2021 Asian Conference on Innovation in Technology (ASIANCON)
Pune, India. Aug 28-29, 2021

SecurePark: Vehicle Intrusion Detection System

Deep U. Nayak, Atharva P. Mohite, Pranav P. Nair and Pramod J. Bide
Sardar Patel Institute of Technology
Mumbai, India
deep.nayak@spit.ac.in, atharva.mohite@spit.ac.in, pranav.nair@spit.ac.in, pramod_bide@spit.ac.in

***Abstract*—A vehicle intrusion detection system includes detection and identification of a vehicle's license plate. This paper describes our web application which uses YOLOv5 (you only look once) as its object detection model for detecting a vehicle's number plate and uses Microsoft Vision API for the OCR (optical character recognition) that identifies the characters from the cropped image of the license plate that is captured during the detection process and sent to the API. Initially different versions of YOLO were tested for object detection and based on the results achieved from the above model training process, YOLOv5 gave the most accurate results among the versions tested. For the identification of the characters on the plate, along with the Microsoft Vision API, the Tesseract OCR engine was used but did not produce satisfactory results as compared to the API. Our main motive was to build an end-to-end integrated web application that makes use of our machine learning model to perform detections using the power of a local computer so that it can be used by any person irrespective of their knowledge about the technologies working under the hood.**

***Index Terms*—YOLO, Object Detection, OCR, Deep Learning, Convolution Neural Network**

I. Introduction

Many residential society administrations, tolls, business complexes, and parking spaces in India lack an automated system for car parking and vehicle monitoring. Most places face an impending problem of illegal vehicle parking inside their premises. This issue is not bound to just parking, but also adds to the security concerns inside those establishments. Though there are solutions that exist in the market, affordability comes into question.

In the outbreak of a pandemic like the Covid-19, physical security personnel poses a major health concern, and hence housing societies and business complexes prefer to rely on automated solutions rather than putting the life of the security guard as well as the office workers at risk. One of the major security concerns of any establishment is the entry of unauthorized vehicles into the premises. Many solutions exist that solve this problem but as mentioned before feasibility and affordability come into question. This is exactly where our product steps in the game. Usually, any automatic vehicle identification system requires state-of-the-art G.P.U.s and cloud services which causes societies to stream their CCTV footage to a third party 24x7. However, our implementation focuses on a completely offline implementation where the processing of all of the CCTV footage happens in real-time and makes use of a simple laptop or computer. We plan to solve this problem of security and privacy through our product so that any common man can rest assured while using our application. The main aim of our application is to create a machine learning model that is trained to identify the vehicles entering the premises and extract their license plate number which is further used to classify whether the particular vehicle is authorized to enter the premises.[1] The model would be trained to work irrespective of the ongoing season causing problems with data retrieval and lighting conditions. The proposed application is an easily scalable and modular system with inbuilt alert functionality to notify users about such illegal detections and can be used by the majority of the Indian housing societies as well as business complexes. Lots of research has been done by previous researchers in the field of image processing to detect the car license plate using the grayscale method, localization, segmentation, and recognition[2].

The rest of the paper is organized in the following manner. In section II, we do an extensive literature survey where we reviewed various papers and methodologies used in those papers. In section III, we go through the process and workflow of the project, data collection, training of the machine learning model and the selection of the OCR engine to be used. In section IV, we go through the web application created to incorporate the model. Lastly, section V concludes the paper and addresses the possible future scope.

II. Literature Survey

P. Kulkarni et al propose insights on ANPR technology in their paper and how ANPR algorithms are used for vehicle surveillance. ANPR algorithms are generally divided into four steps: (1) Vehicle image capture (2) Number plate detection (3) Character segmentation and (4) Character recognition. The paper [3] was a cumulative survey of several papers and each data their datasets. Different results based on different sizes of the images and different types of methods used. Certain factors like different illumination conditions, vehicle shadow and non-uniform size of license plate characters, different font and background color affect the performance of ANPR.

2021 Asian Conference on Innovation in Technology (ASIANCON) | 978-1-7281-8402-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/ASIANCON51346.2021.9544923

In [4], the authors propose an intelligent approach of detecting vehicular number plates automatically using three efficient algorithms namely Ant colony optimization (ACO) used in plate localization for identifying the edges, a character segmentation, and extraction algorithm, and a hierarchical combined classification method based on inductive learning and SVM for individual character recognition. The dataset used as input consists of noisy, blurred, night, and daylight license plate images. Sobel edge mask and Connected Component Analysis (CCA) - the accuracy of 80%. SVM-based character recognition system - 79.84% success rate. Real-time ANPR system - 97.3% success rate

Atul Patel et al, in their paper, go over a system consisting of the integration of algorithms like: 'Feature-based number plate Localization' for locating the number plate, 'Image Scissoring' for character segmentation, and statistical feature extraction for character recognition; which is specifically designed for Indian number plates[5]. The system was tested with a set of images not used during testing, having wide variations in illumination conditions. In cases where the number plate script is non-English or the number plate is badly distorted are excluded then, 82% of the plates were recognized correctly. The performance of individual sections is 87% for number plate localization, 95% for character segmentation, and 85% for character recognition. There are certain restrictions on parameters like the speed of the vehicle, script on the number plate, skew in the image which can be aptly removed by enhancing the algorithms further.
Sarbjit Kaur, in the paper[6] goes over ANPR systems that consist of 4 phases: - Acquisition of Vehicle Image and Pre Processing, Extraction of Number Plate Area, Character Segmentation, and Character Recognition. A database consisting of five categories of images are Light Images, Dark Images, Low Contrast Images, Blurred Images, and Noisy Images. The existing success rate for number plate extraction, character segmentation, and character recognition is 90.83%, 88.33%, and 86.67% respectively while the proposed success rate is 97.5%, 97.50%, and 96.67% respectively. The existing ANPR method works well for dark and bright/light categories images but it does not work well for Low Contrast, Blurred, and Noisy images.

Paper [7] presents a robust and efficient ALPR system based on the YOLO object detector. The Convolutional Neural Networks (CNNs) are trained and fine-tuned for each ALPR stage so that they are robust under different conditions (e.g., variations in the camera, lighting, and background). Especially for character segmentation and recognition, they designed a two-stage approach employing simple data augmentation tricks such as inverted License Plates (LPs) and flipped characters. The first dataset, composed of 2,000 frames from 101 vehicle videos. The second dataset contains 150 videos and 4,500 frames captured when both camera and vehicles are moving and also contains different types of vehicles. SSIG dataset: Vehicle recognition: recall of 100% and precision above 99% LP Detection: recall and precision = 100% Character Recognition: Without temporal redundancy info: 81.8% With temporal redundancy info: 93.53% UFPR-ALPR Data set (FAST - YOLO) Vehicle recognition: recall of 97.33% LP Detection: recall 98.33% Character Recognition: Without temporal redundancy info: 64.89% With temporal redundancy info: 78.33% Analyzing the results they noticed that a substantial part of the errors was in motorcycle images, highlighting this constraint in both systems. This suggests that those systems are not so well trained for motorcycles.

The paper written by K. B. Sathya et al. [8] analyzes various Vehicle License Plate Recognition (VLPR) methods, classifies the VLPR literature based on feature extraction and classification schemes, presents the survey of approaches for detection, localization, and recognition of plate character and numbers, and addresses the challenges in performance improvement. A data set containing different types of pictures of vehicle license plates which includes photos taken in the rainy season shadowed regions, and focuses on images with pose variation. License Plate Recognition can be improved by preprocessing all the test images by using methods like Gaussian filtering and Mean Filtering followed by removal of raindrops by sparse coding and dictionary learning. Segmentation methods like boundary-based are used to detect the characters present on the license plate followed by an OCR step that converts it to readable text. Though this paper covers all the steps that are necessary to improve the accuracy of detecting license plates and extracting numbers from them, it does not outline any algorithms or methods to implement this ideology further. This could have been included in the paper so that all the methodologies discussed in the paper could be successfully realized.

Lastly, in their paper, Xie, Xuemei et al. [9] proposed a novel illegal vehicle parking detection system. Illegal vehicles captured by the camera are firstly located and classified by the famous Single Shot MultiBox Detector (SSD) algorithm. To improve the performance, it proposes to optimize SSD by adjusting the aspect ratio of the default box to accommodate their data set better. A custom data set was used which focused on cars, motorcycles, and truck detection under different circumstances like heavy traffic, rainy season, and sunny conditions. Single Shot MultiBox Detector Method (SSD) results in 99% accuracy and real-time (25FPS) detection with strong robustness in complex environments. Although this method improves on the shortcomings of YOLO and Faster RCNN and is also much better at identifying vehicles in rough weather conditions, this method might struggle in countries like India, since the model is trained to identify cars with more than 15s waiting times as illegally parked vehicles.

III. Object Detection Model

A. Initial Research

Our initial approach to the problem of detecting license plates from a given image or a frame of a video included YOLOv4[10] and the Tesseract OCR technology. We made the following observations after implementing our model in the above-mentioned technologies. While training our model using the YOLOv4 framework and using the Google Colab platform for the advantage of training on GPUs, it took 4-5 hours to train our model on a data set of 200 images. Apart from this, the detections performed by the model were not accurate in the majority of the scenarios and the time is taken to process a 2 to 3 seconds video on a local computer took around 30 minutes. This is the reason we move to YOLOv5 which is a PyTorch based implementation of the previous versions of YOLO and are known for their lightweight and accurate models.[11]

B. Data Preparation

Labeled images for training purposes were taken from an online resource. Additional images were added to the training data set to ensure that outliers are included in the data set and it doesn't hamper the performance of the model. For these additional images, we make use of the LabelIMG tool to label parts of the given image and save the coordinates of the object in question, which in this case is the license plate, in a suitable text file. After this, a round of augmentation of these images was performed which included rotating, tilting, blurring, and changing the contrast of a few images, and re-calculating the coordinates of the license plate. All of these images amounted to a dataset of 200 labeled images. Along with this, a validation set was created of 36 images. These validation images are used by YOLOv5 to calculate and monitor the accuracy of the model between different iteration of the training phase.

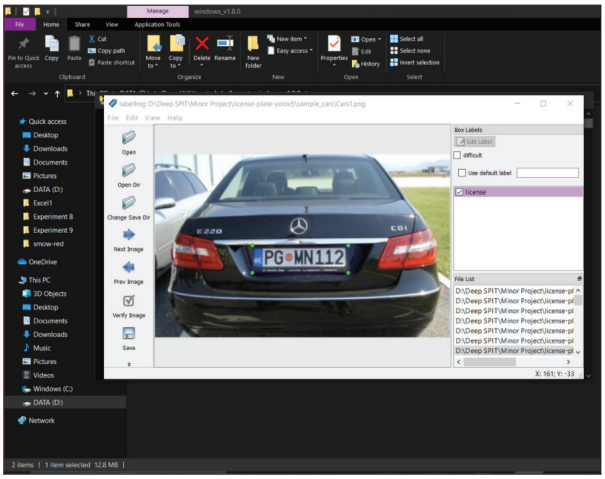

Fig. 1: Labelling images using the LabelIMG tool

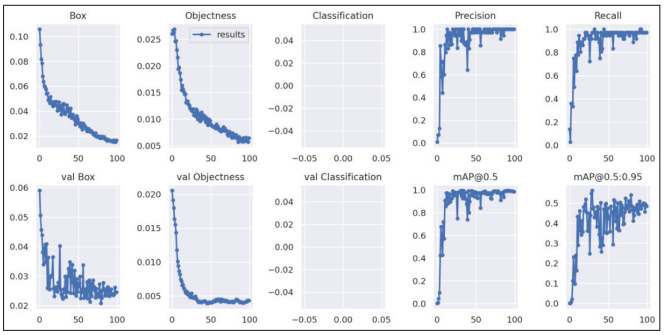

Fig. 2: Training Results - YOLOv5

C. Training with YOLOv5

To train our model using the YOLOv5 framework, we first configure the structure of the entire project such that YOLOv5 can find the appropriate "train" and "valid" folders[12] and accordingly access the files inside them. At the same time, we specify the training parameters that decide the nature of our training process. YOLOv5 offers 4 different types of models (pre-trained on the COCO data set) each differing from the other based on the size of the neural network.[13] After testing out all the different types of models we decide to go ahead with YOLOV5s which have the lowest training time and maintains a reasonable accuracy while performing detections. We train the model with the batch size set to 8 images and the epoch to 100. With the above configuration, the model is trained in 40 minutes with the highest accuracy.

Fig. 3: Image augmentation - YOLOv5

YOLOv5 itself performs image augmentations (in the form of Mosaic Dataloader first introduced in YOLOv4[14]) while training the model (Fig 4) and this results in an increase of the overall accuracy of the model. Training statistics show that the model was trained to output detection results with 0.97 accuracies. After testing the model in different scenarios

which included images with two license plates, images with very low or poor visibility of the license plates as well as those with rotated objects and overall contrast, we could conclude that the accuracy given by the training statistics is correct.

D. OCR

```
0  Orientation and script detection (OSD) only.
1  Automatic page segmentation with OSD.
2  Automatic page segmentation, but no OSD, or OCR.
3  Fully automatic page segmentation, but no OSD. (Default)
4  Assume a single column of text of variable sizes.
5  Assume a single uniform block of vertically aligned text.
6  Assume a single uniform block of text.
7  Treat the image as a single text line.
8  Treat the image as a single word.
9  Treat the image as a single word in a circle.
10 Treat the image as a single character.
11 Sparse text. Find as much text as possible in no particular order.
12 Sparse text with OSD.
13 Raw line. Treat the image as a single text line, bypassing hacks that are Tesseract-specific.
```

Fig. 4: Different page segmentation modes of Tesseract[15]

After extracting the coordinates of the license plates from the result given by our model, we perform optical character recognition on the cropped image to convert the license number present in the image to a machine-readable string which can then be saved in the database of the application for future use. Initially, we planned to use the Tesseract Engine[16] to perform OCR on the cropped images. Tesseract works offline and can be configured in different ways to detect different types of characters. However, even after testing out different modes as well as whitelisting alphanumeric characters, Tesseract was giving poor results for images where the license number was visible. This resulted in us moving over to Microsoft Vision API for OCR. Microsoft Vision API has to be configured to use in our project. This includes generating an API key and using special methods mentioned in the documentation of Vision API to upload an image and receive the string containing our required license number. This is followed by the processing of the string received from the API. This needs to be done because the string received includes special characters as well as punctuation marks that might be detected due to discrepancies in the license plate image. Apart from this, while processing a video, the same license plate image will be passed to Vision API and a different result will be received every time. To include only the correct detection in the final license plate list we convert all the characters to their ASCII equivalents and then check the length of the string received. After this, we check the percentage of numbers in the given string and if it is within the range of 10% of the desired value, we consider the string as a correct detection.

IV. SecurePark: Web App

The objective while creating the web app was to develop a fully automated end-to-end system that continuously monitors the CCTV footage of any business complex or housing society. The web app is an easily scalable and modular system that can be used by the majority of the Indian housing societies

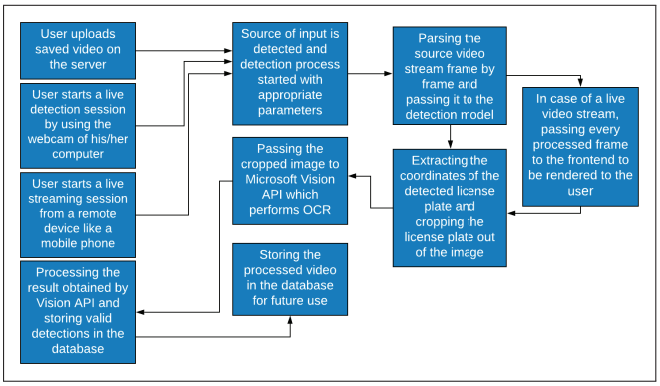

Fig. 5: Flowchart - Application Life Cycle

as well as business complexes. The web app incorporates a user-friendly and intuitive interface that is closely integrated with the machine learning model in the back end, which makes it easier for the user to make use of our application without worrying about the interior level of complexity. It includes an alert system that notifies the owner of the application in case of any intruder detection and keeps a log of the time of entry as well as the detected video. The web app curates the daily, weekly, and monthly data to give basic statistics for the given period so that it is easier for the user to get a top view of the data generated for his or her premises.

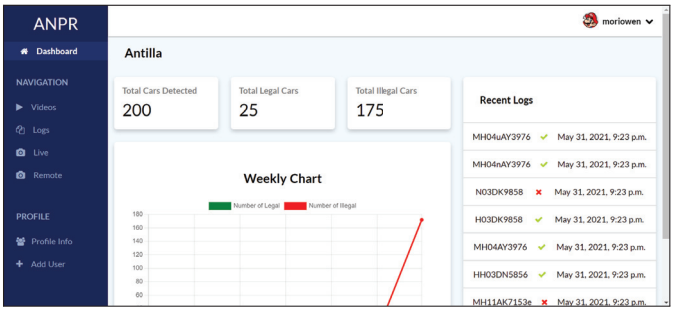

Fig. 6: Web App - Dashboard

The users can log into the web app by entering their credentials and they will be added to their respective societies. The user is prompted to give details like contact number, and car-specific information like the car model and the respective license plate number. Once the car is stored in the system, it won't be added to the illegal list and hence the other users won't be notified of the same. The user is given an option to opt for the alert systems that notify the users of the detection of illegal vehicles. When some illegal vehicle is detected in real-time with the help of the model, they are notified of the license plate number through messaging services like text, or WhatsApp, which can be verified by the users.

The users are also given the list of past recordings which they can go through if the need arises. The videos are displayed with the detection timestamps on the right side so that the user can navigate easily to their de-

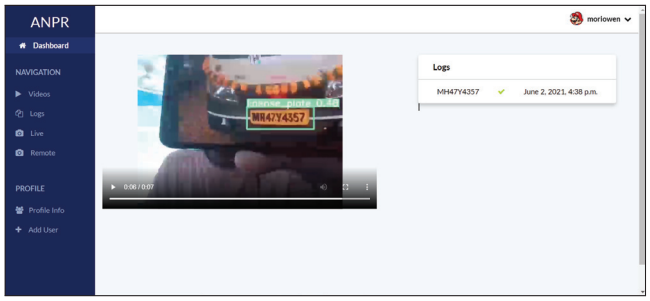

Fig. 7: Web App - Detections on images shown through the laptop webcam

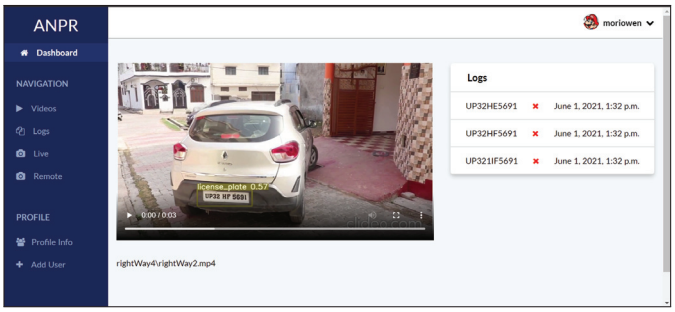

Fig. 8: Web App - Detections on uploaded video

sired location in the video. The user is shown daily, weekly, and monthly statistics of the number of detections, the time frame for the most detections throughout the day, through various charts and graphs in the dashboard. The user can also search through logs by searching for tags or applying filters to get the desired information.

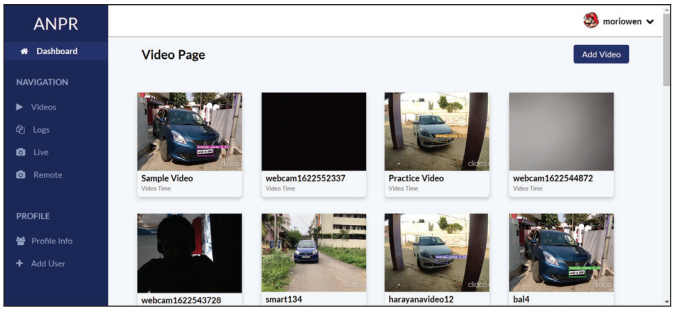

Fig. 9: Web App - Video Page

For the live detection feature, the web app would take in video footage from the CCTV camera that is pre-installed in the society premises without any extra overhead costs. For testing purposes, we have used an application named "IP Webcam" which allows remote access to the camera of the phone. The phone camera feed which acts as a proxy for a CCTV camera is then fed to the model, in real-time and thus one can see a live video with detections through the web app.

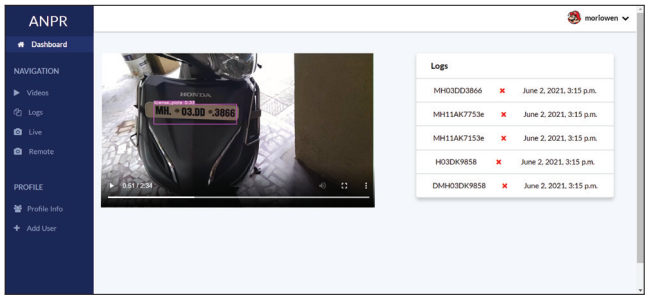

Fig. 10: Web App - Live detection using mobile camera on two wheelers

V. Conclusion

We were able to successfully implement an end-to-end automated system that constantly monitors the video footage provided by the user and also sends intruder alert messages to the application user whenever a suspicious vehicle is detected.

We made use of YOLOv5 as the base of our machine learning model and improved the implementation of YOLOv5 so that it suits our requirements and also provides extra features to the end-user. During our initial stage, we passed all the images and frames of a video directly to the model, but we realized that the final results can be made more accurate if we augmented the images that we received which include cropping, changing the color contrast, and fixing the orientation of the image so that the model gets a better test data to work upon. We also made use of Microsoft Vision API instead of the Tesseract system as Tesseract requires special machine learning training to perform OCR on custom objects and this was not in line with our objective of building a lightweight vehicle detection system.

We were also able to integrate the entire machine learning model with our front-end website which was a completely new experience for us. We made use of asynchronous function calls and multi-threading to improve the performance of our system and allow any user to perform object detection using local resources at the user's disposal.

References

[1] M. Y. Aalsalem, W. Z. Khan, and K. M. Dhabbah, "An automated vehicle parking monitoring and management system using anpr cameras," in *2015 17th International Conference on Advanced Communication Technology (ICACT)*, pp. 706–710, 2015.

[2] A. A. Shahraki, A. E. Ghahnavieh, and S. A. Mirmahdavi, "License plate extraction from still images," in *2013 4th International Conference on Intelligent Systems, Modelling and Simulation*, pp. 45–48, 2013.

[3] P. B. P. Kulkarni, A. Khatri and K. Shah, "Automatic number plate recognition (anpr) system for indian conditions," in *19th International Conference Radioelektronika, 2009*, pp. 111–114, 2009.

[4] S. S. . C. A. N. Sasi, A., "Automatic car number plate recognition," in *International Conference on Innovations*

in Information, Embedded and Communication Systems (ICIIECS), 2017.

[5] D. . P. A. Patel, Chirag Shah, "Automatic number plate recognition system (anpr): A survey," in *International Journal of Computer Applications (IJCA)*, 2013.

[6] S. Kaur, "An automatic number plate recognition system under image processing," in *International Journal of Computer Applications (IJCA)*, vol. 8, pp. 14–25, 2016.

[7] R. L. et al, "A robust real-time automatic license plate recognition based on the yolo detector," in *2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–10, 2018.

[8] V. V. K. B. Sathya and G. Kavitha, "Vehicle license plate recognition (vlpr)," in *2017 Trends in Industrial Measurement and Automation (TIMA)*, pp. 1–6, 2017.

[9] C. . C. S. . S. G. . Z. Z. Xie, Xuemei Wang, "Real-time illegal parking detection system based on deep learning," 2017.

[10] Ayoosh Kathuria, "What's new in yolo v3?." https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b, 2018. Accessed: 2021-06-20.

[11] Mihir Rajput, "Yolo v5 — explained and demystified." https://towardsai.net/p/computer-vision/yolo-v5%E2%80%8A-%E2%80%8Aexplained-and-demystified, 2015. Accessed: 2021-06-15.

[12] Glenn Jocher, "Train custom data in yolov5." https://github.com/ultralytics/yolov5/wiki/Train-Custom-Data, 2020. Accessed: 2021-06-26.

[13] Glenn Jocher, "Yolov5 documentation." https://github.com/ultralytics/yolov5pretrained-checkpoints, 2020. Accessed: 2021-06-26.

[14] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," 2020.

[15] Latif Vardar, "How does tesseract-ocr work with python?." https://medium.com/@latifvardar/how-does-tesseract-ocr-work-with-python-a6bccf85a002, 2018. Accessed: 2021-06-15.

[16] R. Smith, "An overview of the tesseract ocr engine," in *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, vol. 2, pp. 629–633, 2007.