approach, we determined that displaying the bounding box to customers at the base of their feet was an ideal solution. This solution allowed people to better observe the points in front of the shelves and reduced complexity. The images [Figure 9–Figure 11] used in the data collection phase of this article were sourced from Pixabay. Pixabay is a platform offering copyright-free images and videos, allowing for the free distribution of downloaded photos and graphics. The particular images are licensed under the Creative Commons CC0 license, indicating that they are freely available for public and non-commercial use.

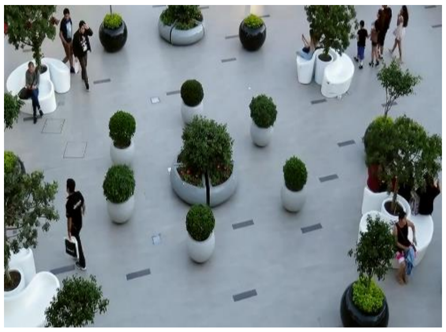


FIGURE 9. Heatmap visualization with Customized Yolo-Deep SORT System.

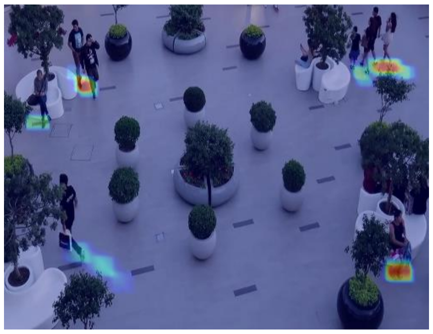


FIGURE 10. Heatmap visualization with Customized Yolo-Deep SORT System.

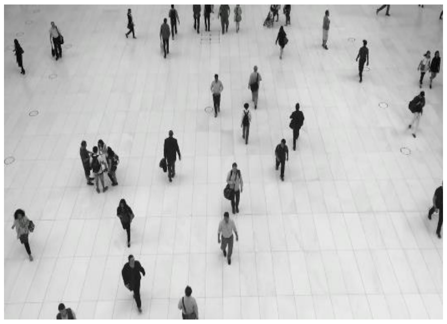


FIGURE 11. Data collecting.

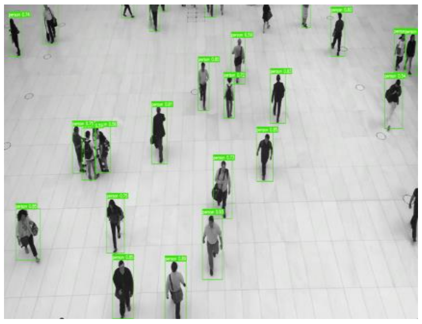


FIGURE 12. Use and development of detection algorithm.

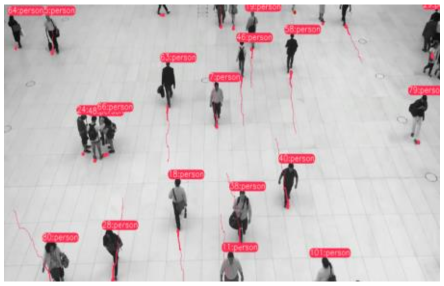


FIGURE 13. Use and development of tracking algorithm.

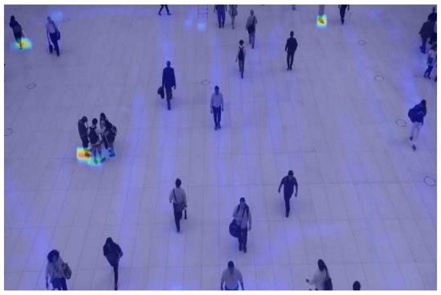


FIGURE 14. Heatmap visualization with Customized Yolo-Deep SORT System.

Author Contribution Statements Authors are equally contributed to the paper. All authors read and approved the final copy of the manuscript.

Declaration of Competing Interests The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement This research did not receive any specific grant from funding agencies in public or not-for-profit sectors.

References

[1] Liu, M., Lee, J., Kang, J., Liu, S., What we can learn from the data: a multiple-case study examining behavior patterns by students with different characteristics in using a serious game, *Tech. Knowl. Learn*., 21, (2016), 33-57, https://dx.doi.org/10.1007/s10758-015-9263-7.

[2] Fernandez, N., Gundersen, G., Rahman, A., Grimes, M., Rikova, K., Hornbeck, P., Ma'ayan, A., Clustergrammer, A web-based heatmap visualization and analysis tool for high-dimensional biological data, *Sci. Data*, 4, (2017), 170151, https://dx.doi.org/10.1038/sdata.2017.151.

[3] Gu, Z., Complex heatmap visualization, *iMeta*, 1 (3), (2022), https://doi.org/10.1002/imt2.43.

[4] Deng, W., Wang, Y., Liu, Z., Cheng, H., Xue, Y., Hemi: a toolkit for illustrating heatmaps, *PLoS ONE*, 9 (11), (2014), https://doi.org/10.1371/journal.pone.0111988.

[5] Mondal, S., Das, S., Musunuru, K., Dash, M., Study on the factors affecting customer purchase activity in retail stores by confirmatory factor analysis, *ESPACIOS*, 38 (61), (2018), 30.

[6] Girgensohn, A., Shipman, F., Wilcox, L. D., Determining activity patterns in retail spaces through video analysis, *Proc. ACM Conf. Multimedia*, (2008), 889-892, https://doi.org/10.1145/1459359.1459514.

[7] Oliveira, K., RetailNet: A Deep Learning Approach for People Counting and Hot Spots Detection in Retail Stores, Rio de Janeiro, Brazil, 2019.

[8] Onıga, F., Bacea, D., Single stage architecture for improved accuracy real-time object detection on mobile devices, *Img. Vis. Comput.*, 130 (9), (2023), 104613, https://doi.org/10.1016/j.imavis.2022.104613.

[9] Diwan, T., Anirudh, G., Tembhurne, J. V., Object detection using YOLO: challenges, architectural successors, datasets and applications, *Multimed. Tools Appl.*, 82 (6), (2023), 9243-9275, https://doi.org/10.1007/s11042-022-13644-y.

[10] Lakshmi Rishika, A., Aishwarya, Ch., Sahithi, A., Premchender, M., Real-time vehicle detection and tracking using yolo-based deep sort model: a computer vision application for traffic surveillance, *Turkish J. Comp. Math. Edu*., 14 (1), (2023), 255-264, https://doi.org/10.17762/turcomat.v14i1.13530.

[11] Aich, S., Stavness, I., Improving Object Counting with Heatmap Regulation, (2018), https://doi.org/10.48550/arXiv.1803.05494.

[12] Ilikci, B., Chen, L., Cho, H., Liu, O., Heat-map based emotion and face recognition from thermal images, *Comput. Commun. IoT Appl.*, (2019), 449-453.

[13] Bulat, A., Tzimiropoulos, G., Human Pose Estimation via Convolutional Part Heatmap Regression, Amsterdam, Netherlands, (2016).

[14] Pharr, M., Humphreys, G., Bounding box, *Physically Based Rendering*, 3, (2017).

[15] Huang, Z., Li, W., Xia, X.-G., Tao, R., A general Gaussian heatmap label assignment for arbitrary-oriented object detection, *IEEE Transc. Img. Process*., (2022), https://doi.org/10.1109/TIP.2022.3148874.

[16] Salim, M. P., Ong, J. J., IS, E., Surhatono, D., Object detection for child Learning media, *Inter. Conf. Sci. Tech*. (ICST), 8, Yogyakarta, Indonesia, (2022), 1-6.

[17] He, Y., Zhu, C., Wang, J., Savvides, M., Zhang, X., Bounding box regression with uncertainty for accurate object detection, *Proc. IEEE/CVF Conf. Comp.Vision Pattern Recog*., (2019), 2888-2897, https://doi.org/10.48550/arXiv.1809.08545.

[18] Hosang, J., Benenson, R., Schiele, B., Learning non-maximum suppression, *Proc. IEEE Conf. Comp. Vision Pattern Recog*. (CVPR), (2017), 4507-4515, https://doi.org/10.48550/arXiv.1705.02950.

[19] Córdova-Esparza, M., Terven, J., A comprehensive review of yolo: from yolov1 to yolov8 and beyond, *Mach. Learn. Knowl. Extr*. 5, (2023), 1680-1716 https://doi.org/10.3390/make5040083.

[20] Chandel, R., Gupta, G., Image filtering algorithms and techniques: a review, *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, 3 (10), (2013).

[21] Hicks, S. A., Strümke, I., Thambawita, V., Hammou, M., Riegler, M. A., Halvorsen, P., Parasa, S., On evaluation metrics for medical applications of artificial intelligence, *Sci. Rep*., 12 (1), (2022), 5979, https://doi.org/10.1038/s41598-022-09954-8.

[22] Ajayi O. G. , Ashi J., Guda B., Performance evaluation of YOLO v5 model for automatic crop and weed classification on UAV images, *Smart Agricult. Tech.*, 5, (2023), 100231.

[23] Atik, M. E., Duran, Z, Ozgunluk, R., Comparison of YOLO versions for object detection from aerial images, *Int. J. Environ. Geoinform.*, 9 (2), (2022), 87-93, https://doi.org/10.30897/ijegeo.1010741.

[24] Karadağ, B., Arı, A., Akıllı mobil cihazlarda YOLOv7 modeli ile nesne tespiti, *Politeknik J*., 26 (3), (2023), 1207-1214, https://doi.org/10.2339/politeknik.1296541.

[25] Özel, M. A., Baysal, S. S., Şahin, M., Derin öğrenme algoritması (YOLO) ile dinamik test süresince süspansiyon parçalarında çatlak tespiti, *Eur. J. Sci. Technol.*, (26), (2021), 1-5, https://doi.org/10.31590/ejosat.952798.

[26] Bayram, A. F., Nabiyev, V., Derin öğrenme tabanlı saklanan kamufle tankların tespiti: son teknoloji YOLO ağlarının karşılaştırmalı analizi, *Gümüşhane Univ. J. Nat. Appl. Sci.*, 13 (4), (2023), 1082-1093, https://doi.org/10.17714/gumusfenbil.1271208.

INSTRUCTIONS TO CONTRIBUTORS

Communications Faculty of Sciences University of Ankara Series A2-A3: Physical Sciences and Engineering is a single-blind peer reviewed open access journal which has been published since 1948 by Ankara University, accepts original research articles written in English in the fields of Physics, Engineering Physics, Electronics/Computer Engineering, Astronomy and Geophysics. Review articles written by eminent scientists can also be invited by the Editor.

Article-processing charges: The publication costs for Communications Faculty of Sciences University of Ankara Series A2-A3: Physical Sciences and Engineering are covered by the journal, so authors do not need to pay an article-processing and submission charges. The PDF copies of accepted papers are free of charges and can be downloaded from the webside. Hard copies of the paper, if required, are due to be charged for the amount of which is determined by the administration each year.

Submission: All manuscripts should be submitted via our online submission: https://dergipark.org.tr/en/journal/2457/submission/step/manuscript/new Note that only two submissions per author per year will be considered. Once a paper is submitted to our journal, all co-authors need to wait 6 months from the submission date before submitting another paper.

Cover Letter: Manuscripts should be submitted in the PDF form used in the peer-review process together with THE COVER LETTER and the source file (Supporting File). In the cover letter the authors should suggest the most appropriate Area Editor for the manuscript and potential four reviewers with full names, universities and institutional email addresses. Proposed reviewers must be experienced researchers in your area of research and at least two of them should be from different countries. In addition, proposed reviewers must not be co-auhors, advisors, students, etc. of the authors. In the cover letter, the author may enter the name of anyone who he/she would prefer not to review the manuscript, with detailed explanation of the reason. Note that the editorial office may not use these nominations, but this may help to speed up the selection of appropriate reviewers.

Preparing your manuscript: Manuscripts should be typeset using as DOC or LaTex. Authors will submit their manuscript and the cover letter via our submission system. A template of manuscript can be reviewed in https://dergipark.org.tr/tr/download/journal-file/20554 (or can be reviewed in pdf form).

Title Page: The title page should contain the title of the paper, full names of the authors, affiliations addresses and e-mail addresses of all authors. Authors are also required to submit their Open Researcher and Contributor ID (ORCID)'s which can be obtained from http://orcid.org as their URL address in the format http://orcid.org/xxxx-xxxx-xxxx-xxxx. Please indicate the corresponding author.

Abstract and Keywords: The abstract should state briefly the purpose of the research. The length of the Abstract should be between 50 to 5000 characters. At least 3 keywords are required.

Math Formulae: Formulas should be numbered consecutively in the parentheses ( ).

Tables: All tables must have numbers (TABLE 1) consecutively in accordance with their appearance in the text and a legend above the table. Please submit tables as editable text not as images.

Figures: All figures must have numbers (FIGURE 1) consecutively in accordance with their appearance in the text and a caption (not on the figure itself) below the figure. Please submit figures as EPS, PDF, TIFF or JPEG format.

Authors Contribution Statement, Declaration of Competing Interests and Acknowledgements should be given at the end of the article before the references.

References: The following format for the references should be used. Authors are urged to use the Communication.csl style (https://dergipark.org.tr/en/download/journal-file/18514) in Mendeley Desktop or Zotero automated bibliography. If manual entry is prefered for bibliography, then all citations must be listed in the references part and vice versa. Below, It has no relationship with the text, but can be used to show sample citations such as; for articles [1, 4], for books/booklets/theses [3], and for proceedings/conferences etc. [2].

[1] Demirci, E., Unal, A., Özalp, N., A fractional order SEIR model with density dependent death rate, Hacettepe J. Math. Stat., 40 (2) (2011), 287–295.

[2] Gairola, A. R., Deepmala, Mishra, L. N., Rate of approximation by finite iterates of q-Durrmeyer operators, Proc. Natl. Acad. Sci. India Sect. A Phys. Sci., 86 (2) (2016), 229–234.

[3] Lehmann, E. L., Casella, G., Theory of Point Estimation, Springer, New York, 2003.

[4] Özalp, N., Demirci, E., A fractional order SEIR model with vertical transmission, Math. Comput. Model., 54 (1-2) (2011), 1–6, https://dx.doi.org/10.1016/j.mcm.2010.12.051.

Peer-review policy: The Editor may seek the advice of two, or three referees, depending on the response of the referees, chosen in consultation with appropriate members of the Editorial Board, from among experts in the field of specialization of the paper. The reviewing process is conducted in strict confidence and the identity of a referee is not disclosed to the authors at any point since we use a single-blind peer review process.

Copyright: Copyright on any open access article in Communications Faculty of Sciences University of Ankara Series A2-A3: Physical Sciences and Engineering is licensed under a Creative Commons Attribution 4.0 International License (CC BY).

Declarations/Ethics With the submission of the manuscript authors declare that:

- All authors of the submitted research paper have directly participated in the planning, execution, or analysis of study;
- All authors of the paper have read and approved the final version submitted;
- The contents of the manuscript have not been submitted, copyrighted or published elsewhere and the visual-graphical materials such as photograph, drawing, picture, and document within the article do not have any copyright issue;
- The contents of the manuscript will not be copyrighted, submitted, or published elsewhere, while acceptance by the Journal is under consideration.
- The article is clean in terms of 'plagiarism', and the legal and ethical responsibility of the article belong to the author(s). Author(s) also accept that the manuscript may go through plagiarism check using IThenticate software;
- The objectivity and transparency in research, and the principles of ethical and professional conduct have been followed. Authors have also declared that they have no potential conflict of interest (financial or non-financial), and their research does not involve any human participants and/or animals.

Archiving: Research papers published in Communications Faculty of Sciences University of Ankara are archived in the Library of Ankara University (Volume 1-63) and Dergipark immediately following publication with no embargo.

Editor in Chief
Commun. Fac. Sci. Univ. Ank. Ser. A2-A3.
Ankara University, Faculty of Sciences
06100 Beşevler, ANKARA – TÜRKİYE

C O M M U N I C A T I O N S

FACULTY OF SCIENCES
UNIVERSITY OF ANKARA

DE LA FACULTE DES SCIENCES
DE L'UNIVERSITE D'ANKARA

Series A2-A3: Physical Sciences and Engineering

Volume: 66 **Number: 1** **Year: 2024**

Research Articles