2022 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)
Amity University, Noida, India. Oct 13-14, 2022

2022 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO) | 978-1-6654-7433-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICRITO56286.2022.9965111

Classification and Prediction of License Plates Using Deeply Learned Convolutional Neural Networks

Manik Rakhra
School of Computer Science and Engineering,
Lovely Professional University
Phagwara, Punjab, India Email-rakhramanik786@gmail.com

Dalwinder Singh
School of Computer Science and Engineering,
Lovely Professional University
Phagwara, Punjab, India
Dalwinder.singh@lpu.co.in

Arun Singh
Student, Integrated B. Tech M. Tech, Computer Science and Engineering,
Lovely Professional University
Phagwara, India
arunmandiarun2001@gmail.com

Ahmed Al Ahdal
School of Computer Science and Engineering,
Lovely Professional University
Phagwara, Punjab, India
Ahmed.alahdal21@gmail.com

Deepa Gupta
AIIT,
Amity University
Uttar Pradesh Noida India
dgupta@amity.edu

***Abstract*—Classifying and detecting of vehicle license plates using image processing techniques is an exciting research topic in IoT technology and the Internet of Things. Recognizing a vehicle's plate number is necessary since the number of vehicles on the streets is increasing and human capacity to perform this activity is limited. so it is simple for humans to read and recognize license plates that belonging to which country or state, but would the machines do the same when detecting or identifying plate numbers that correspond to different country or state?in this paper will discuss and implement a model that can read any License plates belong to any American states using Deep Learning Model, and (CNN) convolutional neural network in this paper we used DenseNet201 applied to Us license plate dataset which available on Kaggle scored 90.38%, then we used InceptionResNetV2 from keras for the model and it score 89% accuracy, EfficientNetB5 and score 91.5% accuracy, and Xception it scored 81%accuracy.**

Keywords— Deep Learning, Artificial Intelligence, license plates, model, keras.

I. INTRODUCTION

The automobile's licence plate, commonly known as a number plate (British English), licence plate (English American), is a metal or plastic plate fitted to a car or truck for official identification. Registration plates are required in all regions for road vehicles such as automobiles, lorries, and motorbikes. The need for other vehicles, such as bicycles, boats, or tractors, varies by municipality. The registration ID is a digital or alphabetical digital ID that is individually generated within the vehicle register in the release area by the vehicle or the owner of the vehicle. In certain places, the ID is unique throughout the country, and in others, it is unique inside the state or province.

The initial processing stage of a License Plate Recognition system is the identification and extraction of the licence plate sector from a larger image in order to decrease subsequent computations and algorithm complexity. There are various approaches to this problem, each with a distinct computational cost and success rate.

License Plate Detection: This method analyses an input picture and generates possible licence plate bounding boxes. Despite the fact that several plate identification algorithms are being suggested in recent years, it remains a difficult challenge to recognise licence plates correctly in the wild from an alternate capturing angle, with partial occlusion, or with multiple occurrences.

Fig. 1. Standard Michigan plate, using the term "SAMPLE"

Edge-based approaches seek for licence plates in areas where the edge density is higher than the rest of the image. Given that the brightness shift is more obvious and frequent in the licence plate region than elsewhere.

Color-based techniques are based on the observation that the licence plate colour differs from the colour of the vehicle's body. The HSI colour model [1-3] was used to identify probable licence plate segments, which were then validated using a position histogram. Jia and his coworkers First, using the mean-shift method, divide the picture into parts based on distinct colours.

Texture-based techniques: Use an unique pixel intensity pattern in plate regions to try to recognise licence plates. Zhang et al. [4] propose a method for identifying licence plates that incorporates both global statistical characteristics and local Haar-like features. Global feature classifiers eliminate more than 70% of the background region, but local Haar-like feature classifiers are resistant to licence plate brightness, colour, size, and placement.

Character-based techniques: Consider the licence plate to be a string of characters and search for the existence of characters in the image to detect it. Lin et al. [5-7] suggest that image saliency be used to detect licence plates. This approach first employs an intensity saliency map to segment out high-recall characters in an image, then applies a sliding window to these characters to calculate different saliency-related variables and recognise licence plates. prior to advancing to the next set of panels addressing the progress standards Automatic License Plate Recognizing (ALPR) systems handle vehicle identification for a variety of traffic-related applications, including the detection of stolen automobiles, traffic management, and public parking entry validation. Automatic detection systems have been vastly improved by recent developments in Parallel Processing and Deep Learning, in particular with regards to Objects Detection, Recognition, and Optical Character Recognition (OCR). In the real world, licence plate recognition is where deep Convolutional Neural Networks (CNNs) shine (LPs). Our methodology is based on the characteristics of the characters. CNN is used to separate characters from a busy backdrop. The CNN model's outstanding classification capacity assures character detection performance. The CNN classifier has been used to detect licence plates on typical text in the images, significantly reducing strong false positives. Character recognition on licence plates is a type of image classification task in which the segmental text must be divided into 36 segments. Template matching and learning-based techniques are examples of existing algorithms. The remining parts of this paper second section is literature view, third section is methodology, forth section is result analysis and finally with the conclusion.

II. LITRITURE VIEW

Machine learning is strongly aligned to computational statistics, that are using mathematical optimization to offer methodologies, theories, & specific applications help address real-world medical, industrial, social, and business issues. It is broadly classified into two categories: There are two kinds of learning: supervised learning and unsupervised learning. The method in supervised learning develops a mathematical model from a dataset which contains both the inputs and the expected outputs. Unsupervised learning is a technique that creates a mathematical model from a set of data that contains only inputs and no output labels. Deep learning is a type of machine learning and artificial intelligence (AI) that simulates the way of humans learn particular aspects of knowledge. Deep learning techniques, unlike traditional machine learning algorithms, are layered in a hierarchy of increasing complexity and generalisation.

The following is the existence studies have been conducted Shraddha S. Ghadge et al, [8], the system uses image processing and deep learning methods to distinguish an unauthenticated vehicle. The LAMP server stores a record of an authenticated vehicle as well as the owner information in residential zones. When the vehicle approaches the parking system, an ultrasonic sensor identifies the vehicle's distance and existence. The camera will then initialise and begin capturing video frames.

Cheng-Hung Lin et al,[9]. introduce a different two-stage deep learning-based technique that detects all licence plates in an image and extracts licence plate pictures before performing character recognition employing Convolutional Neural Networks. This primary step of the licence plate detection model captures the location of the licence plate on the screen as the region of interest (ROI). This ROI picture is again passed to a character recognition model to recognise licence plate characters. To decrease complexity, it employs the YOLOv2 object detection framework for directly conduct character identification, as well as overexposed, underexposed, blurred, and skewed licence plate images as training samples.

Indian vehicle number plates are the focus of the work of Ravi Kiran Varma Pa et al [10-13], which is devoted to their detection and recognition. Taking into account challenging conditions including illumination, distorted, skewed, and noisy photos, the study contributes significantly. Numerous image processing techniques, such as morphological modification and Gaussian smoothing, were used in the pre-processing phase. Following the application of border contours for licence plate segmentation, the subsequent contours are filtered depending on character dimensions and geographic localization. Finally, the K- closest neighbour algorithm is used for character identification after the region of interest has been filtered and de-skewed. To further this research, a Convolutional Neural Network will be used to combine the detection and recognition processes. When a substantial amount of data is available, CNNs have been demonstrated to work well with photos.

In this study the dataset available in Kaggle only three work have been done to this dataset till the day of writing this paper. The following table 1: showing the Results that have been achieved by other authors and it's available on Kaggle their accuracy and techniques that been used.

TABLE I. COMPRESSION RELATED WORK

Author	Model	Accuracy
GERRY[14]	EfficientNetB3	91%
KSR [15]	NFNet	0.9
STPETE_ISHII[16]	DenseNet201	0.12

In the table1: compressions of previous work that used deep learning model from TensorFlowand keras, EfficientNetB3 kares has achieved 91% which is good with loss 0.6 and by using NFTNet the author scored 0.9 accuracy but for DenseNet201 the performance is too low 0.12 this paper will work in this model using DenseNet201 to improve the accuracy and compare it to the previous result.

III. MRTHODOLOGY

In this study, we detect and classify licence plate numbers on US automobiles We used the object classification approach to determine the number from the captured image of the vehicle's US licence plate, detect the text from the image, and classify which state it belongs to. The image will be segmented, and in that segment, the section will be conducted for feature extraction and certain mathematical computation using Tensor Flow open source.[24,25]

Deep learning be used to improve the segmented image. CNN was trained to predict the specific figure and character in the image. And the predicted image is the feature extraction of the classification step and outcome.

The dataset available in Kaggle website which is contain of image The data comes from two sources: wikipedia and plateshack.com. Some plates date back to early 1900s. The dataset is relatively small and possesses intra-class variability,

There are Steps taken To fulfill the objective of this paper, the following steps used in this paper.

1, Download the data set “US License Plates” from Kaggle “ https://www.kaggle.com/ “

2. Load the data
3. Pre-process the data
4. Compile the model
5. Fit the model
6. Evaluate the model
7. Save the model.

A. Loading and visualization the data:

Dataset information: The data comes from two sources: Wikipedia and plateshack.com. Some plates date back to early 1900s.

1-1Attribute Information : the dataset contains:

Image (4462)

Labels (51)

['Kentucky', 'Wisconsin', 'Oklahoma', 'Alaska', 'Hawaii', 'Colorado', 'Utah', 'NorthCarolina', 'Ohio', 'NewYork', 'Virginia', 'Louisiana', 'Idaho', 'SouthCarolina', 'Arizona', 'NewMexico', 'Oregon', 'Indiana', 'Massachusetts', 'Washington', 'Montana', 'Minnesota', 'Maryland', 'NewJersey', 'Maine', 'Michigan', 'Mississippi', 'Kansas', 'California', 'RhodeIsland', 'Alabama', 'Tennessee', 'Missouri', 'NorthDakota', 'Georgia', 'NewHampshire', 'Nebraska', 'Connecticut', 'Wyoming', 'Florida', 'Illinois', 'Pennsylvania', 'Texas', 'Vermont', 'WashingtonDC', 'WestVirginia', 'Delaware', 'SouthDakota', 'Arkansas', 'Iowa', 'Nevada'] 51

1) Visualization

After we got to know data information with all labels Attribute, the following is figure 2: showing the different labels with the number of images in each label.

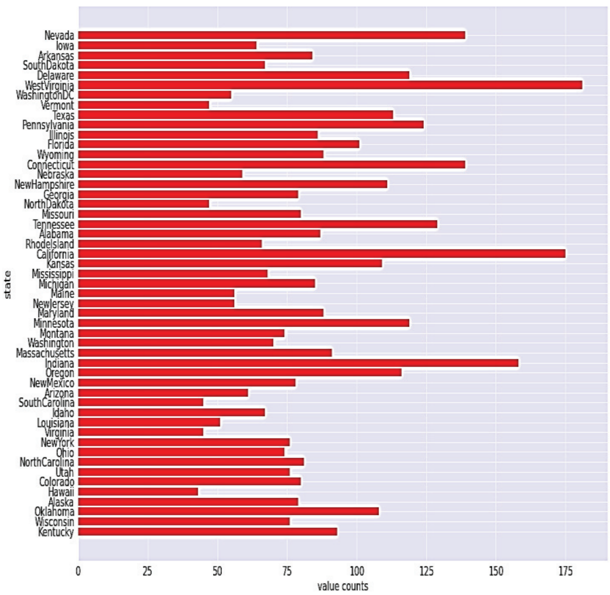

Fig. 2. data visualization

The below plot showing all state (Labels) and number of images available on each label, the following figure 3: is a sample of images available in the dataset and to which label it’s belongs.

Fig. 3. sample of images and labels

B. Pre-processing :

Which include Splitting the data into train, test, and validation :

```
# split df into train_df and test_df
    ‘dsplit=vsplit/(1-trsplit)
    strat=df['labels']
    train_df, dummy_df=train_test_split(df, train_size=trsplit
, shuffle=True, random_state=random_seed, stratify=strat)
    strat=dummy_df['labels']
    valid_df, test_df=train_test_split(dummy_df, train_size=d
split, shuffle=True, random_state=random_seed, stratify=st
rat)
    print('train_df length: ', len(train_df), ' test_df length: ',le
n(test_df), ' valid_df length: ', len(valid_df))
     # check that each dataframe has the same number of clas
ses to prevent model.fit errors
    trcount=len(train_df['labels'].unique())
    tecount=len(test_df['labels'].unique())
    vcount=len(valid_df['labels'].unique())
    print(list(train_df['labels'].value_counts()))
    return train_df, test_df, valid_df”
```

Sklearn's train test split function is used for model selection, and it divides data sets into two at random before ensuring that both dataframes have the same number of classes. discrepancies in the fit That's what I got out of it.

[145, 140, 126, 111, 111, 103, 99, 95, 95, 93, 90, 89, 87, 8

6, 81, 74, 73, 70, 70, 70, 69, 68, 67, 65, 64, 64, 63, 63, 62,

61, 61, 61, 59, 59, 56, 54, 54, 54, 53, 51, 49, 47, 45, 45, 4

4, 41, 38, 38, 36, 36, 35]

There are 51 unbalanced classes, therefore we build a function that takes a dataframe and two integer variables named max samples and min samples to bring everything into harmony. It utilises the trim function to limit the amount of samples in a string column's defined class to the specified value. If there are less than min samples observations for a given class, that class is dropped from further analysis. If certain classes don't have enough photos to reach max samples, enhanced versions of those images are generated and saved to the working dir.

To do this, we'll employ Augmentation methods, which modify or generate synthetic versions of existing data to boost the total quantity of available information. Used as a regularizer, it keeps machine learning models from becoming too specialised.

After augmenting, the photos are stored in a new directory under the working dir with the name aug df. As a result of combining the aug df with the train df, a balanced training set is produced with precisely max sample photos in each class.

Total Augmented images = 2601

[120, 120, 120, 120, 120, 120, 120, 120, 120, 120, 120, 120,

120, 120, 120, 120, 120, 120, 120, 120, 120, 120, 120, 120,

120, 120, 120, 120, 120, 120, 120, 120, 120, 120, 120, 120,

120, 120, 120, 120, 120, 120, 120, 120, 120, 120, 120, 120,

120, 120, 120]

The train_df dataframe is now balanced with 120 samples per class will create train, test and validation generators.

```
"trgen=ImageDataGenerator(preprocessing_function=scal
ar, horizontal_flip=True)
tvgen=ImageDataGenerator(preprocessing_function=scala
r)
train_gen=trgen.flow_from_dataframe( train_df, x_col='file
paths', y_col='labels', target_size=img_size, class_mode='c
ategorical',
                              color_mode='rgb', shuffle=True, batc
h_size=batch_size)
test_gen=tvgen.flow_from_dataframe( test_df, x_col='filepa
ths', y_col='labels', target_size=img_size, class_mode='cate
gorical',
                              color_mode='rgb', shuffle=False, bat
ch_size=test_batch_size)

valid_gen=tvgen.flow_from_dataframe( valid_df, x_col='fil
epaths', y_col='labels', target_size=img_size, class_mode='
categorical',
                              color_mode='rgb', shuffle=True, batc
h_size=batch_size)
classes=list(train_gen.class_indices.keys())
class_count=len(classes)
train_steps=int(np.ceil(len(train_gen.labels)/batch_size))"
```

the following figure 4: is represent the output of traininggenerator

Fig. 4. training generator

In the figure 4: it is showing that generator training extracting and classifying each plat licence belonging to which state.

C. *Create the model using* DenseNet201

The DenseNet201 CNN architecture is used in this work for feature extraction and classification. The figure below describes all of the phases of procedure, from input to output. Every layer uses the feature maps of the previous layers as input variables, as well as the generated extracted features are the inputs of the successive layers. Finally, a transition layer is applied to minimize the final output mappings. The DenseNet201 approach has several advantages, including one with a large reduction in the amount of parameters, improved performance due to reused features, and the elimination of vanishing gradient concerns[17].

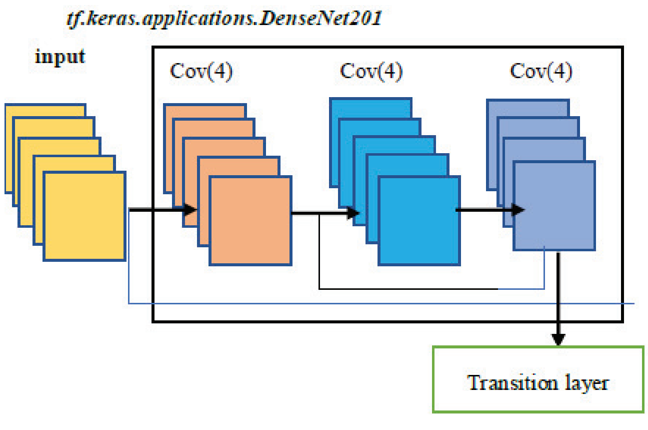

Fig. 5. DenseNet Architecture

DenseNets have several compelling features: it solve the vanishing-gradient problem, optimize component propagation, boost feature reuse, and greatly reduce the computational time. On the bulk of these, DenseNets outperform the state-of-the-art while using less computation to achieve great result.

```
“model_name='CA Project'
base_model=tf.keras.applications.DenseNet201(include_top
=False, weights="imagenet",input_shape=img_shape, pool
ing='max')
x=keras.layers.BatchNormalization(axis=-1, momentum=0.
99, epsilon=0.001 )(x)
x = Dense(256, kernel_regularizer = regularizers.l2(l = 0.0
16),activity_regularizer=regularizers.l1(0.006),
            bias_regularizer=regularizers.l1(0.006) ,activatio
n”
```

model name CA Project which will store in new diractry as dir_csv for further work then

Compile

Compile is in responsible of analyzing the loss function, optimizer, and metrics. This has nothing to do with weight training, and you may generate a model as many times as you like without affecting your pretrained weights.

Fit

Model fitting determines whether efficiently a learning algorithm generalises across the data that is identical to that one which had already trained. A well-fitted order model higher accurate results. An overfitted model closely fits the data. An underfitted model does not match closely sufficient.

The equations are an exception to the prescribed specifications of this template. You will need to determine whether or not your equation should be typed using either the Times New Roman or the Symbol font (please no other font). To create multileveled equations, it may be necessary to treat the equation as a graphic and insert it into the text after your paper is styled.

IV. Result Analysis

Training model have been trained for 10 epochs the following figure 6: showing the Accuracy, Loss, validate Accuracy and validate loss.

CA2 Project

Notebook Data Logs Comments (0) Settings

Epoch	Loss	Accuracy	V_loss	V_acc	
1 /10	7.007	23.578	5.92822	34.753	s: 7.00653
2 /10	4.474	54.886	3.92606	61.435	s: 4.47373
3 /10	3.036	77.598	2.87291	75.336	s: 3.03567
4 /10	2.075	91.062	2.20406	83.857	s: 2.07467
5 /10	1.474	97.010	1.77962	84.305	s: 1.47368
6 /10	1.095	99.020	1.45264	87.444	s: 1.09466
7 /10	0.865	99.526	1.30334	86.099	s: 0.86524
8 /10	0.727	99.771	1.22042	86.547	s: 0.72722
9 /10	0.661	99.624	1.22236	86.547	0.66084
10 /10	0.610	99.951	1.00431	91.256	s: 0.61040

Fig. 6. Accuracy, loos training set

Starting Epoch score high loss 7.007 and low accuracy count down for each epoch till last epoch with high accuracy which is 99.9% and low loos 0.6 Validation Accuracy is 91.2% which is good to the accuracy and validation loss is 1.00. The figure 7: below show the training with validation loss

Evaluate Model On The Test Set

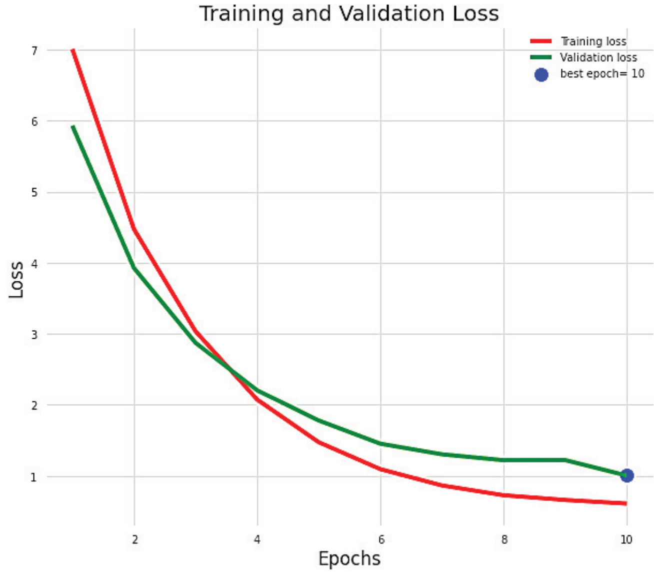

Fig. 7. training and validation loss

From figure 7: Loss Validation Training loss + validation loss through times is among the most commonly utilised measure combinations. The training loss shows how well the model fits the training data, whereas the validation loss represents how well the model fits new data. If the validation loss is bigger than the training loss, the model is overfitting.

Fig. 8. Training and validation Accuracy

Training and validation Accuracy usually refer to Validation accuracy is the accuracy determined on the data set that is not utilised for training and is used (usually during the training phase) as validating accuracy (or "evaluating")

Predicting

Now it's time for predicting and to know how our model will work and predict a single image the figure 9: below shows the probability of image matching the same image in the label

From the figure 9: the probability of classifying the image belong to the correct label is 98.75%.

And overall, the evaluated model on test set score accuracy on the test set is 90.83 % by using DenseNet201

The following table showing the different techniques used on this paper with their performance accuracy.

Fig. 9. Predictor image

TABLE II. ACCURACY COMPRESSION

DL Techniques	Accuracy
DenseNet201	90.83 %
InceptionResNetV2	89%
EfficientNetB5	91.5%
Xception	81%

V. RESULT ANALYSIS

Deep learning neural network model, that enables us to extract higher representations for picture content, (CNN) convolutional neural network, which is a sort of neural network with distinct layers. in this paper we used DenseNet201 using less parameters and can be deeper than the usual networks by 201 layers and easy to optimize. US license plate dataset scored in training accuracy 99.96% and in evaluating test score 90.38% using DenseNet201 from CNN this result had been improved from previous work, this prove that the machine have the ability to read, extract and predict as human with respect to the differences after training and evaluating, from this training model which used DenseNet201 the machine was able to match and predict and recognizing plates number belonging to which state in the united states America by 98.75% probability of matching, then we used InceptionResNetV2 from keras for the model and it score 89% accuracy, then EfficientNetB5 and score 91.5% accuracy, last Xception it scored 81%accuracy.

REFERENCES

[1] Robertson, Patrick (1974). The first writers. CN Potter: Crown Publishers Distribution. P. 51. Recovered August 13, 2011.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] Sushovan Chaudhury, Manik Rakhra, Naz Memon, Kartik Sau, Melkamu Teshome Ayana, "Breast Cancer Calcifications: Identification Using a Novel Segmentation Approach", Computational and Mathematical Methods in Medicine, vol. 2021, Article ID 9905808, 13 pages, 2021. https://doi.org/10.1155/2021/9905808.

[3] A. A. Ahdal, D. Prashar, M. Rakhra and A. Wadhawan, "Machine Learning-Based Heart Patient Scanning, Visualization, and Monitoring," 2021 International Conference on Computing Sciences (ICCS), 2021, pp. 212-215, doi: 10.1109/ICCS54944.2021.00049..

[4] Geschiedenis van de kentekenplaat" [license plate history]. Autogids (Dutch). Netherlands. Archived from the original on February 16, 2012..

[5] Wang Y. Lin W. Horng S.: 'A sliding window technique for efficient license plate localization based on discrete wavelet transform', Expert Syst. Appl., 2011, 38, (4), pp. 3142– 3146 CrossrefWeb of Science®Google Scholar.

[6] Manik Rakhra, Amitabh Bhargava, Deepshikha Bhargava, Ramandeep Singh, Astha Bhanot, Abdul Wahab Rahmani, "Implementing Machine Learning for Supply-Demand Shifts and Price Impacts in Farmer Market for Tool and Equipment Sharing", Journal of Food Quality, vol. 2022, Article ID 4496449, 19 pages, 2022. https://doi.org/10.1155/2022/4496449

[7] Manik Rakhra, Sumaya Sanober, Noorulhasan Naveed Quadri, Neha Verma, Samrat Ray, Evans Asenso, "Implementing Machine Learning for Smart Farming to Forecast Farmers' Interest in Hiring Equipment", Journal of Food Quality, vol. 2022, Article ID 4721547, 17 pages, 2022. https://doi.org/10.1155/2022/4721547.

[8] Wichai Puarungroj, Narong Boonsirisumpun, "Thai License Plate Recognition Based on Deep Learning", 3rd International Conference on Computer Science andComputationalIntelligence 2018.

[9] Cheng-Hung Lin and Yi-Sin Sie, "Two-Stage License Plate Recognition System Using Deep learning", National Taiwan Normal University, Taipei, Taiwan.

[10] Ravi Kiran Varma Pa, Srikanth Gantaa, Hari Krishna Bb, Praveen SVSRKc, "A Novel Method for Indian Vehicle Registration Number Plate Detection and Recognition using Image Processing Techniques" International Conference on Computational Intelligence and Data Science 2019.

[11] Manik Rakhra, Ramandeep Singh, Tarun Kumar Lohani, Mohammad Shabaz, "Metaheuristic and Machine Learning- Based Smart Engine for Renting and Sharing of Agriculture Equipment", Mathematical Problems in Engineering, vol. 2021, Article ID 5561065, 13 pages, 2021. https://doi.org/10.1155/2021/5561065

[12] Shruti, R. Soumya, M. Rakhra, N. S. Chintagunti, B. Singh and D. Singh, "Modern Data Mining Approach to Handle Multivariate Data and to Implement Best Saving Services for Potential Investor," 2022 International Mobile and Embedded Technology Conference (MECON), 2022, pp. 611-616, doi: 10.1109/MECON53876.2022.9752101.

[13] M. Rakhra et al., "An Analytical Study of the Types of Implements used by Farmers in Mechanized Agriculture," 2022 International Mobile and Embedded Technology Conference (MECON), 2022, pp. 683-687, doi: 10.1109/MECON53876.2022.9751983

[14] https://www.kaggle.com/koshirosato/us-license-plates- recognition-using-nfnet/notebook

[15] https://www.kaggle.com/gpiosenka/efficientnetb3-f1-score-91

[16] https://www.kaggle.com/koshirosato/us-license-plates- recognition-using-nfnet

[17] https://keras.io/api/applications/densenet/#densenet201- function