Reproducibility Challenge: Efficient License Plate Recognition via Holistic Position Attention

Anonymous Author(s)
Affiliation
Address
email

Reproducibility Summary

Scope of Reproducibility

The authors propose a holistic position attention (HPA) license plate recognition (LPR) approach which consists of a position network and shared classifier, claiming that it achieves higher recognition accuracy and computational efficiency compared to state-of-the-art methods. The approach is end-to-end trainable, where character recognition can be performed concurrently, without the need for post-processing. The authors claim that concurrent training is important to the LPR performance and improves the accuracy and speed of the model. The second claim will not be tested, as the results for the end-to-end training are not significantly better than sequential training, and the original authors did not thoroughly prove that their concurrent training resulted in improved computation times compared to sequential training. The authors also claim that ResNet-101 implemented as a base network results in the best accuracy compared to other ResNet models, and that failure cases are due to poor semantic segmentations due to low quality license plate images.

Methodology

We re-implemented the 4-part LPR pipeline in Tensorflow, primarily by referring to the original paper's descriptions and figures of the model architecture. The author's code is available on GitHub implemented in PyTorch; we referred to this resource when the paper did not provide enough detail in order to reproduce the architecture. The original paper implemented BiSeNet [1] for position and semantic feature extraction, we thus referred to [1] in order to implement it. A geforce gtx 1660 GPU was used for computation.

Results

Coming Soon

What was easy

Understanding the overall flow of the framework was easy as the paper had figures and concise explanations. The paper also describes in detail how to build the shared classifier, after having implemented the BiSeNet.

What was difficult

Reproducing this paper provided many difficulties to be faced with. The first difficulty was that the original paper was not clear regarding how BiSeNet was implemented into the pipeline as multiple parts of BiSeNet were renamed in the LPR pipeline. Accessing the proper datasets was also difficult, as minimal details provided in the paper regarding how the datasets were acquired.

Communication with original authors

At this point there has been no contact with the original authors.

Submitted to ML Reproducibility Challenge 2020. Do not distribute.

1 Introduction

License plate recognition (LPR) is one of the most important aspects of intelligent transportation systems since license plates uniquely identify vehicles [2]. To perform LPR, the semantic category and position information of each character should be extracted. Traditional methods include two steps performed separately; first detecting all the characters in the license plate and then cropping character subimages to perform the recognition. More recent methods attempt to perform these two steps in a parallel manner using techniques that involve LSTM, multiple classifiers, and semantic segmentation. The downfall of these attempts include noisy features, poor run-times, difficulty extracting position information of characters, and the need for post-processing.

Zhang et al propose a novel LPR pipeline which achieves superior accuracy and speed compared to previous state-of-the art methods. The proposed pipeline implements holistic position attention to encode the position information of each character in the license plate. Then, a shared classifier is used specifically for character recognition. The methods allow all characters in a license plate to be processed in parallel and eliminates the need for post-processing.

2 Scope of reproducibility

The original work proposes a novel LPR method which utilizes HPA to encode the position information of the license plate characters and a shared classifier to perform character recognition. The authors make many claims in the original study, three of which will be tested in this reproducibility study:

- The HPA approach achieves higher recognition accuracy compared to previous state-of-the-art methods.
- The ResNet-101 based HPA model achieves the best recognition accuracy compared to ResNet-50, ResNet-34, and ResNet-18 when tested on the AOLP, Media Lab, CCPD-base, and CLPD datasets.
- Failure cases are not caused by faults in the network, but are caused by poor semantic segmentations due to low quality license plate images.

These claims will be tested using the methodology in Section 3 and the reproduced results will be reported in Section 4.

3 Methodology

We aimed to re-implement the author's approach from the descriptions in their paper. When this failed, as not enough detail was provided, the author's code published on GitHub, which was implemented in PyTorch was used as a resource. The method also implemented BiSeNet as a large part of the model architecture, and thus we referred to [1] in order to implement it correctly as the original paper lacked a clear description. All experiments were performed on a geforce gtx 1660 GPU.

3.1 Model descriptions

The model is a 4-part pipeline consisting of a backbone network, a semantic network, a position network, and a shared classifier. The backbone network transforms the input image into global features. A ResNet model, pretrained on the ImageNet dataset [cite] was used for this purpose. ResNet-18, -34, -50, and -101 models were all tested as the backbone network, to compare if ResNet-101 had superior performance, which the original paper reported. The semantic network and position network separately extract the semantic and position information of each character and produce semantic features and HPA maps. The shared classifier then takes the semantic features and HPA maps and performs character recognition on the entire character sequence. The first three parts of the pipeline are implemented through a BiSeNet architecture, which consists of spatial and context paths and a feature fusion module. The spatial and context paths are part of the backbone network and are used for common feature extraction. The feature fusion module (FFM) is called upon twice, once for the position network and again for the semantic network to produce the position and semantic features, respectively.

Backbone Network: Spatial Path

The model begins by sending the image into the spatial path. The spatial path consists of three rounds of convolution, batch normalization, and a ReLU activation function. Convolution was performed with a (3x3) kernel, a stride of 2, and padding to preserve the image dimensions. The first round used 64 layers, the second used 128, and the third used 256. The output of the spatial path is then utilized in the position and semantic networks.

Backbone Network: Context Path

The context path consists of a ResNet model pretrained on the ImageNet dataset. The final two layers of the ResNet model, layerA and layerB, are fed into attention refinement modules (ARM). Each ARM consists of global average pooling, 1x1 convolution, batch normalization, and a sigmoid activation function. To produce the ARM output, the output of these 4 layers is multiplied by the original ARM input. The ARM outputs of layerA and layerB, called layerA2 and layerB2, are then upsampled to be of size 28x28. LayerB2 is then multiplied by the tail of the ResNet model, and this result is concatenated with LayerA2 to produce the context path output. This result is used for both the position network and semantic network.

Position Network: FFM

The position network is the FFM of BiSeNet. The output of the spatial and context paths is fed into the FFM and concatenated. Then 3x3 convolution with a stride of 1, batch normalization, and ReLU are performed consecutively to produce output1. Global average pooling is applied next to output1, followed by another 1x1 convolution with a stride of 1, ReLU activation, one more 1x1 convolution with a stride of 1, and sigmoid activation to produce output2. Next, output1 is multiplied by output2, and this product is added to output1 to produce the final output. For each convolution layer in the FFM, the number of filters is equal to the number of classes. The position network aims to extract position features from the image. Typically, a license plate contains 7 character positions, therefore, the number of classes and number of filters used in the position network FFM is 7. To produce the HPA maps, a softmax activation is applied to the final output.

Semantic Network: FFM

The semantic network is also implemented as a FFM. To produce the semantic network final output, the same steps are used as explained above for the position network, however the number of classes is now equal to the number of different characters that could exist in a license plate. There are 26 possible letters and 10 possible characters, therefore the number of filters used in the semantic network FFM is 36. Instead of applying a softmax, batch normalization is performed on the final output to produce the semantic features.

Shared Classifier

In the final section of the pipeline, the HPA map of each character is used to modulate the semantic features separately. The shared classifier is then used for character recognition. It consists of a 5x5 convolution layer, batch normalization, and ReLU activation function to process the features. Global pooling and a fully connected layer are then applied to perform the classification of each character.

3.2 Datasets

AOLP: this dataset consists of 1874 images of Taiwan license plates.

Media Lab: this dataset consists of 706 images of Greek license plate.

CCPD: this dataset consists of 290,000 images of Chinese license plates.

CLPD: this dataset consists of 1200 images of Chinese license plates.

3.3 Hyperparameters

Coming soon.

3.4 Experimental setup and code

Coming soon. Our current code can be found at `https://github.com/emmaritcey/CISC867Project`.

3.5 Computational requirements

Coming soon. The experiments will be performed on a geforce gtx 1660 GPU.

4 Results

No results obtained at this point. 3 out of 4 parts of the model are finished. We were held up on the final part but have figured that out, however not in time to produce results.

5 Discussion

5.1 What was easy

The paper had many figures and clear explanations, which lead to a clear understanding of the overall flow of the framework of the model architecture. The paper also describes in detail how to build the shared classifier layer-by-layer, after integrating the BiSeNet into the model as a whole.

5.2 What was difficult

We faced many difficulties while teproducing this paper. The original paper was not clear regarding how BiSeNet was implemented into the pipeline. Originally, it seemed as though BiSeNet was just the backbone network of the pipeline, however, once we dove deeper into the BiSeNet implementation, we realized that this structure was used for both the semantic network and position network. Accessing the proper datasets was also difficult, as there are many different versions available and there was minimal details provided in the paper regarding how the datasets were acquired.

5.3 Communication with original authors

No communication with the original authors has taken place at this point.

References

[1] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, “Bisenet: Bilateral segmentation network for real-time semantic segmentation,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[2] Y. Zhang, Z. Wang, and J. Zhuang, “Efficient license plate recognition via holistic position attention,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, May 2021.