Digit Detection and Classification Within Images

Lance Wilhelm
Georgia Institute of Technology
North Ave NW, Atlanta, GA 30332
lance.wilhelm@gatech.edu

Abstract

This paper attempts to detect and classify digits within images with complex scenes accurately. A two-step algorithm is proposed, using MSER with various filtering techniques to identify ROIs containing digits. Then, a fine-tuned ResNet18 CNN classifies the digit contained within the regions of interest and further filters out unlikely candidates. The algorithm performs well-detecting ROIs for sharp images that contain limited complicated blobs and bold text. The CNN performs well classifying digits within the image squares and achieves a 98.421% accuracy when evaluated against the SVHN test dataset. More work should be done to improve the detection of the ROI detector by using state-of-the-art deep learning models to detect text.

1. Introduction

Image classification using artificial intelligence techniques to include deep learning has been a topic of much research in the previous half-century. The ability of a computer to interpret the raw data within an image, extract meaningful features, and classify those features lies at the core of the task. Feature extraction techniques such as Scale Invariant Feature Transform (SIFT) [19] and Histogram of Gradients (HOG) [7] combined with classification techniques such as support vector machines (SVM) or K-means clustering allowed for the early ability to classify images. However, the advent of the convolutional neural network (CNN) brought about major increases in the ability of computers to quickly and accurately classify images.

1.1. CNN

Fukushima [10] laid the foundation for the CNN in an early journal article with their ”neocognitron,” a network that learned without a teacher to recognize patterns based on geometrical similarity. This work was much later turned into the first effective CNN by LeCun *et al.*[17], called LeNet, which was able to achieve very low error rates on the now famous MNIST [17] dataset. But it wasn't for another 14 years that a CNN breakthrough would come and usher in the current swell of CNN advancements thanks to the increased availability of data, computing power, and interest in deep learning. Krizhevsky *et al.* [16] developed AlexNet in 2012 to tackle the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [24] which utilizes the famous ImageNet dataset [8]. Krizhevsky *et al.* used modern parallel graphical processing units (GPU) and more data to innovate a new network, which included novel architecture to combat overfitting and convergence.

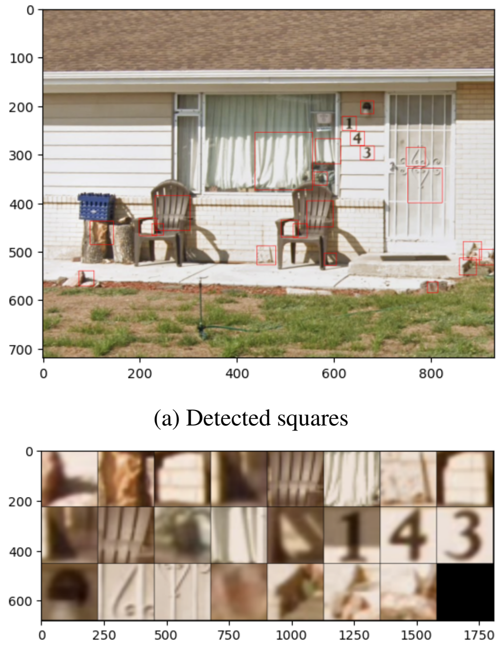
Since AlexNet, modern CNNs have outperformed previous benchmarks at an increasing rate. Ten years after the development of Alexnet, the accuracy of the state-of-the-art models has increased from 63.3% to 91.1% [1]. Notable CNN models include VGG-16 developed by Simoyan & Zisserman [25] and ResNet developed by He *et al.* [13]. Both networks utilized repeated blocks of convolutional layers that increase in planes with depth. Furthermore, ResNet introduced the concept of shortcut connections between the terminal points of each block as a way to prevent vanishing/exploding gradients. Since then, increasingly complex network architectures have provided even more performance gains.

1.2. Text Detection

Text detection within images has also seen a boost in performance since the advent of CNNs. However, text detection and classification have been performed extensively outside of the use of CNNs, much in the same way images were classified before the efficacy of CNNs. Ohya *et al.* used thresholding to select character candidates and then matched them against defined patterns. Coates *et al.* [6] also detected and recognized text within images using features learned using K-means clustering and a sliding window technique. Finally, Goodfellow *et al.* [12] achieved early performance using the AdaBoost classifier [9]. However, Matas *et al.* [20] developed a technique to extract maximally stable extremal regions (MSER) from images that are invariant to rotation and scale and robust to changes in luminance and noise. This technique proved very useful in detecting text objects within an image, as they typically have consistent and stable intensity values. Others have combined detected MSERs with other techniques to filter non-text regions out of the candidate pool. One technique includes the use of geometric features such as region aspect ratio, Euler number, solidity, compactness, etc., to filter our regions either using statistical distributions or through the use of SVMs [3] [11] [27] [18] [23]. Other means of filtering out non-text regions include utilizing the stroke width transform (SWT) [26] [5] [21] [14] [23] [18] [4] and Canny edge detection [4] [14] [21] [5].

1.3. SVHN

This paper tackles specifically digit detection and classification within images. Netzer *et al.* [22] established a dataset containing over 600,000 labeled digits cropped from Street View images which will be used to train a CNN classifier. Their paper discusses the difficulty of reliably recognizing characters in complex scenes such as photographs compared to the practically solved early datasets such as MNIST. Additionally, their paper discusses the inherent combined difficulty of finding and recognizing characters in an image and the compounding potential for failure if one digit in a string of multiple is incorrect. Lastly, their assumption about horizontal text with no vertical overlapping and the apparent spacing between digits leaves room for improvement within their detection and classification techniques.



(a) Detected squares

(b) Extracted Tiles

Figure 1: Example of detected squares after step 1. The algorithm employed in this paper correctly returned ”143” for this image.

2. Approach

This paper utilizes a two-step approach for detecting and classifying text within images.

1. Region of interest (ROI) detection using MSER and filtering using aspect ratio, Canny edges, and non-maximum suppression (NMS).

2. CNN digit classification of the ROIs detected in step 1, including non-text thresholding.

The MSER detector was implemented using the OpenCV python package. The detector was tuned to search for individual digits and not whole words. First, the detected regions and bounding boxes were filtered based on their aspect ratio. For this paper, any region/bounding box with an aspect ratio less than 1 or greater than 3.5 was filtered out. Any regions with aspect ratios outside this range are less likely to be text. Next, the regions/bounding boxes without Canny edges covering more than 10% of their pixels were filtered out. Text within the image is very likely to have detected Canny edges. Therefore, regions not consisting of a minimal amount of canny edge pixels are not likely to contain text. Lastly, the remaining bounding boxes are converted to squares and arranged in ascending order by the x coordinate of the top left point of the square. An example of the detected squares within an image is given in figure 1a, and the resulting tiles arranged in ascending order is given in figure 1b.

The digit classification CNN was trained on the total SVHN dataset to include the additional images provided. The training set was split into a training and validation set to evaluate the model's performance during training and enable the selection of the best model. The model with the best accuracy on the validation set was chosen as the final model and evaluated on a separate test dataset. The dataset sizes are as follows: training: 589736, validation: 14652, test: 26032. All models were trained using a single Nvidia RTX 3090 Ti GPU.

The ROIs detected in step 1 are passed through the trained CNN to receive digit classifications. The output for each image from the CNN is a one-hot vector of 10 length which contains likelihood values for each of the digits (0, 1,. . . , 10). Taking the argmax of the resulting vector indicates which digit is the most likely detected. Early analysis of predictions from this CNN indicated that regions containing text typically have values greater than 5.5. Any ROI with a predicted maximum value less than or equal to 5.5 were filtered out at this point. The resulting classifications are then converted to strings and returned to the console as

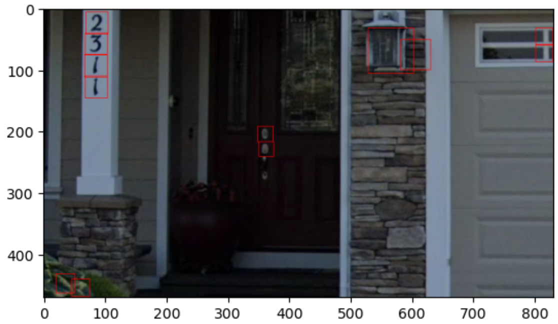


Figure 2: Example of one failure of the algorithm. Vertical squares are not handled correctly in this algorithm. Also, one of the door elements was classified as a ”0” and one of the actual digits was filtered by the CNN threshold. The result of this classification was ”3320”.

the detected string of digits in an image.

3. Results and Analysis

The text ROI detector performed with marginal success. Images that contained bolder digits with fewer conflicting blobs of similar size tended to perform better than images with thinner text in more complex scenes. The MSER detector also does not handle blurry blobs well, and some geometric feature constraints could have filtered out positive ROIs. Cho *et al.* [5] raise the concern of MSER-based detectors improperly filtering out positive text candidate regions in their paper and relate it to the recall criterion. These complications are shown by the lack of digit candidate detection in 3 and the many additional negative ROIs that made it through the filter system in 1. Using CLAHE for image normalization helped the algorithm handle changes in contrast and luminance. An example of its robustness is given in the ”1” digit tile in figure 1b, which contains a shading that splits the image in half.

The final results from the training selection of the CNN model used for step 2, digit classification, are given in table 1 and the resulting training accuracy, training loss, validation accuracy, and validation loss for the most significant models are given in figures 4, 5, 6, and 7, respectively. The best model came from finetuning a pretrained ResNet18 model over 24 epochs. The newer architecture of ResNet proved more effective in training efficiency and validation performance when compared to VGG16. Finetuning a pretrained VGG16 model took approximately 6 times longer per epoch to train and resulted in lower performance. Feature extraction from pretrained models proved ineffective and less appreciably faster to train than the fine-tuned models. Additional models were trained using the original 32 square pixel images in color and grayscale. While their performance was relatively high, they did not outperform the models which transformed the 32 square pixels images into 224-pixel squares. Finally, models that utilized training image augmentation (i.e., scale, rotation, solarisation, affine, warp, etc.) yield appreciable gains in the model's generalization as shown by the validation loss curves in figure 7.

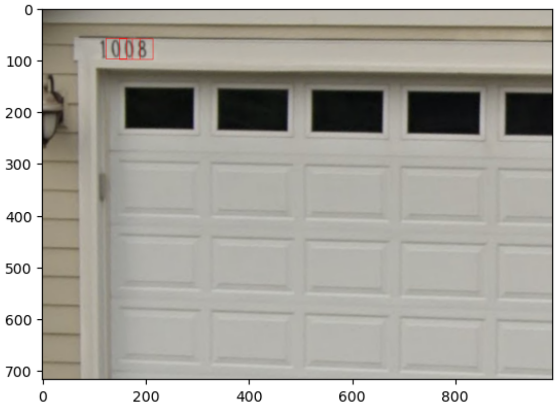


Figure 3: Example of another failure of the algorithm. Not all digits are detected within an image. This could be due to poor MSER performance or too strict of geometric filtering. This issue pertains to the recall criterion.

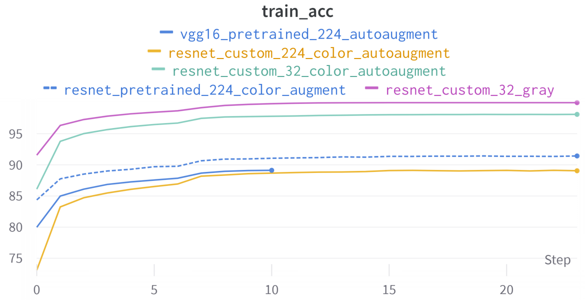


Figure 4: Training accuracy for CNN models

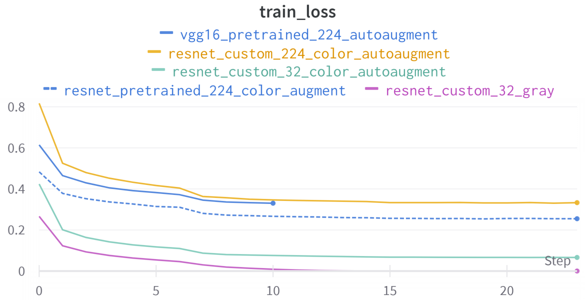


Figure 5: Training loss for CNN models

The loss function for all models, which compares the predicted output with the true labels, was the cross-entropy

Name	Runtime (s)	batch_size	epochs	image_size	learning_rate	optimizer	train_acc (%)	train_loss	val_acc (%)	val_loss
resnet_pretrained_224_color_augment	5978	512	24	224x224	0.0003	Adam	91.43	0.2551	96.44	0.1394
vgg16_pretrained_224_autoaugment	16567	128	11	224x224	0.01	SGD	89.13	0.3305	95.54	0.1701
resnet_custom_224_color_autoaugment	7204	512	24	224x224	0.0003	Adam	89.05	0.3333	95.49	0.1694
resnet_custom_32_color_autoaugment	1706	512	24	32x32	0.0003	Adam	98.11	0.0664	94.57	0.1939
resnet_pretrained_32_color	679	512	24	32x32	0.0003	Adam	99.99	0.0006	94.52	0.4911
resnet_pretrained_32_gray	679	512	24	32x32	0.0003	Adam	99.98	0.0009	94.21	0.4624
resnet_custom_32_gray	699	512	24	32x32	0.0003	Adam	99.999	0.0001	93.71	0.4460

Table 1: Results from CNN training

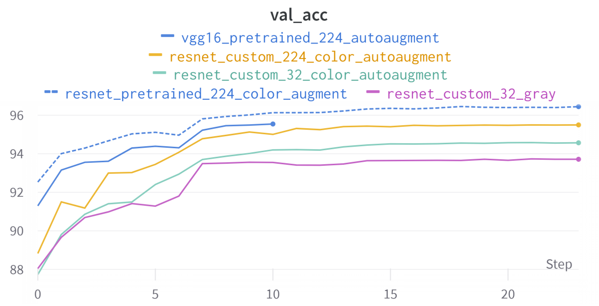


Figure 6: Validation accuracy for CNN models

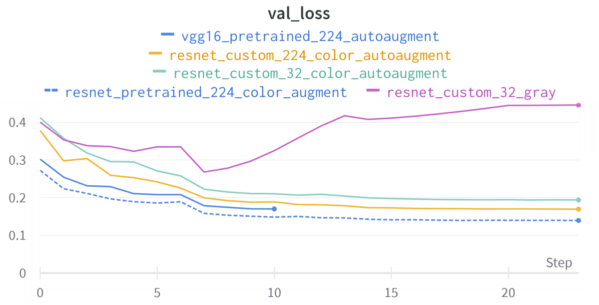
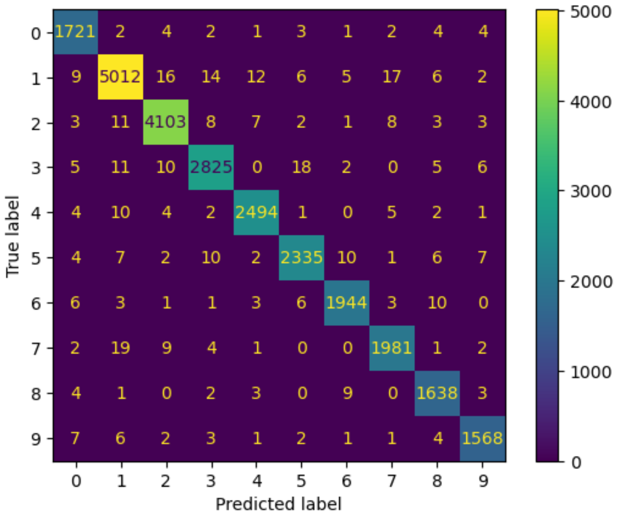


Figure 7: Validation loss for CNN models

loss, as it provides an effective probabilistic comparison between classes in a multi-class classification model. The Adam optimizer was chosen because it converges quickly and efficiently to a solution [15]. SGD has been known to converge to more optimal solutions, but more slowly than Adam. Adam was chosen to allow more model testing in a short time. The learning rate, which determines how new loss information influences the change in the model parameters during training, was set initially to 3×10^{-4} for the Adam optimizer based on cursory research and 1×10^{-2} for SGD. A learning rate scheduler was used to decrease the learning rate by a factor of 10 every 7 epochs, whose effects can be seen in figures 6 and 7. Effectively controlling the learning rate allows the model to find optimums better without passing over them. Lastly, the batch size, the number of samples used to train the model in a given pass, was 512 for all models except VGG16. The batch size was chosen to maximize GPU memory usage to speed up training. Smaller batch sizes, which can result in longer training times, have shown improvements in model generalization.



	0	1	2	3	4	5	6	7	8	9
0	1721	2	4	2	1	3	1	2	4	4
1	9	5012	16	14	12	6	5	17	6	2
2	3	11	4103	8	7	2	1	8	3	3
3	5	11	10	2825	0	18	2	0	5	6
4	4	10	4	2	2494	1	0	5	2	1
5	4	7	2	10	2	2335	10	1	6	7
6	6	3	1	1	3	6	1944	3	10	0
7	2	19	9	4	1	0	0	1981	1	2
8	4	1	0	2	3	0	9	0	1638	3
9	7	6	2	3	1	2	1	1	4	1568

Figure 8: Confusion matrix for the results of the classification of the test SVHN dataset. The final ResNet18 model achieved an accuracy of 98.421% on the dataset. n = 25621

The selected model yielded an accuracy of 98.421% when evaluated on the test dataset. The confusion matrix given in figure 8 shows consistent performance in the classification of digits, with all having an accuracy of 98% or greater, except for the ”5” digit with an accuracy of 97.9%. Further training epochs may have yielded a greater accuracy, but the decreasing performance increase per epoch would have necessitated many more epochs to see a significant gain.

4. Future Work

Better detection of ROIs can be achieved through deep learning. Two notable models that handle text detection within images are EAST [28] and CRAFT [2]. These models are trained to find text within an image and would improve step 1 of the algorithm presented in this paper. Boosting the accuracy of the ROI detection would impact the algorithm’s overall performance, as most of the failures occur in the incorrect classification of text regions.

Furthermore, more exploration of CNNs for digit classification could be performed. There exist published models that currently handle the classification of the SVHN dataset better than ResNet18. However, any gains in classification performance for this algorithm would not outweigh the benefits of increasing step 1 performance. Initial efforts should be focused on improving ROI detection.

5. Conclusion

An attempt at accurately detecting and classifying digits in images was given in this paper. The two-step algorithm proposed uses MSER with various filtering techniques to identify ROIs that may contain digits. Then, a finetuned ResNet18 CNN classifies the digit contained within the regions of interest and further filters out unlikely candidates. The algorithm performs well-detecting ROIs for sharp images that contain limited complicated blobs and bold text. The CNN performs well classifying digits within the image squares, as indicated by the accuracy on the test dataset. More work should be done to improve the detection of the ROI detector by using state-of-the-art deep learning models to detect text. The results of this paper's algorithm would be greatly improved by modernizing the ROI detector.

References

[1] Papers with code - imagenet benchmark (image classification). 1

[2] Youngmin Baek, Bado Lee, Dongyoon Han, Sangdoo Yun, and Hwalsuk Lee. Character region awareness for text detection, 2019. 4

[3] Huizhong Chen, Sam Tsai, Georg Schroth, David Chen, Radek Grzeszczuk, and Bernd Girod. Robust text detection in natural images with edge-enhanced maximally stable extremal regions. pages 2609–2612, 09 2011. 2

[4] Xiangrong Chen and A.L. Yuille. Detecting and reading text in natural scenes. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–II, 2004. 2

[5] Hojin Cho, Myungchul Sung, and Bongjin Jun. Canny text detector: Fast and robust scene text localization algorithm. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3566–3573, 2016. 2, 3

[6] Adam Coates, Blake Carpenter, Carl Case, Sanjeev Satheesh, Bipin Suresh, Tao Wang, David J. Wu, and Andrew Y. Ng. Text detection and character recognition in scene images with unsupervised feature learning. In *2011 International Conference on Document Analysis and Recognition*, pages 440–445, 2011. 1

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. 1

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 1

[9] Yoav Freund and Robert E. Schapire. A short introduction to boosting. 1999. 1

[10] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, 1980. 1

[11] Álvaro González, Luis M. Bergasa, J. Javier Yebes, and Sebastián Bronte. Text location in complex images. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 617–620, 2012. 2

[12] Ian J. Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay Shet. Multi-digit number recognition from street view imagery using deep convolutional neural networks, 2014. 1

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 1

[14] Soukjun Kang, Daewoong Cha, Youngwoo Kim, and Dong Han. Text region extraction in high contrasting image. *International Journal of Future Computer and Communication*, 6:106–109, 09 2017. 2

[15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 4

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25, 01 2012. 1

[17] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[18] Yao Li and Huchuan Lu. Scene text detection via stroke width. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 681–684, 2012. 2

[19] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 1

[20] Jiri Matas, Ondrej Chum, Martin Urban, and Tomas Pajdla. Robust wide baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22:761–767, 09 2004. 1

[21] SNEHA MOHAN.M and VINODKUMAR K. Canny edge detection and mser featuresfor text matching. *International Journal of Advances in Electronics and Computer Science*, 4(8):25–28, Aug 2017. 2

[22] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. 2

[23] Lukáš Neumann and Jiří Matas. Real-time scene text localization and recognition. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3538–3545, 2012. 2

[24] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and

Li Fei-Fei. Imagenet large scale visual recognition challenge, 2015. 1

[25] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. 1

[26] Adiba Tabassum and Shweta A. Dhondse. Text detection using mser and stroke width transform. In *2015 Fifth International Conference on Communication Systems and Network Technologies*, pages 568–571, 2015. 2

[27] Jin-Liang Yao, Yan-Qing Wang, Lu-Bin Weng, and Yi-Ping Yang. Locating text based on connected component and svm. In *2007 International Conference on Wavelet Analysis and Pattern Recognition*, volume 3, pages 1418–1423, 2007. 2

[28] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. East: An efficient and accurate scene text detector, 2017. 4