Recognizing Characters From Google Street View Images

Guan Wang, Jingrui Zhang

guanw@stanford.edu jingrui@stanford.edu

Abstract

Recognizing arbitrary characters in unconstrained natural photographs is a hard problem. In this paper, we address an equally hard sub-problem in this domain - recognizing arbitrary single characters from Street View images. Similar problems, such as recognizing arbitrary multi-digits in street view images [Goodfellow, et al., 2013] and Recognizing Text in Google Street View Image [Lintern, et all., 2008], were well-investigated with decent solutions including localization, segmentation, image feature generation as well as applying machine learning models. In this paper, we propose a unified approach that integrates both localization and segmentation via the use of convolutional neural networks that operates directly on the image pixels. We have experimented with two main types of convolutional neural networks - a thin deep network like Google Network proposed in ILSVRC-2012 competition and a flat shallow network like Alex Network proposed in ILSVRC-2012 competition. We find that the performance of neural network solution works much better than the traditional approach of classification based on image features, while the performance of this approach increases little with the depth of the convolutional network. We evaluate this approach on the publicly available Chars74K dataset and achieve over 84% accuracy in recognizing individual characters in the street view images. Our work could serve as the first step for recognizing a sequence of characters of the text in natural scenes.

1. Introduction

Recognizing characters in photographs captured at street level has significant impact on the quality of map resource utilization. For example, when a user enters a query into Google Maps, they are presented with a list of street level view for the address or target they have searched for. However, problems occur when a user searches for the street view image for a specific business or landmark, they prefer to see their object of interest centered in the Google Street View (GSV) viewport, instead of a list of un-oriented, un-organized street view images around the target or within an arbitrary distance from the target. In order to solve this problems, text recognition is the ideal solution as words recognized from the Google Maps search query can be identified in the nearby images so that the view can be centered upon an instance of text from the search query. Furthermore, the task of image text recognition is typically broken down into two distinct phases, - "text detection" and "word recognition."

In this paper, we presents work towards automatic recognition of text in photographs captured at street view, where text recognition means to classify regions of the image which may contain text, without attempting to determine what the text says. In particular, we narrowed the problem as the recognition of individual characters in such photographs. Previously, optical character recognition (OCR) community spent great effort in recognizing characters in photographs. OCR on constrained domains like document processing is well studied, but arbitrary multi-character text recognition in photographs is still highly challenging. This is because OCR is only a very small subset of the problem in arbitrary multi-character text recognition in photographs with strong constraints enforced to relax the problem. In fact, OCR techniques cannot be applied out of the box precisely due to the wide variability of the text in real world. Figures 1 and 2 are sample images from such scenes, which demonstrate the challenge of this problem [1, 2]. The main reason come from the variety of the data source, where variety studied by Teófilo's [2] comes from (a) font style and thickness; (b) background as well as foreground color and texture; (c) camera position and geometric distortions; (d) illumination; (e) character arrangements; (f) image acquisition factors such as resolution, motion, and focus blurs; (g) lighting, shadows, specularities, and occlusions. All these factors combine to give this a real problem of object recognition rather than OCR, which is much more complicated and general, without any additional constraints enforced to relax the initial problem.

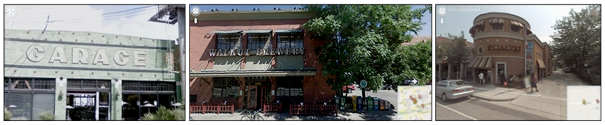

Figure 1: These are sample street view images taken by Google. We found that, there are wide variability of the text in the image, on account of a large range of fonts, colors, styles, orientations, character arrangements, lighting, shadows,

specularities, and occlusions, and image acquisition factors such as resolution, motion, and focus blurs.

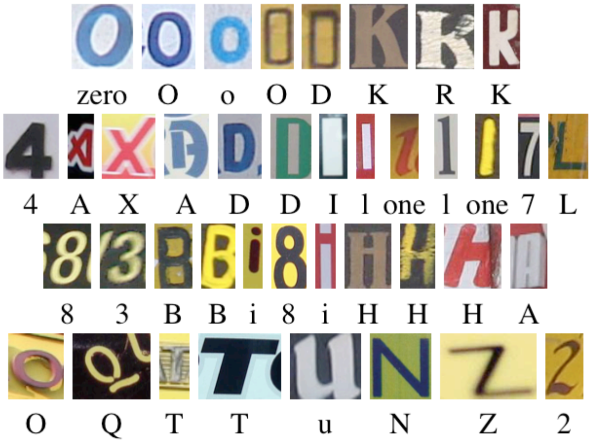

Figure 2: These are examples of individual characters that appear in the text of street view photography. We found that there is high visual similarity between samples of different classes caused mainly by the lack of visual context. For example, character 'O' and '0', '1', 'I' and 'l' are extremely similar in the real world street view images.

In our work, we used Support Vector Machine based on Histogram of Oriented Gradients (HOG) and HSV (hue-saturation-value) features from images to classify individual characters as a baseline for character recognition, while we further designed both shallow and deep convolutional neural networks (CNN) to improve the accuracy for classification. We reached similar best performance of the two types of CNN - over 91% in validation set. We have evaluated this approach on the publicly available dataset Chars74K dataset and achieve over 84% accuracy in recognizing single characters in the street view images. Our work could serve as the first step for recognizing a sequence of characters of the text in natural scenes.

The rest of the paper is organized as follows: Section 2 explores. Sections 3 list the problem definition and describe the processed data. Section 5 and 6 describes the methods and the experimental results. Discussion and future work are concluded in Section 6.

2. Related Work

The task of character recognition in natural scenes is related to problems considered in camera-based object classification and detection. Previously, most of the work in this field is based on locating and rectifying the text areas [6], followed by the application of OCR techniques [7]. However, such approaches have significant limitations - (a) it works well only in scenarios where OCR works well and it is almost impossible to address the cases where foreground/background color and texture varies; (b) rectification step assumes that the image is dominated by text since it is based on the detection of printed document edges. Thus, text recognition largely fails when faced with substantial variation in lighting, viewing angle, text orientation, size, lexicon, etc.

Later, more approaches attempted to go outside of the limited scope of document OCR and deal with variations in foreground/background color and texture. Recognition pipelines based on classifying raw images have been widely explored for digits recognition [5] on the MNIST and USPS datasets. The main idea is to treat each type of digit as a category and build a model that classify each digit in the image as accurate as possible. There are two state-of-arts approaches in this solution. The first approach is to generate features from the image that contains the digits or characters and then apply machine learning models to predict the probability of each class given the input images. If the image consists more than characters, probabilistic graphical models are used to accommodate contextual relationships. For example, Teófiloet al., [2] assess the performance of various features based on nearest neighbour and SVM classification and indicated that the performance of the proposed method, using as few as 15 training images, can be far superior to that of commercial OCR systems. Furthermore, Linternet al., [4] uses a Support Vector Machine based on locally aggregated statistical features from natural scene photographies, reaching an accuracy around 40% in recognizing texts in google street view images. Another approach is to apply convolutional neural networks that operates directly on the image pixels. The earliest work in this approach is done by Matan, Ofer, et al [5], who proposed a feed-forward network architecture for recognizing an unconstrained handwritten multi-digit string. This earliest work reaches an accuracy less than 70%, mainly due to the lack of enough data and computation resources. With the increase in the availability of computational resources and the size of available training sets, it is possible to train a much more complicated and deeper neural networks. Also, algorithmic advances such as dropout training [8] have led to many recent successes in image recognition using deep convolutional neural networks. For example, Krizhevsky et al. [9] made significant improvements in object recognition from a large scale of images - imageNet [10]. In 2013, Goodfellow et al.[3] applied a deep convolutional neural network to recognize arbitrary multi-character text in unconstrained natural photographs, and found that the performance increased with the depth of the convolutional network with over 96% accuracy in recognizing complete street numbers of SVHN dataset and 97.84% accuracy on a per-digit recognition task. Based on the related work, we narrowed our task to recognize single digits from street view images and experimented on both approaches (feature based approach and CNN based approach) discussed above.

3. Problem Description & Data Processing

In this work, we intend to develop a classifier that correctly classifies single characters from street view images into 62 categories – '0~9', 'a~z', 'A~Z'.

3.1. Problem Description

We frame this task as a classification problem where the eventual goal is to map the single characters to the correct type of the class. First, we label 62 categories of characters – '0~9', 'a~z', 'A~Z' as numerical number 0~61 in the same order. For example, label 0 is mapped to character '0', label 10 is mapped to character 'a', and label 61 is mapped to character 'Z'. Then, the problem is, given an input image containing a single character, to find a model that could predict the probability/score of each class and the class with highest probability/score is the correct class for that character.

3.2. Data Processing

We used dataset from the public available *Chars74K dataset* [11], which includes 74k street view images containing both English and Kannada. In our task, we only considered recognition of English character and hence reduced to the initial 74k images into 63k images, covering 64 classes of letters (0-9, A-Z, a-z). From those images, we collected 7705 characters obtained from natural images, 3410 hand drawn characters using a tablet PC, and 62992 synthesized characters from computer fonts. In figure 3, a subset of samples for single characters are shown with various backgrounds in street view images.

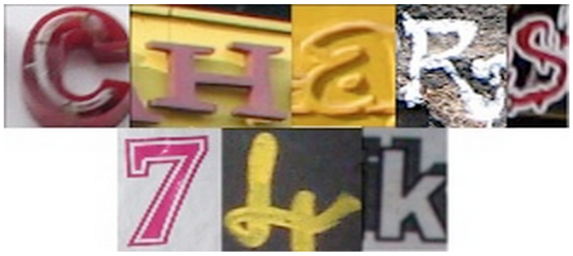

Figure 3: Sample single characters with various backgrounds in *Chars74K.*

We pre-processed image data from *Chars74K dataset* contains 63k images covering 62 different classes of English characters. In order to reduce the I/O cost, we pre-processed each image data and corresponding label, and cached them into a binary format that is readable in Caffe pipeline. We also built a dictionary that maps the 62 classes into numerical labels. Furthermore, since images are of different sizes, we resized the all images into 64x64 pixels with R/G/B 3 channels. In summary, we eventually collected 63k images with 64x64x3 dimensions with numerical labels from 0 to 61.

3.3. Data Split

As listed in the Table1, we split the 63k images into three subsets for training, validation and testing.

Data Split		
TRAINING	VLIDATION	TESTING
40k	12k	11k

Table 1: split data for training, validation, and testing.

3.4. Expected Results

We expect to build a classifier that predicts an unknown image (containing a single character) with probability/score values for each numerical label. The expected model could be SVM, Soft-Max, and Convolutional Neural Networks (CNN).

3.5. Evaluation

We will use 11k withheld images in the test set to evaluate our method. The 11k test images were pre-processed in the same way as those 40k training images and 12k validation images - each were rescaled into size 64x64x3 and a class label. We won't touch this test data unless the classifiers were developed. We will apply the trained classifiers on those 11K withheld images and compare the predicted class label with the true class label. Finally, we mapped the predicted class label to the class name so that our pipeline could serve an application to recognize real-world characters.

4. Technical Approach

4.1. SVM

Our baseline approach is to utilize histogram of oriented gradients (HoG) and color histogram based features on Support vector machine(SVM). Specifically, for each image, our model computes HoG feature as well as a color histogram using the hue channel in HSV color space and combine them together as a vector for each image.

4.2. Convolutional neural networks

4.2.1 Shallow, Flat CNN

We trained two convolutional neural networks: First model has a similar architecture as the Alex's model[12], which has proved to be highly effective in general image classification tasks. Our modified models have the architecture shown in Table 2. All of the modified models have same architecture with only various numbers of filters.

ConvNet Configuration		
A	B	C
Input 64x64 RGB image		
Conv3-128 RELU	Conv3-256 RELU	Conv3-256 RELU
Maxpool 2x2		
Conv5-256 RELU	Conv3-256 RELU	Conv3-256 RELU
Maxpool 2x2		
Conv3-512 RELU	Conv3-256 RELU	Conv3-512 RELU
Conv3-512 RELU	Conv3-256 RELU	Conv3-512 RELU
Conv3-256 RELU	Conv3-256 RELU	Conv3-256 RELU
Maxpool 2x2		
FC-4096		
FC-4096		
Relu		
Dropout		
FC-62		
SoftMax		

Table 2: Simple CNN models with similar architecture but different convolution parameters

4.2.2 Deep, Thin CNN

Intuitively, the most straightforward and effective way to enhance CNN accuracy is to train deep and complicated models, and we have trained a modified model based on GoogleNet [13], which is shown in figure 4. Our model has different

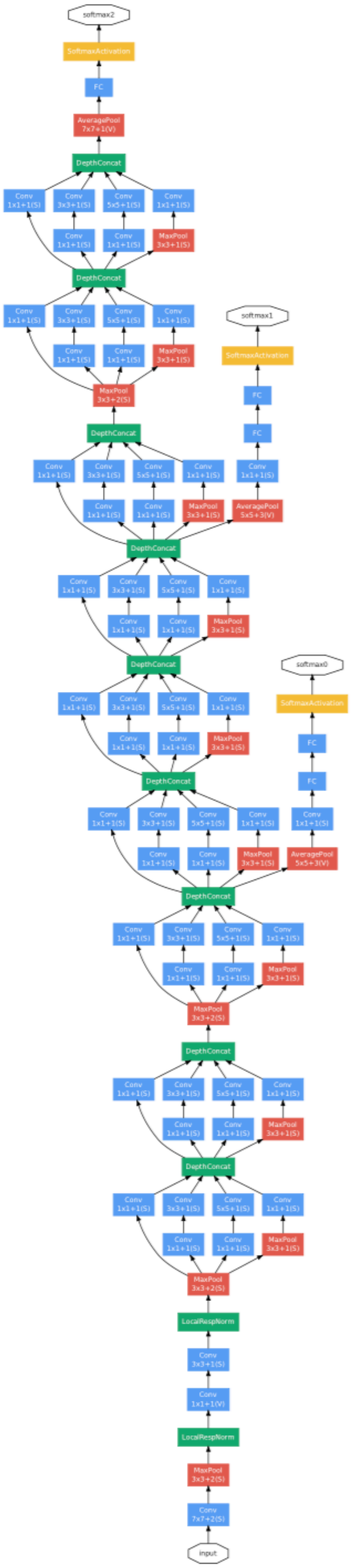

Figure 4: GoogleNet architecture

convolution parameters like filter size, filter number, padding and stride to meet our input data size 64 x 64. Also, size 2 max pooling would be good enough since the input image size is not very large and our model could handle it. Additionally, a 62 classes Fully connected layer is used at the end of the architecture instead of a 1000 classes Fully connected layer in the original Google Net as our task is to predict 10 digit and 26 lowercase letters with 26 uppercase letters. The architecture of the original GoogleNet is shown in Figure 4.

5. Experiment and Results

We utilize Top1 and Top5 evaluation method. As expected, CNN models perform much better than the baseline SVM model. However, it is very interesting that all the three AlexNet have very similar result and more astonishingly that the really complicated GoogleNet model could not outperform the AlexNet models. All the result are shown in Table 2.

	Training	**Validation**	**Test**
SVM with HOG/HSV	88.64%	79.82%	70.01% (Top1)
AlexNet	98.39%	91.22%	84.98% (Top1)
GoogleNet	100%	92.37%	84.46% (Top1) 97.28% (Top5)

Table 3: Evaluation result for all the models

6. Discussion and Future Work

6.1. Discussion

In this paper, we have utilized multiple different models to approach the problem of identify character from images. As the problem is greatly simplified with the help of localization, even the basic HoG/HSV based SVM would be able to reach a decent accuracy. Furthermore, we demonstrate that convolutional neural network is very efficient in image recognition. CNN combines the model selection with feature selection together so that more data specific features would not be missed and over-fitting could be prevented by utilizing dropout. Interestingly, the much more complex googleNet-like CNN model has performed similarly for this tasks. Such similar result could be resulting from the simplicity of the task. The letters and the digits have relatively simple structure and the information provided would not as more rich as more complicated other language characters, such as Chinese and Arabian characters. As a result, all the useful patterns are well extracted from the simple architecture and deeper network would not able to take out new information.

The reasons that our models were unable to reach higher accuracy are twofold. The first reason is that the task is to recognize single character from the image so that there is no context information about the letter or digit as in traditional text recognition. Such context information would benefit a lot on the identification between characters but we would not expect it provide very meaningful context for the digits. The second reason came from the similarity of the certain letters and digit. For example, digit “zero”, letter “o” and uppercase letter “D”; lowercase letter “L”, uppercase letter “i” and digit “1” as shown in figure 5, are originally very similar. With deviation coming from various font and hand written habit, identify character between some of them could be extremely difficult.

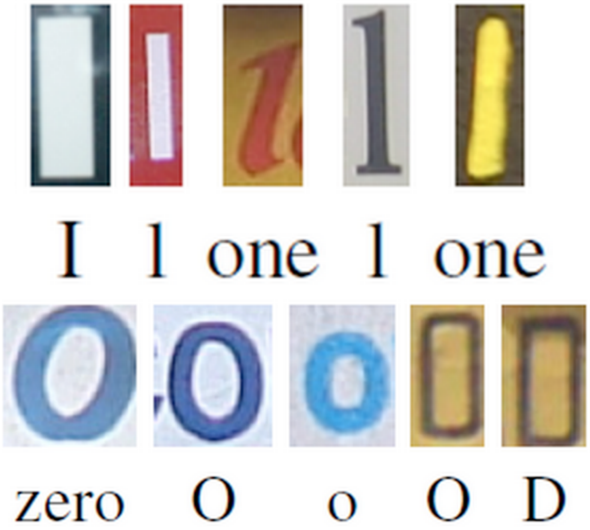

Figure 5: letter I, L and digit 1 in the first row; digit zero, letter o and letter d in the second row have very similar patterns.

6.2. Future Work

Although complicated CNN would unable to outperform simple CNN models, we would expect different models would still have different confidence on predicting different image characters. In order to combine the distinction of different model confidence, training with more CNN models and ensemble their prediction probability would help.

References

[1] The Street View Text Dataset, http://vision.ucsd.edu/~kai/svt/

[2] Teófilo E. de Campos,. Xerox Research Centre Europe,. CHARACTER RECOGNITION IN NATURAL IMAGES. 6 chemin de Maupertuis, 38240 Meylan, France.

[3] Goodfellow, I. J., Bulatov, Y., Ibarz, J., Arnoud, S., & Shet, V. (2013). Multi-digit number recognition from street view imagery using deep convolutional neural networks. arXiv preprint arXiv:1312.6082.

[4] Lintern, James. "Recognizing Text in Google Street View Images." Statistics 6 (2008).

[5] Matan, Ofer, et al. "Multi-digit recognition using a space displacement neural network." (1995).

[6] Kumar, S., Gupta, R., Khanna, N., Chaudhury, S., and Joshi, S. (2007). Text extraction and document image segmentation using matched wavelets and mrf model. IEEE Transactions on Image Processing, 16(8):2117– 2128

[7] Kise, K. and Doermann, D. S., editors (2007). Proceedings of the Second International Workshop on Camera-based Document Analysis and Recognition CBDAR, Curitiba, Brazil. http://www.imlab.jp/cbdar2007/.

[8] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinv, R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. Technical report, arXiv:1207.0580.

[9] Krizhevsky, A., Sutskever, I., and Hinton, G. (2012). ImageNet classification with deep convolutional neural networks. In NIPS'2012.

[10] ImageNet, http://www.image-net.org/

[11] The Chars74K dataset, www.ee.surrey.ac.uk/CVSSP/demos/chars74k/#download

[12] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In NIPS'2012.

[13] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. GOOGLENET: GOING DEEPER WITH CONVOLUTIONS. arXiv:1409.4842