Automatic Number-Plate Recognition
Image Processing Group Project Summary
2022/23 Fall Semester

Barnabás Börcsök BARNABAS.BORCSOK@EDU.BME.HU
Gergő Haragos GERI.HARAGOS@GMAIL.COM
Zoltán Simon ZOLTAN.SIMON@EDU.BME.HU
Bence Sándor Szabó SZABO.BENCE.SANDOR@GMAIL.COM

Budapest University of Technology and Economics

1. Introduction

There are many use cases e.g. in traffic control, where the License Plate (LP) of vehicles must be read. This task can be automated by computers. A wall-mounted or even hand-held camera can take pictures of cars. The image then can be processed by various algorithms to detect the LP, segment the characters and finally recognize these characters. In recent years with the increasing amount of traffic, the need for well-performing Automatic License Plate Recognition (ALPR) systems has increased substantially.

After a discussion of the current state-of-the-art, we propose our own ALPR implementation, evaluating it in multiple visual scenarios.

Our implementation is available at `https://github.com/bobarna/bme-image-processing`, with detailed, reproducible steps for all stages of our pipeline.

2. Previous Solutions

2.1 Automatic Number-Plate Recognition

In the past years many research projects have been focused on ALPR. We have surveyed some of the most recent and most promising works in this field. A recent survey by Shashirangana et al. (2021) gives an overview of different methods and practices used in ALPR.

Texture-based methods use characters present on the LP as the basis for ALPR. Significant color difference between the board and its characters creates a high-frequency color transition. If the image is grayscale, there is an easy to distinguish change of colors between the characters and the background of the board. This creates a unique pixel intensity distribution in the region of the plate. The plate region should have a high edge density. This is used in edge-based systems. In ke Xu et al. (2005) the authors used scan-line technique for ALPR.

Introduced by Redmon et al. (2016) as a novel object detection method, You Only Look Once (YOLO) serves as the basis for the ALPR introduced by Laroca et al. (2019). The naming of YOLO comes from the fact that it performs the object detection for the full image in a single pass. The employed Neural Network (NN) divides the image into regions and predicts bounding boxes and probabilities for each region. Building on this method the authors achieved a license plate recognition rate of 96.9%. The method was tested on multiple different datasets with outstanding results. Besides the novel approach, the authors of this work also released a public dataset of $38,351$ manually labeled bounding boxes on $6,239$ images.

A benchmark for ALPR is introduced by Gonçalves et al. (2016). This benchmark is composed of a dataset helping the License Plate Character Segmentation (LPCS) step. High success rate of this step is crucial for end-to-end success of ALPR. Besides the dataset, the authors also propose a new evaluation

measure of the location of the bounding box within the ground-truth annotation. To further optimize the LPCS step, they suggest a more straightforward approach to perform it efficiently.

3. Method

In this section we discuss the different methods used in our solution. We deconstruct the task of license plate recognition into arbitrary smaller parts. The first great challenge of license plate recognition is determining the location of the plate in the picture. In an upcoming subsection we will introduce the used bounding box detection system. The other major challenge of license plate recognition is actually reading the plate. After we have assigned a bounding box the rest is up to an Optical Character Recognition (OCR) subsystem. An OCR recognizes characters printed on the plate.

3.1 YOLO Object Detection

We used transfer learning to train a YOLOv7 based on the original implementation of the paper by Wang et al. (2022). We detect a single object on each image: license plates. We trained our model for 100 epochs. We achieved around 90% precision on both training, data and validation data, which carried over to images of Hungarian license plates as well.

Although the goal of the project is the detect Hungarian license plates, we observed that a model trained on international license plates generalizes well enough for the object detection problem. This also made us easier to find datasets online, as our Hungarian license plate dataset did not include bounding box data. (See our code repository [1] for more details on the datasets used.)

We used a test data set not seen during training for verifying the model's generalization capabilities after training. Performance on a randomly sampled subset of these test images can be seen in Figure 1.

These results show that the model detects the license plates in almost all cases. Although we can see that the model usually gave a high confidence to the right detection, while further (usually incorrect) detections have been given a lower confidence value, we intentionally kept the confidence threshold low in order not to miss harder to see license plates, even when some other object got incorrectly recognized with a higher confidence.

1. `https://github.com/bobarna/bme-image-processing`

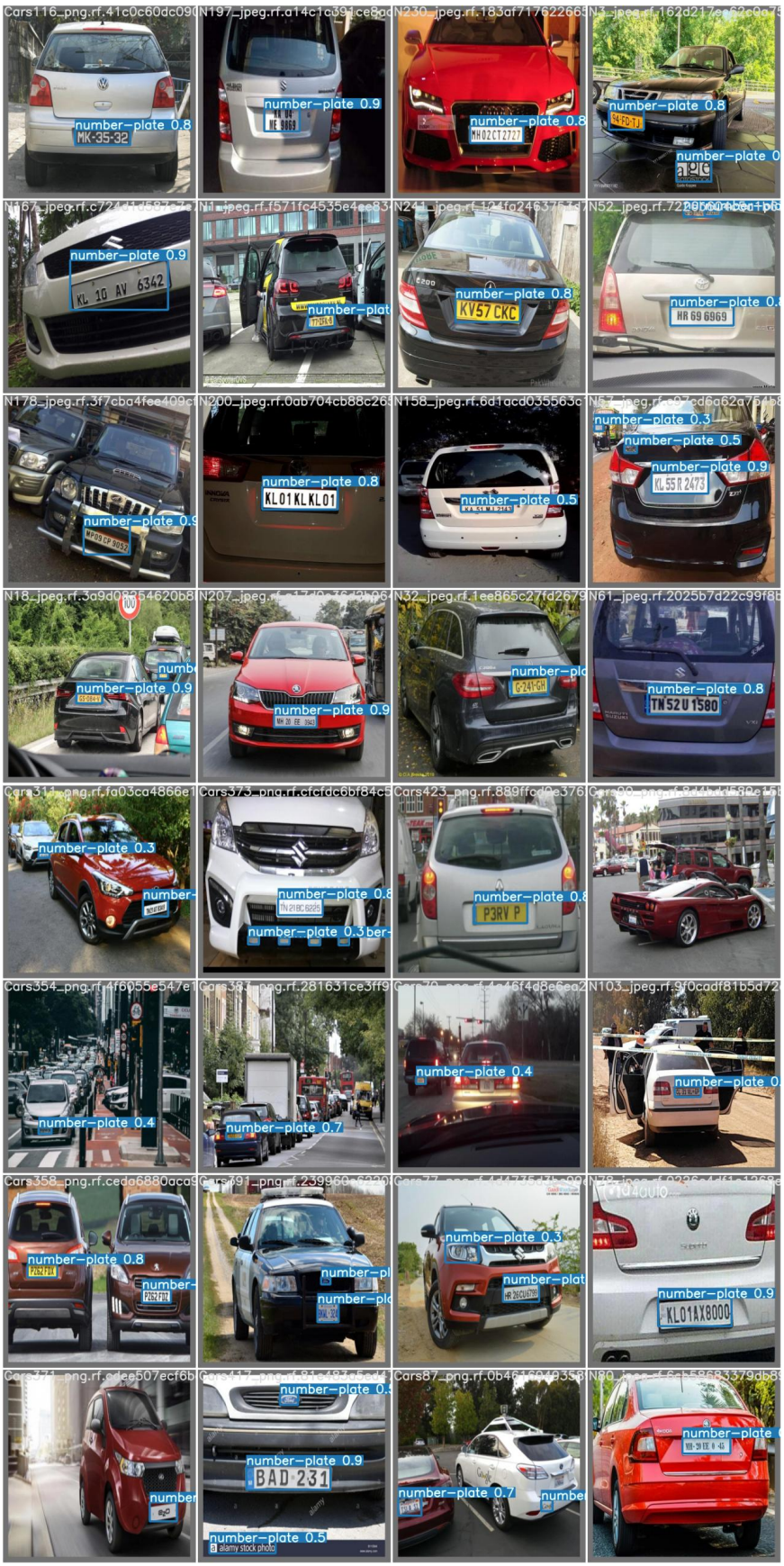

Figure 1: YOLOv7 transfer learning performance on test data. Confidence values are also shown next to each detection.

3.2 Optical Character Recognition

When it comes to OCR we experimented with different methods. Our first approach followed the architecture published by Shi et al. (2017). Unfortunately at the end of the development cycle we had to use a different model, because the original method did not provide the desired results. In the final solution we utilize the *paddleOCR* system by Du et al. (2020). Nevertheless we wanted to describe the original method proposed by Shi et al. (2017), since this was in the focus of our development efforts for the majority of the project. Thus, the following section serves as a brief introduction to an otherwise promising OCR architecture.

We deal with the classic problem of computer vision, image-based sequence recognition. Serial objects, such as scene text, are usually recognized as a sequence of tags rather than a single tag. RNNs can be trained and optimized end-to-end, but require complex post-processing steps before being used. CRNN is a combination of DCNN and RNN and forms an end-to-end system for sequence recognition. It can be learned directly from sequence labels, does not require detailed annotations (such as words), and only requires height normalization in the training and testing sessions.

A CRNN is a type of Recurrent Neural Network (RNN) that can be trained to make a prediction for each frame of a sequence of features that is output by convolutional layers. In CRNN, each column of the field maps corresponds to a region of a rectangle (called the receptive field) of the original image, and the regions are in the same order as the corresponding columns in the feature maps from left to right. A deep bidirectional recurrent neural network (CRNN) is built on top of convolutional layers as recurrent layers.

The recurrent layers predict a label distribution y_t for each frame x_t of the feature sequence. Each time it receives a frame in the sequence, it updates its internal state with a non-linear function that takes both the current input state and the past state as input. Long-Short-Term Memory (LSTM) is a type of RNN designed to capture long-range dependencies that often occur in image-based sequences. LSTM consists of a memory cell and three multiplicative gates, namely input, output and forget gates. It has made a huge improvement in its speech recognition function. Transcription is a process that produces frame-by-frame predictions into a tag sequence using RNN. Each input in CTC is a sequence $y = y_1, \ldots, y_T$, where T is the length of the sequence.

We have developed a neural network that can be trained end-to-end on pairs of images, eliminating the manual labeling of individual components on the training images. The network is trained with error differences calculated using the backpropagation algorithm and stochastic gradient descent (SGD). CRNN is not limited to recognizing a word in a known dictionary and can handle random strings, sentences or other scripts. The recognition accuracy is plotted as a function of d. A larger d results in more candidates, thus a more accurate lexicon-based transcription. On the other hand, the computational cost increases with larger d, due to the longer BKtree search time. CRNN outperforms commercial OMR engines such as Capella Scan and PhotoScore by a wide margin.

CRNN uses convolutional features that are highly robust against noise and bias. It can be easily applied to other image-based sequence recognition probabilities, requiring minimal domain knowledge. CRNN, a new neural network architecture that integrates the advantages of both deep convolutional neural networks and recurrent neural networks. It runs directly on coarse-level labels (such as words) and does not require detailed annotations for each eleventh (i.e., character) in the training phase.

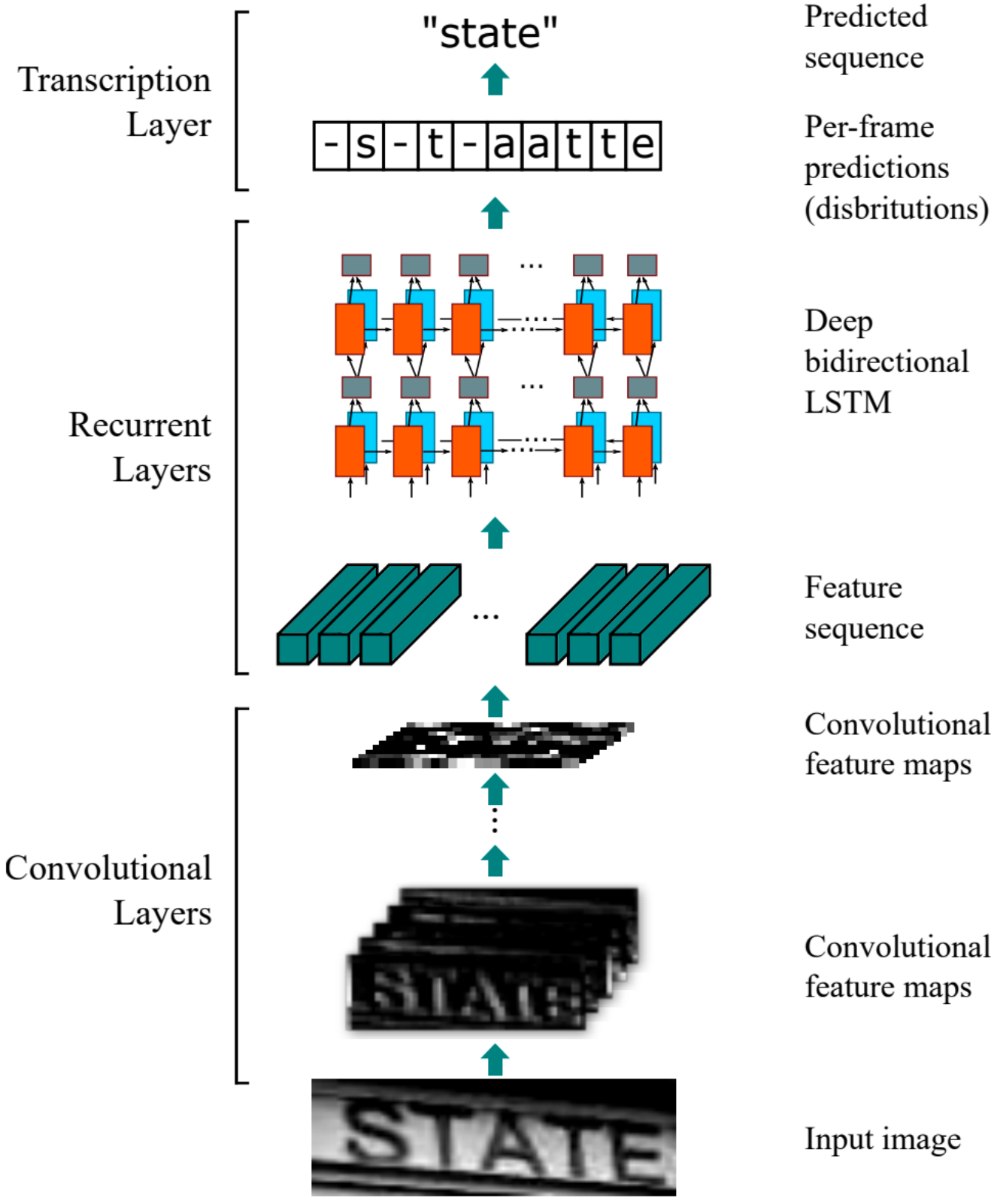

Figure 2: The architecture published in Shi et al. (2017)

Figure 3: Preprocessing results.

3.3 Preprocessing for OCR

After cutting the license plate, we apply the following preprocessing methods with OpenCV:

- *Normalization:* using the `cv2.normalize` method.
- *Noise removal:* using the `fastNlMeansDenoisingColored` function of OpenCV.
- *Erodation:* using a 5×5 kernel matrix of ones.
- *Blur clarification:* for the blur clarification, we used a matrix with a value 9 center surrounded with -1 values.
- *Adaptive Threshold:* using the `adaptiveThreshold` function of OpenCV.

This results in a much better image for the OCR, converting the image to grayscale and sharpening the label of the license plate.

$$\text{Kernel}_{\text{erodation}} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\text{Kernel}_{\text{blur clarification}} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Figure 4: Example YOLO detections, and the corresponding cut outs. Background elements might get incorrectly labeled as number plates. As we mention later on, our goal is to get a high as possible recall. If a lower false positive count was desired, we could increase the confidence threshold.

4. Evaluation

4.1 YOLO Object Detection

Our main goal is to never miss the detection of a potential license plate. This means that we favor recall over precision: we want the correct license plate to be included in the set of detected objects even if this means an increased number of false positives.

Figure 4 shows random detections in different scenarios. We cut out each of the detected objects to separate image files. Some examples can be seen on Figure 5. These cut out detections serve as the input of the next step of our ALPR pipeline.

Figure 5: Some example object detection results that serve as the input for the OCR stage.

4.2 Optical Character Recognition (OCR)

When it comes to OCR, we first tried to train our own NN. We have based our work on the architecture described in Shi et al. Shi et al. (2017). Unfortunately in the end this endeavor did not succeed. We have gained a lot of insight into the given pipeline. Some parts of this OCR implementation worked successfully. First we trained this NN with automatically generated text samples. The text generation was achieved with a *Python* console application called *trdg*. This allowed us to generate images based on a text document. These images contained noise and different levels of distortions. This was meant to precede the training on real world data.

We also experimented with different data augmentation methods. These methods included changing the lighting conditions, skewing images, applying noise and others. One of the used augmentation algorithms is capable of changing the weather conditions of the scene on the picture. Besides applying distortions for data augmentation purposes, we have used inverse, denoising methods to improve the confidence of the recognition process.

After successful training and testing on the automatically generated data, we decided to train our model on the real world images. We used the provided test database containing cars with Hungarian license plates. The NN was trained for hundred epochs. Each epoch contained fifty batches of images. One batch contained thirty-five images. After this amount of training the model began to struggle to further decrease the loss. At this point we decided to use a different model. In the final application we use the *PaddleOCR* Du et al. (2020).

4.3 Qualitative Comparison

For qualitative comparison, we consider scenarios with viewing conditions corresponding to more difficult visual settings. We categorize these cases by perceived difficulty, and show examples.

In conclusion, we found these factors to correspond to an increased difficult for the ALPR task:

- Direct sunlight
- Partial occlusion of the license plate
- Miscellaneous weather effects such as rain or fog
- Small size of the license plate (i.e. the object is further away)

We implemented some of these effects in OpenCV, and show corresponding results in Figure 6.

4.4 Artifical Weather Effects

Inspired by Saxena, we implemented the following natural-looking weather effects with OpenCV to verify the generalization of our method, and improve training:

- *light-rain:* A little blur to the image and random generated lines
- *normal-rain:* Same as light-rain, but more lines
- *snow:* Change the color of some part of picture to white to simulate snow.
- *sun:* Increase brightness for simulating a sunny day

We show these effects in action in Figure 6.

(a) Original Image

(b) Light rain. Although the object detection correctly recognizes the license plate present in the scene, our OCR fails to correctly recognize the letters. This can be due to our preprocessing step introducing unwanted artifacts.

(c) Heavy rain. Even the object detection fails to pick up on the license plate present in the scene.

(d) Snow and Sun filter. The object detection stage of our pipeline performs well in both cases, although the OCR fails to correctly recognize the letters under sunny conditions.

Figure 6: Different types of natural-looking weather effects, and corresponding immediate results: original cut out from the object detection step, and the preprocessing step, that serves as the input to the OCR step.

References

Yuning Du, Chenxia Li, Ruoyu Guo, Xiaoting Yin, Weiwei Liu, Jun Zhou, Yifan Bai, Zilin Yu, Yehua Yang, Qingqing Dang, and Haoshuang Wang. Pp-ocr: A practical ultra lightweight ocr system, 2020. URL https://arxiv.org/abs/2009.09941.

Gabriel Resende Gonçalves, Sirlene Pio Gomes da Silva, David Menotti, and William Robson Schwartz. A benchmark for license plate character segmentation. *CoRR*, abs/1607.02937, 2016. URL http://arxiv.org/abs/1607.02937.

Hong ke Xu, Fu hua Yu, Jia hua Jiao, and Huan sheng Song. A new approach of the vehicle license plate location. In *Sixth International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT'05)*, pages 1055–1057, 2005. doi: 10.1109/PDCAT.2005.24.

Rayson Laroca, Luiz A. Zanlorensi, Gabriel Resende Gonçalves, Eduardo Todt, William Robson Schwartz, and David Menotti. An efficient and layout-independent automatic license plate recognition system based on the YOLO detector. *CoRR*, abs/1909.01754, 2019. URL http://arxiv.org/abs/1909.01754.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2016.

Ujjwal Saxena. Image augmentation: Make it rain, make it snow. how to modify photos to train self-driving cars. URL https://www.freecodecamp.org/news/image-augmentation-make-it-rain-make-it-snow-how-to-modify-a-photo-with-machine-learning-163c0

Jithmi Shashirangana, Heshan Padmasiri, Dulani Meedeniya, and Charith Perera. Automated license plate recognition: A survey on methods and techniques. *IEEE Access*, 9:11203–11225, 2021. doi: 10.1109/ACCESS.2020.3047929.

Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(11):2298–2304, 2017. doi: 10.1109/TPAMI.2016.2646371.

Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, 2022. URL https://arxiv.org/abs/2207.02696.