Final Project: Classification and Detection with Convolutional Neural Networks

Ali Alrasheed
Ajar3@gatech.edu

***Abstract*—**Recognizing house number could be particularly useful in many applications. This report proposes a way to recognize house number using two-stage method: Maximally Stable External Regions (MSER) is used to identify possible regions where digits might exist in the image (ROIs), and a trained Convolutional Neural Network (CNN) to classify these regions of interest. The CNN was trained using Goolge Street View House Numbers dataset (SVHN). A custom model of CNN was proposed and compared with pre-trained and re-trained VGG16 architecture.

1 INTRODUCTION AND RELATED WORK

With the advancement of deep learning and computing power, computer vision has been gaining a lot of attention in recent years. In addition, industries such automation e.g self-driving cars, healthcare, and retails has been heavily utilizing computer vision applications. Object detection is one of the most common computer vision techniques which has been a hot area of research in the past decade.

The straightforward way to approach object detection is the use of sliding window and a CNN classifier. However, this method is slow and computational expensive. Therefore, other methods have been proposed in the literature. For example, Girshick et al., 2015 has proposed a region based CNN (R-CNN) that identify about 2000 regions of interests that are then used as an input to a CNN to classify them. In addition, the same author in Girshick et al., 2015 has proposed fast R-CNN that utilizes a CNN to extract a feature map that is used to extract the region of interest (ROIs) Girshick, 2015. The ROIs are then used as an input to a fully-connected layers to classifying them.

However, a more modern and faster way for object detection are using Single Shot methods that utilize one single CNN for both classification and localization of objects in the image. This provides a much faster inference than the previously cited algorithms. For example, Ren et al., 2015 has also proposed a faster-RCNN method that replaced the slow selective search algorithm in Fast-RCNN by a fast Proposal Region Network (PRN). Another popular method is You Only Look Once (YOLO) that was proposed in Redmon et al., 2016. It utilizes a single CNN to both classify and localize the bounding box of the object using an NxN grid and anchor boxes.Similarly, Single Shot Detector (SSD) was proposed in Liu et al., 2016 that also use a single CNN network to both classify and localize the

bounding boxes of objects in the image.

2 METHOD

The method used in this report is two-stage approach using MSER as region proposal algorithm, and a CNN network to classify each proposed region. The output of the CNN is a probability that the image contains a digit labeled from 0 to 9 or non-digits labeled 10. Any image that has a maximum probability of being non-digit (label 10) is rejected, while other images are accepted with their highest probability label. In addition, Non-Maximum Suppression (NMS) was used to get rid of duplicate bounding boxes. The custom CNN model was designed and then compared to a pre-trained and re-trained VGG16 architecture. More details about the custom model can be seen in 1. It must be noted that the fully connected layers of the VGG16 was also modified to suit the house number recognition challenge (11 labels).

Figure 1—Custom model

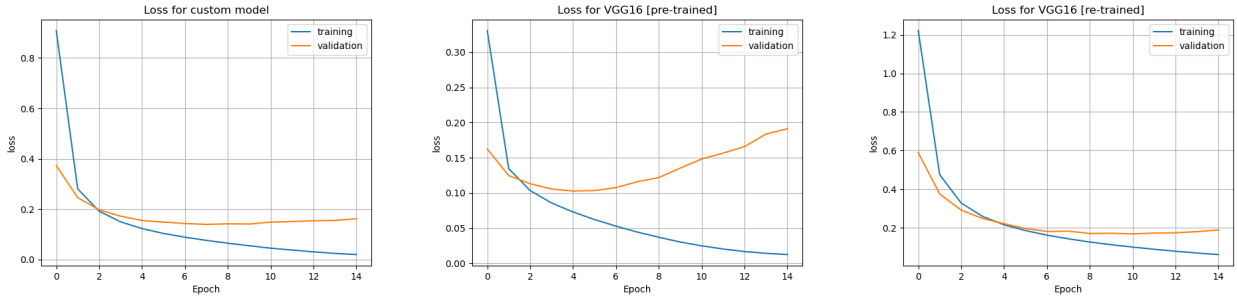
It is important to note that the proposed method in this report is more computational expensive and hence slower than the the state of the art algorithms as it is a two-stage method.

2.1 Training the models

Google Street View House Number view (SVHN) dataset Netzer et al., 2011 was used to train the custom CNN model, pre-trained, and retrained VGG16 models. The format2 (32x32 images) was used for recognizing digits from 0 to 9, and format1's images were used to extract negative images (non-digits). %10 of the training data was splitted randomly for validation, while the test data was obtained directly from the SVHN dataset. Cross entropy loss function was used as it is one of the most effective loss function used for classification challenges. The batch size used for training was 64. The batch size is particularly crucial

hyper-parameter in the training as it can affect the learning stability. The batch size decides the number of images used to estimate the error gradient that is used by optimizer to improve the model's accuracy. In addition, two optimizer was explored in the training: Stochastic Gradient Decent (SGD), and Adam. The SGD optimizer is proven to provide a better generalization than Adam. However, Adam optimizer converges much faster than SDG. The final optimizer choice was Adam as it proved a good result in a reasonable training time (15 Epoch).

As it is seen from figure 2b that the loss for the pre-trained VGG16 model decreased much faster than the custom and re-trained models. This is expected as the pre-trained VGG16 is utilizing the transfer learning technique. The VGG16 was also better at generalizing to new images and the other two models. It must be noted that the weights of the models are only saved once the loss of the validation dataset at its minimum. This is done to avoid overfitting to the training data.



(a) Custom Model *(b)* Pre-trained VGG16 *(c)* Re-trained VGG16

Figure 2—Training Loss

In addition, table 1 shows the accuracy of the saved trained models. The models were saved at the minimum validation loss during training to avoid overfitting to the training data. It can be seen from table 1 that the VGG16 was better in generalizing to the test data than the custom or the retrained VGG16 models.

Model	**Training**	**Validation**	**Testing**
Custom Model	%98	%96	%94
Pre-trained VGG16	%99	%98	%96
Re-trained VGG16	%96	%95	%93

Table 1—Accuracy of the trained models.

3 EXPERIMENT AND DISCUSSION

Figure 3—Correctly labeled images

This section will present some of the results of applying the pre-trained VGG16 model that was trained in section 2.1. The VGG16 model was able to generalize to different fonts, orientations, and scales. It also was able to deal with reasonable Gaussian noise. Figure 3 shows some of the correctly labeled images. It must be noted that there are some false positive in some of the images in figure 3. The negative images was generated from area that doesn't contain digits in the SVHN dataset. Thus, it seems that the most probably reason for the false positives is that these images were not very representative negative images so models could generalize well to non-digits images.

(a) *(b)* *(c)*

Figure 4—Mislabeled images

In addition, the method used in this report could not correctly recognize the house number of all the test images. Figure 4 shows some of images that were mislabeled by the presented method. There are couple of reason that could cause proposed method to fail. Firstly, the Google SVHN dataset is not perfectly labeled. There are many distracting digits in the labeled bounding boxes which could cause some confusion in training the models. Another fail case is when the MSER could not correctly identify the exact location of the digits such as in (a) in figure 4. Furthermore, the model was not trained to detect digits that are very tilted in the image such as in (b) and (c) in figure 4. Another important factor to mention is the imbalance of the dataset as it can be seen in Figure 5 that shows the classes distribution for both training and testing datasets.

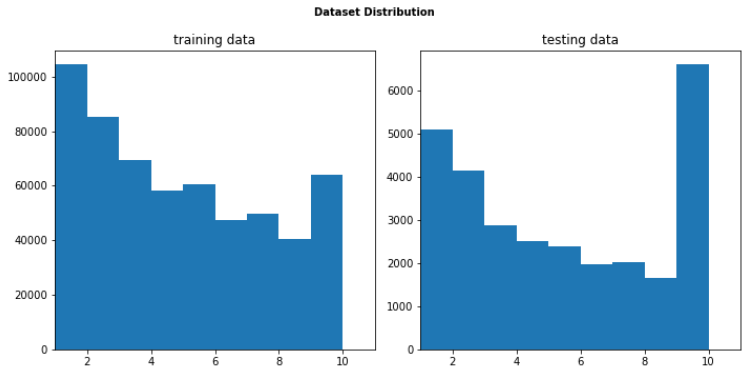


Figure 5—Number of labels per class

Furthermore, to improve the accuracy of the this methods, a correctly labeled dataset with better image resolution could be used. The SVHN dataset could be also relabeled to get rid of the distracting digits. More data could be created by superimposing synthesize data using different font type and sizes as well as different angles into different type of houses. Moreover, single shot methods could be used such as YOLO and SSD to get rid of the

MSER that does not always identity the correct region of interests.

4 CONCLUSION

Two-stage object detection methods using MSER and CNN classifier was used to recognize house number in a given image. This method preformed reasonably well achieving an accuracy of %96 in the testing dataset. It was also shown that although the proposed method was able to correctly recognize the house number using some test images, it also failed to recognized some of the test images. This was due to some reason that was discussed previously such as distractions digits in the labeled images, large tilt angles of the digits, imbalance of the dataset, and failure to identify the exact ROIs using MSER algorithm.

5 DOWNLOAD LINKS

Download link for the VGG16's weights:

https://gatech.box.com/s/q8bu48aozh2ilze233bnz8ecx992nsev

Download link for demo video:

https://gatech.box.com/s/t92lwjnrsjv9tg056i3jndozulap4clm

6 REFERENCES

[1] Girshick, Ross (2015). "Fast r-cnn". In: *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448.

[2] Girshick, Ross, Donahue, Jeff, Darrell, Trevor, and Malik, Jitendra (2015). "Region-based convolutional networks for accurate object detection and segmentation". In: *IEEE transactions on pattern analysis and machine intelligence* 38.1, pp. 142–158.

[3] Liu, Wei, Anguelov, Dragomir, Erhan, Dumitru, Szegedy, Christian, Reed, Scott, Fu, Cheng-Yang, and Berg, Alexander C (2016). "Ssd: Single shot multibox detector". In: *European conference on computer vision*. Springer, pp. 21–37.

[4] Netzer, Yuval, Wang, Tao, Coates, Adam, Bissacco, Alessandro, Wu, Bo, and Ng, Andrew Y (2011). "Reading digits in natural images with unsupervised feature learning". In.

[5] Redmon, Joseph, Divvala, Santosh, Girshick, Ross, and Farhadi, Ali (2016). "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788.

[6] Ren, Shaoqing, He, Kaiming, Girshick, Ross, and Sun, Jian (2015). "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems* 28.