Reading Digits in Natural Images with Unsupervised Feature Learning

Yuval Netzer[1], Tao Wang[2], Adam Coates[2], Alessandro Bissacco[1], Bo Wu[1], Andrew Y. Ng[1,2]
{yuvaln,bissacco,bowu}@google.com
{twangcat,acoates,ang}@cs.stanford.edu

[1]Google Inc., Mountain View, CA
[2]Stanford University, Stanford, CA

Abstract

Detecting and reading text from natural images is a hard computer vision task that is central to a variety of emerging applications. Related problems like document character recognition have been widely studied by computer vision and machine learning researchers and are virtually solved for practical applications like reading handwritten digits. Reliably recognizing characters in more complex scenes like photographs, however, is far more difficult: the best existing methods lag well behind human performance on the same tasks. In this paper we attack the problem of recognizing digits in a real application using unsupervised feature learning methods: reading house numbers from street level photos. To this end, we introduce a new benchmark dataset for research use containing over 600,000 labeled digits cropped from Street View images. We then demonstrate the difficulty of recognizing these digits when the problem is approached with hand-designed features. Finally, we employ variants of two recently proposed unsupervised feature learning methods and find that they are convincingly superior on our benchmarks.

1 Introduction

Reading text from photographs is a difficult unsolved computer vision problem that is important for a range of real world applications. For instance, one application of interest is the problem of identifying house numbers posted on the fronts of buildings. With this information, more accurate maps can be built and navigation services can be improved. Unfortunately, while highly restricted forms of character recognition are essentially solved problems (e.g., OCR of printed documents, or recognition of hand-written digits), recognizing characters in natural scenes is more difficult: characters and digits in photographs are corrupted by natural phenomena that are difficult to compensate for by hand, like severe blur, distortion, and illumination effects on top of wide style and font variations. As a result systems based on hand-engineered representations perform far worse on this task than a typical human. In this work, we seek to solve this problem by employing recently proposed feature learning algorithms to a new benchmark dataset captured from Google Street View images. Specifically, our dataset includes over 600,000 digit images and closely relates to the recognition task that one must solve inside a deployed system. We then demonstrate the consistent superiority of learned representations for this task and their effect on the performance of an end-to-end system. We expect that improvements made by future researchers on this benchmark will directly transfer to improved performance on the complete system.

The problem of recognizing characters in images has been extensively studied in the last few decades, mostly in the context of scanned documents and books [1, 2]. Handwriting recognition has also been widely addressed by both academia and industry [3]. As a result of focused research over many years, automated systems can now perform many of these tasks with accuracy rivaling human beings. For instance, the MNIST digit dataset [4] has been thoroughly addressed, with essentially off-the-shelf algorithms approaching perfect performance [5]. With rapid advances in mobile phone cameras and computation capabilities, however, the more difficult problem of recognizing and understanding scene text is receiving increased attention. In these scenarios, it is clear that specialized models might work better than generic object recognition systems, yet the application is sufficiently new that far less effort has gone into developing successful solutions. Indeed, only recently have end-to-end systems become a target of focused research [6, 7].

In this paper, we will focus on a restricted instance of the scene text problem: reading digits from house-number signs in street level images. As we will show later, traditional vision features are ill-suited to this task. Such features attempt to encode prior knowledge about image structure that tends to be rather brittle, and is often specialized to a specific application, or even a specific dataset. The development of such features is typically a very large part of the expense and difficulty in building successful machine learning systems. In the last few years, however, feature learning algorithms have enjoyed a string of successes in fields such as visual recognition [8], audio recognition [9, 10] and video action recognition [11]. For our application we will use two different feature learning schemes applied to a large corpus of digit data and compare them to hand-crafted representations.

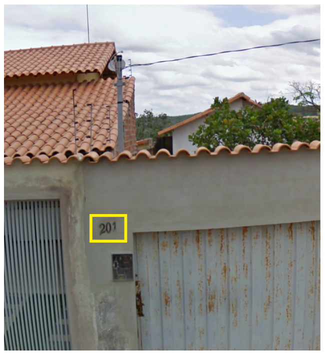

Figure 1: We consider the problem of reading digits from house numbers found on street level images.

In addition to our experimental results that will show the promise of feature learning methods applied to these new types of vision problems, we will introduce a new benchmark dataset for use by researchers. As mentioned above, the venerable MNIST dataset has been a valuable goal post for researchers seeking to build better learning systems whose benchmark performance could be expected to translate into improved performance on realistic applications. However, computers have now reached essentially human levels of performance on this problem—a testament to progress in machine learning and computer vision. The Street View House Numbers (SVHN) digit database that we provide can be seen as similar in flavor to MNIST (e.g., the images are of small cropped characters), but the SVHN dataset incorporates an order of magnitude more labeled data and comes from a significantly harder, unsolved, real world problem. Here the gap between human performance and state of the art feature representations is significant. Going forward, we expect that this dataset may fulfill a similar role for modern feature learning algorithms: it provides a new and difficult benchmark where increased performance can be expected to translate into tangible gains on a realistic application.

To begin, we will explain, at a high level, the goals of our system and introduce the SVHN dataset in Section 2. We will then describe the models, both hand-designed and learning-based, that we have used as first efforts on this dataset in Section 3, and compare their performance to human performance in our experiments in Section 4 and Section 5. Finally, we will conclude with a brief description and experimental evaluation of a full end-to-end application that uses our models in Section 6. In total, we will show not only that learned representations are better benchmark performers in our novel, hard benchmark, but also that these improvements translate to better end-to-end system performance.

2 The Street View House Numbers (SVHN) Dataset

Our main goal is to detect and read house-number signs in Street View images. The entire end-to-end system (described later) includes two main stages: (i) a detection stage that locates individual house numbers in a large image, and (ii) a recognition stage that performs a search over possible character locations in the detected house number, classifying each candidate frame as one of ten digits (0 through 9). The detection stage is similar to the one presented in [12]. Our recognition stage, detailed in Section 6.2, uses a typical image recognition pipeline for each of the candidate digit locations: given a single candidate frame from the detector, the recognition system extracts a set of features and then applies a one-versus-all classifier to make a decision about the character class.

The performance of the recognition module is a major factor determining the overall system performance. Even with perfect character detection and segmentation, a success rate of 90% would make it impossible to perform at better than 90% accuracy in the final system. Worse, most house numbers are composed of multiple digits, and a single error results in a completely incorrect result: for instance, identifying the house number 1600 as 1500. In order to help achieve the (necessarily high) performance requirements for this module, we have collected a very large labeled training set of cropped house-number signs taken from Street View images. Our Street View House Numbers (SVHN) dataset can be downloaded freely from the web: http://ufldl.stanford.edu/housenumbers/

The SVHN dataset was obtained from a large number of Street View images using a combination of automated algorithms and the Amazon Mechanical Turk (AMT) framework [13], which was used to localize and transcribe the single digits. We downloaded a very large set of images from urban areas in various countries. From these randomly selected images, the house-number patches were extracted using a dedicated sliding window house-numbers detector [12] using a low threshold on the detector's confidence score in order to get a varied, unbiased dataset of house-number signs. These low precision detections[1] were screened and transcribed by AMT workers. In total, the dataset comprises over 600,000 labeled characters, and has been made available in two formats:

- Full Numbers - the original, variable-resolution, color house-number images as they appeared in the image file. Each image includes a transcription of the detected digits as well as character level bounding boxes. Figure 2 shows the large variation in character height as measured by the height of the bounding box of the characters in the original Street View images. Figure 3 shows a sample from the Full Numbers ground truth set.
- Cropped Digits - character level ground truth - in this MNIST-like format all digits have been resized to a fixed resolution of 32-by-32 pixels. The original character bounding boxes are extended in the appropriate dimension to become square windows, so that resizing them to 32-by-32 pixels does not introduce aspect ratio distortions.

The dataset is divided into three subsets:

- SVHN train - 73,257 digits for training
- SVHN test - 26,032 digits for testing.
- SVHN extra (train) - 531,131 additional, somewhat less difficult samples, to use as extra training data.

The first 2 subsets were obtained from a large amount of Street View images. The last subset - SVHN extra - was obtained in a similar manner although in order to generate this large amount of labeled data, we've increased the detection threshold considerably (increasing the precision five-fold but limiting recall to approximately half the recall of the original operating point). The SVHN extra subset is thus somewhat biased toward less difficult detections, and is thus easier than SVHN train/SVHN test.

The SVHN dataset, compared to many existing benchmarks, hits an interesting test point: since most digits come from printed signs the digits are artificial and designed to be easily read, yet they nevertheless show vast intra-class variations and include complex photometric distortions that make the recognition problem challenging for many of the same reasons as general-purpose object recognition or natural scene understanding. Nevertheless, it is obvious that digit recognition is much simpler than fully general object classification. There are fewer classes and many forms of exogenous knowledge that allow humans to perform well in other tasks (such as language models or contextual cues) are useless when dealing with isolated digits—thus, such complications should not be necessary in principle to achieve human-level performance.

Finally, we note that our dataset is composed of an extremely large number of small images. It has been shown that humans are remarkably good at identifying whole objects in low-resolution images [14], and thus the same is likely to be true for digits.

[1]99% of the patches considered by the human labelers were false positives as we were interested in reducing bias from the specific detector used.

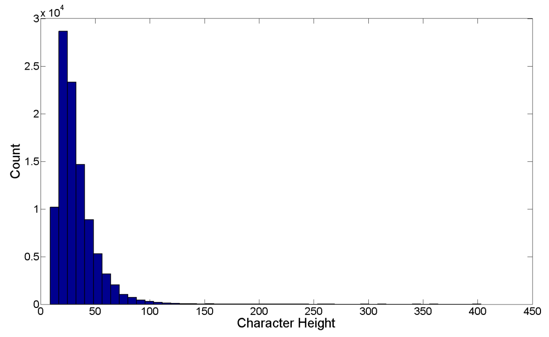

Figure 2: Histogram of SVHN characters height in the original image. Resolution variation is large. (Median: 28 pixels. Max: 403 pixels. Min: 9 pixels.)

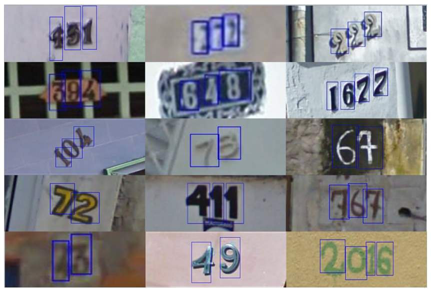

Figure 3: Samples from SVHN dataset. Notice the large variation in font, color, light conditions etc. Blue bounding boxes refer to AMT worker marked bounding boxes of the different characters.

3 Models

We have evaluated various feature representations and classification models on the SVHN datasets. A main thrust of our investigation has been to determine how features generated by feature learning systems compare to hand-constructed feature representations that are commonly used in other computer vision systems. In this section we will briefly overview the models used for our experimental comparisons though we often omit details and refer the reader to prior work.

Each of our models is applied in the same way. Given an input image (32-by-32 pixels, grayscale) we extract features using one of the feature extractors described below to yield a fixed-length feature vector. We then train a linear SVM classifier from the labeled training data using these features as input, and test the classifier on the test set.

3.1 Feature Representations

3.1.1 Hand crafted features

We have tested several hand crafted feature representations on the SVHN dataset: two versions of the widely-used Histograms-of-Oriented-Gradients (HOG) features [15], and an off-the-shelf cocktail of binary image features based on [16].

These features have been popular in traditional OCR systems and thus are natural baseline candidates for our more challenging recognition task. In order to make use of them, we must binarize the grayscale input images. For this purpose we have used the Sauvola binarization method [17] although experiments with various other binarization algorithms (as well as combining multiple binarization algorithms) resulted in similar accuracies.

3.1.2 Learned features

The hand-crafted features described above represent common choices that one might make when attempting to solve the digit classification problem using standard machine learning procedures. Unfortunately, these features both come from application domains with somewhat different desiderata, and thus we would like to find features that are more specialized to our application without engineering them by hand. In this work, we explore the use of several existing unsupervised feature learning algorithms that learn these features from the data itself. Each of these methods involves an unsupervised training stage where the algorithm learns a parametrized feature representation from the training data. The result of this training stage is a *learned* feature mapping that takes in an input image and outputs a fixed-length feature vector to be used for supervised training in the same supervised training and testing pipeline as used for hand-crafted features.

We have experimented with two different feature learning algorithms that may be regarded as fairly standard "off the shelf" tools: (i) stacked sparse auto-encoders and (ii) the K-means-based system of [18].

For the sparse auto-encoders, we have adopted the greedy layer-wise stacking approach of [19]. In each layer of the stacked encoder, we train an auto-encoder with K hidden units using backpropagation to minimize squared reconstruction error with an additional penalty term that encourages the hidden units to maintain a low average activation [20, 21]. After this learning procedure, the decoding layer is discarded and the encoder is used as a nonlinear function that maps its inputs to a K-dimensional feature vector. These features may optionally be used as input to another auto-encoder that is trained in the same way. Once we have built up the feature representation, we add a final softmax classification layer that is trained using maximum likelihood. The full stacked network is finally fine-tuned using back-propagation.

A major drawback of many feature learning systems is their complexity. These algorithms usually require a careful selection of multiple hyperparameters such as learning rates, momentum, sparsity penalties, weight decay, and so on that must be chosen through cross-validation. For exploring the large hyper parameter space of these learned features, we have adopted the high throughput screening framework of [22]. Using a large number of randomly generated models, and using a fast (computationally cheap) screening task on which each model was pre-evaluated, we were able to evaluate thousands of candidate models using an affordable amount of computation.

We have also used the K-means-based feature learning system described by [18], which was recently applied to character recognition in [23]. This system works by first extracting a set of 8-by-8-pixel grayscale patches from training images and then applying a variant of K-means clustering to learn a large bank of K linear filters, where K is the number of centroids produced by K-means. In our experiments we used $K = 500$. The result is a dictionary $D \in \mathbb{R}^{K \times 64}$ of normalized filters that are then convolved with the larger 32-by-32 pixel input image, then passed through the non-linear activation function $g(z) = \max\{0, |z| - \alpha\}$. These features are followed by spatial average pooling in a 5x5 grid to obtain the feature vectors. For classification, we used an L2-SVM[2] as in prior work. No fine-tuning was performed for the features learned by the K-Means clustering algorithm.

4 Experimental Results

We now present our experimental results using the above models applied to the SVHN digit dataset. We selected the best classifiers for each of the different models using cross-validation data taken out from the SVHN-train and SVHN-extra datasets. These top performing models were evaluated on the SVHN-test dataset. We report the final classification accuracies in Table 1.

As can be seen in Table 1, both feature learning methods outperform the hand crafted features by a large margin.

Among the results, the K-means-based system appears to perform slightly better. This appears to come from the spatial-pooling stage, which leads to increased robustness to translations and local distortions. Finally, we note that in prior work binarization has been an important component in scene text applications, driven partly by efforts to re-use existing OCR machinery in new domains [24, 25]. Our results indicate, however, that this approach is unlikely to work well on applications like the one considered here: the binarization algorithm typically fails to separate the characters from the surrounding backgrounds and yields very poor performance.

ALGORITHM	SVHN-TEST (ACCURACY)
HOG	85.0%
BINARY FEATURES (WDCH)	63.3%
K-MEANS	90.6%
STACKED SPARSE AUTO-ENCODERS	89.7%
HUMAN PERFORMANCE	98.0%

Table 1: Accuracies on SVHN-test. See Section 5 for human performance evaluation details.

Figure 4 shows the importance of training on a large labeled training set for this task. With up to 100,000 training examples, performance increases rapidly for all of the methods considered. Though it seems that the performance levels out when using all of our training data, it is clear that the very

[2]The L2 SVM is a SVM trained using squared hinge loss $\max\{0, 1 - \theta^\top x\}^2$.

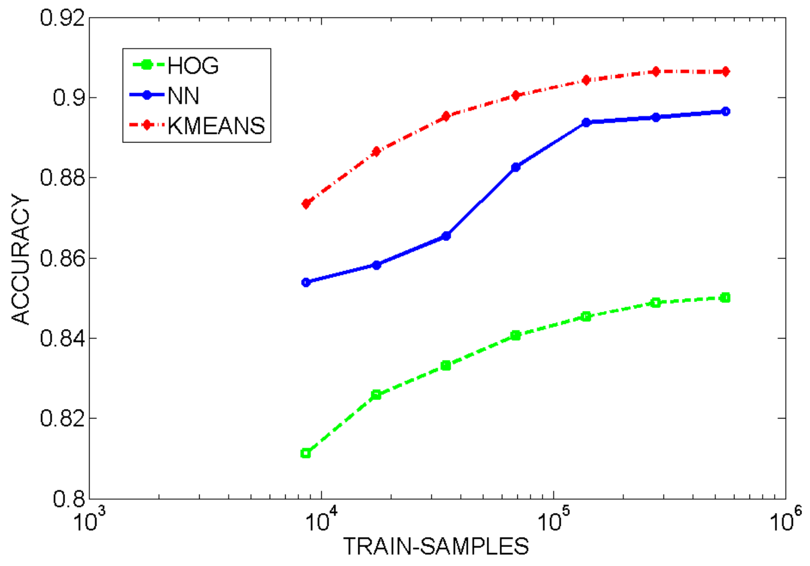

Figure 4: Accuracies vs. number of training samples. Notice the log scale on the x-axis.

large training set is another key to achieving high performance in addition to the use of learned feature representations.

5 Evaluation of Human level performance

As opposed to the clean, isolated, quasi-binary MNIST characters, recognizing a single character of the SVHN dataset is a non-trivial task, even for trained humans. To evaluate how well our models perform in comparison to humans we have sampled images from the set of images that our best model has misclassified. We then evaluated the estimated accuracy of a few of the authors on the original resolution, cropped character, i.e., without the full house-number context. In Table 2, we present our estimated human performance on the failures from the best model as a function of the image resolution. The main causes for human mistakes made in this digit recognition task

Table 2: Human performance on failures of the best model from Section 4

HEIGHT IN PIXELS	$1-25$	$26-50$	$51-75$	$76-100$	$101-125$
HUMAN ACCURACY	$82.0 \pm 2.0\%$	$90.0 \pm 1.5\%$	$96.0 \pm 1.0\%$	100%	100%

were foreground/background confusion and wrong classifications due to very low resolution/blurred images. We've also estimated the unconditioned probability of a human error on samples from SVHN-test and estimate the accuracy of human performance on this dataset as 98.0%.

6 Application: House Numbers Recognition in Street View

6.1 Improving Map Services

In addition to serving millions of users daily with panoramic imagery [26], Street View images taken worldwide are used to improve the accuracy of maps and address geocoding. House numbers detected in the Street View imagery contributes to that goal in various ways.

First, detected house numbers provide us with a view angle in addition to the geocode of the address. This view angle is useful, for instance, when users search for an address in Street View, or when displaying their final destination in driving directions. Without the house number-based view angle the panorama presented will be a default one, which very likely does not point to the desired building. Thus, the user will need to turn the camera by hand in order to achieve the desired view. With the house number-based view angle, the user will be led to the desired address immediately, without any further interaction needed.

Second, house number recognition helps improve address geocoding. In most countries, very few addresses have associated geographic coordinates. The geocode of the majority of addresses is therefore usually computed by interpolating the geocodes of neighboring addresses. Such interpolation introduces errors which may be significant, and may result in poor user experience. With the vehicle's location, and the house number-based view angle, a better geocode of the building of interest can be computed in a straightforward way.

6.2 Automatic detection and recognition of House Numbers in Street View images

Manually detecting and transcribing house numbers in billions of available images is a daunting task. In order to automate this process, our digit recognition system has been deployed within the Street View image processing pipeline. Our aim is to detect and transcribe building numbers in Street View images automatically. Our solution combines three stages: detection, recognition and verification.

First the Street View panos are scanned by a dedicated building number detector. We use a sliding-window classifier based on a set of elementary features computed from image intensities and gradients (see [12] for details). The building number detections are then sent to the recognition module which uses the character classification models described in section 3.1. See also Figure 5.

The building number recognition module is based on a two-step character segmentation/classification approach commonly used for line recognition in Optical Character Recognition systems (see e.g. [27]). Input to the recognizer are the house-number patches obtained by cropping detection boxes from the full Street View images. The approach assumes that digits in these patches are aligned horizontally, with no vertical overlap between them. Although this assumption does not strictly hold for our data (see Figure 3), the approach is quite robust to small deviations and so can handle the slight in-plane rotations present in most house-number patches. To handle larger rotations, affine/perspective distortions due to non-frontal shots or slanted fonts, a preliminary rectification/deskewing processing step would be needed.

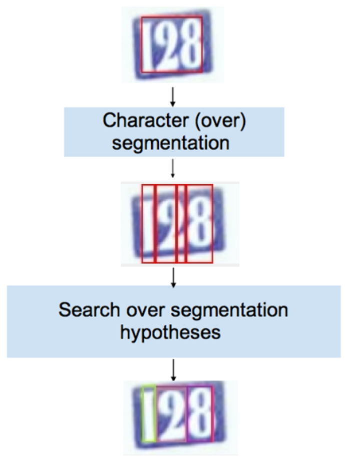
Character (over) segmentation
Search over segmentation hypotheses

Figure 5: Best hypothesis is obtained via a search over the space of possible segmentations.

The recognition consists of two steps: character segmentation, where we find a set of candidate vertical boundaries between characters (breakpoints), and character hypothesis search, where we incrementally search left-to-right the breakpoints pairs, for each pair evaluating the character classifier on the patch having the breakpoints as left and right boundaries, and for each evaluation keeping the top scoring character classes.

In order to limit memory and computational requirements, recognition is performed with a Beam Search approach [28], a modification to Breadth First search where at each step the number of open paths is bounded to the top-N according to an heuristic score function. Devising a good score function is crucial for the success of the approach: In our case the path score is computed from a combination of character classifier scores and a geometry-model score based on deviation of character bounding boxes sizes from a set of reference sizes obtained from typical fonts.

The top results from the recognition module, are sent to a final manual verification step to filter out the false positives and to obtain very high precision at the output of the pipeline.

Figure 6 shows precision/recall curves of various algorithms for the task of house-number recognition. For this experiment, we've used the union of the character level ground truth of SVHN-test as input patches of our OCR engine. As can be seen in Figure 6, the feature learning based system achieves significantly better results than those achieved by the HOG features and both outperform Tesseract, an open source OCR engine which uses an adaptive thresholding algorithm to binarize the

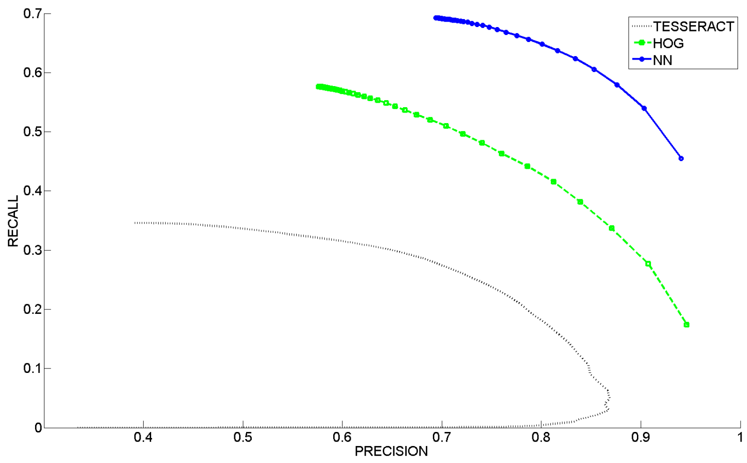

Figure 6: Precision recall curve of various OCR solutions on full house-number images.

input image, and is trained for a significantly different domain, i.e., books and scanned documents. Nevertheless, performance is still far from reaching human level[3], even though the input patches for the recognition are near optimal, as opposed to the loose detections obtained in practice from the detection phase.

7 Conclusion

In this paper we have applied unsupervised feature learning successfully to identify digits in natural scenes for a deployable application. While we have shown the major advantages of learned representations over hand crafted ones, we have also found that there is much room for improvement: computer performance lags well behind human performance on our newly introduced Street View House Numbers dataset. Nevertheless, the increased performance obtained using UFL methods on the SVHN dataset resulted in substantially higher accuracy in our final end-to-end application—a significant success considering that both of the learning methods we applied are virtually ”off the shelf”. We expect that more sophisticated methods developed for the SVHN benchmark may, as it has for MNIST, lead to the human-level performance in the future.

8 Acknowledgment

We thank Krish Chaudhury, German Cheung, Mark Cummins, Tom Dean, Ranjith Unnikrishnan, Shlomo Urbach, Tal Yadid and Jiayong Zhang, for helpful conversations, and for assistance with generation of the Street View dataset and experiments.

References

[1] G. Nagy. At the frontiers of OCR. *Proceedings of IEEE*, 80(7):1093–1100, July 1992.

[2] S. Mori, C. Y. Suen, and K. Yamamoto. Historical review of OCR research and development. *Proceedings of IEEE*, 80(7):1029–1058, July 1992.

[3] R. Plamondon and S. N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Trans. Pattern Anal. Mach. Intell*, 22(1):63–84, 2000.

[4] MNIST. `http://yann.lecun.com/exdb/mnist/`.

[5] D. Cireşan, U. Meier, L. Gambardella, and J. Schmidhuber. Deep, big, simple neural nets for handwritten digit recognition. *Neural computation*, 22(12):3207–3220, 2010.

[6] J. J. Weinman. *Unified Detection and Recognition for Reading Text in Scene Images*. PhD thesis, University of Massachusetts Amherst, 2008.

[3]Human performance on the house number recognition task is reaching 100%, as each house number was transcribed correctly by two different human labelers in order to qualify for the dataset.

[7] K. Wang, B. Babenko, and S. Belongie. End-to-end scene text recognition. In *International Conference on Computer Vision*, 2011.

[8] J. C. Yang, K. Yu, Y. H. Gong, and T. S. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, pages 1794–1801, 2009.

[9] H. Lee, Y. Largman, P. Pham, and A. Y. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Advances in Neural Information Processing Systems 22*, pages 1096–1104. 2009.

[10] G. Dahl, D. Yu, L. Deng, and A. Acero. Context-dependent pre-trained deep neural networks for large vocabulary speech recognition. *Audio, Speech, and Language Processing, IEEE Transactions on*, 2010.

[11] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*, pages 3361–3368. IEEE, 2011.

[12] A. Frome, G. Cheung, A. Abdulkader, M. Zennaro, B. Wu, A. Bissacco, H. Adam, H. Neven, and L. Vincent. Large-scale privacy protection in google street view. In *ICCV*, pages 2373–2380. IEEE, 2009.

[13] Amazon mechanical turk. `https://www.mturk.com/mturk/welcome`.

[14] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *T. PAMI*, 2008.

[15] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages I: 886–893, 2005.

[16] F. Kimura, T. Wakabayashi, S. Tsuruoka, and Y. Miyake. Improvement of handwritten japanese character-recognition using weighted direction code histogram. *Pattern Recognition*, 30(8):1329–1337, Aug. 1997.

[17] F. Shafait, D. Keysers, and T. M. Breuel. Efficient implementation of local adaptive thresholding techniques using integral images. In B. A. Yanikoglu and K. Berkner, editors, *DRR*, volume 6815 of *SPIE Proceedings*, page 681510. SPIE, 2008.

[18] A. Coates, H. Lee, and A. Y. Ng. An Analysis of Single-Layer Networks in Unsupervised Feature Learning. In *AI and Statistics*, 2011.

[19] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11:3371–3408, 2010.

[20] I. Goodfellow, Q. Le, A. Saxe, H. Lee, and A. Ng. Measuring invariances in deep networks. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 646–654, 2009.

[21] H. Lee, C. Ekanadham, and A. Y. Ng. Sparse deep belief net model for visual area V2. In *NIPS*, 2007.

[22] N. Pinto, D. Doukhan, J. J. DiCarlo, and D. D. Cox. A high-throughput screening approach to discovering good forms of biologically inspired visual representation. *PLoS Comput Biol*, 5:e1000579, 11 2009.

[23] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D. Wu, and A. Ng. Text detection and character recognition in scene images with unsupervised feature learning. In *International Conference on Document Analysis and Recognition*, 2011.

[24] A. Mishra, K. Alahari, and C. Jawahar. An MRF model for binarization of natural scene text. In *International Conference on Document Analysis and Recognition*, 2011.

[25] Y. Pan, X. Hou, and C. Liu. A robust system to detect and localize texts in natural scene images. In *International Workshop on Document Analysis Systems*, 2008.

[26] D. Anguelov, C. Dulong, D. Filip, C. Frueh, S. Lafon, R. Lyon, A. Ogale, L. Vincent, and J. Weaver. Google street view: Capturing the world at street level. *Computer*, 43(6):32–38, June 2010.

[27] T. M. Breuel. The OCRopus open source OCR system. In B. A. Yanikoglu and K. Berkner, editors, *DRR*, volume 6815 of *SPIE Proceedings*, page 68150. SPIE, 2008.

[28] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1995.