# Mixed handwritten and printed digit recognition in Sudoku with Convolutional Deep Belief Network

Baptiste Wicht*, Jean Hennebert†
University of Fribourg, Switzerland
HES-SO, University of Applied Science of Western Switzerland
Email: *baptiste.wicht@unifr.ch, †jean.hennebert@unifr.ch

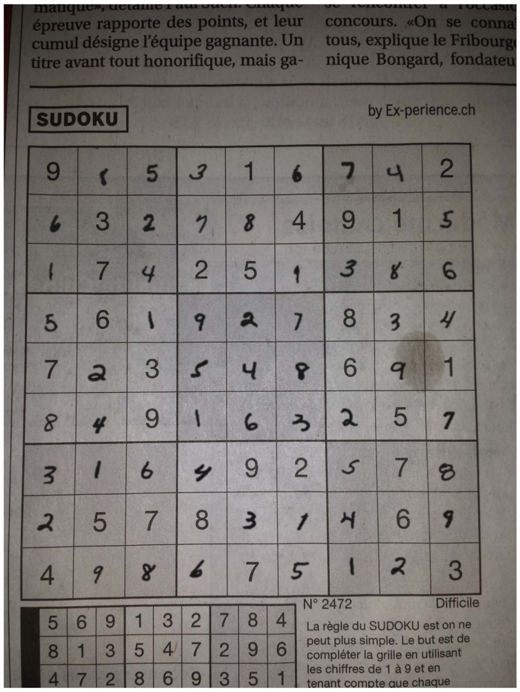
***Abstract*—In this paper, we propose a method to recognize Sudoku puzzles containing both handwritten and printed digits from images taken with a mobile camera. The grid and the digits are detected using various image processing techniques including Hough Transform and Contour Detection. A Convolutional Deep Belief Network is then used to extract high-level features from raw pixels. The features are finally classified using a Support Vector Machine. One of the scientific question addressed here is about the capability of the Deep Belief Network to learn extracting features on mixed inputs, printed and handwritten. The system is thoroughly tested on a set of 200 Sudoku images captured with smartphone cameras under varying conditions, e.g. distortion and shadows. The system shows promising results with 92% of the cells correctly classified. When cell detection errors are not taken into account, the cell recognition accuracy increases to 97.7%. Interestingly, the Deep Belief Network is able to handle the complex conditions often present on images taken with phone cameras and the complexity of mixed printed and handwritten digits.**

***Keywords*—Convolutional Deep Belief Network; Convolution; Text Detection; Text Recognition; Camera-based OCR;**

## I. Introduction

Deep Learning solutions have proved very successful on scanner-based digit recognition [1], [2]. They have also shown good capability to handle complex inputs such as object recognition [2]. The scientific question that we are trying to address in this research is whether such deep learning systems are able to handle mixed contents, for example recognizing both handwritten and printed inputs without separating them in two distinct problems.

In a previous work, we addressed the problem of recognizing Sudoku puzzles from newspaper pictures taken with digital camera such as the ones embedded in our smartphones [3]. The Sudoku puzzle is a famous Japanese game. It is a logic, number-based puzzle. This paper focuses on the standard Sudoku, played on a 9 × 9 grid. Each cell can either be empty or contain a digit from 1 to 9. The game begins with a partially filled grid and the goal is to fill every row, column and sub 3 × 3 square with numbers, so that each number is present only once. Our previous work was based on recognizing initial partially-filled Sudoku, i.e. containing only the printed digits. In this work, we focus on filled Sudoku, containing both handwritten and printed digits. To generate a significant number of Sudoku images, we actually synthesized filled Sudoku images by injecting MNIST digits into the empty cells of the partially-filled Sudoku images. Figure 1 shows an example taken from our dataset. The dataset is made available online for the scientific community.



épreuve rapporte des points, et leur cumul désigne l'équipe gagnante. Un titre avant tout honorifique, mais ga-

concours. «On se conna
tous, explique le Fribourg
nique Bongard, fondateu

SUDOKU

by Ex-perience.ch

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 9 | 1 | 5 | 3 | 1 | 6 | 7 | 4 | 2 |
| 6 | 3 | 2 | 7 | 8 | 4 | 9 | 1 | 5 |
| 1 | 7 | 4 | 2 | 5 | 1 | 3 | 8 | 6 |
| 5 | 6 | 1 | 9 | 2 | 7 | 8 | 3 | 4 |
| 7 | 2 | 3 | 5 | 4 | 8 | 6 | 9 | 1 |
| 8 | 4 | 9 | 1 | 6 | 3 | 2 | 5 | 7 |
| 3 | 1 | 6 | 4 | 9 | 2 | 5 | 7 | 8 |
| 2 | 5 | 7 | 8 | 3 | 1 | 4 | 6 | 9 |
| 4 | 9 | 8 | 6 | 7 | 5 | 1 | 2 | 3 |

N° 2472 Difficile

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 5 | 6 | 9 | 1 | 3 | 2 | 7 | 8 | 4 |
| 8 | 1 | 3 | 5 | 4 | 7 | 2 | 9 | 6 |
| 4 | 7 | 2 | 8 | 6 | 9 | 3 | 5 | 1 |

La règle du SUDOKU est on ne peut plus simple. Le but est de compléter la grille en utilisant les chiffres de 1 à 9 et en tenant compte que chaque

Fig. 1: Image of a Sudoku puzzle from our dataset

In this work, we propose a system which is composed of two parts. In the first part, a set of image processing algorithms including Hough transform and contour detection is used to detect the Sudoku grid and the precise position of each digit inside the grid. In the second part, the isolated digits are recognized using a Convolutional Deep Belief Network (CDBN) and a Support Vector Machine (SVM). Printed and handwritten digits are not distinguished during this process which makes the task, to the best of our knowledge, rather novel.

The rest of this paper is organized as follows. Section II analyzes the previous work achieved in the different fields covered by this research. Section III presents the dataset used to validate the proposed solution. Section IV briefly describes the algorithm used to detect the grid and the digits. Section V presents the architecture used to extract features from the digits. Section VI discusses the overall results of the system. Finally, Section VII concludes this research and presents some ideas for further improvements of the solution.

978-1-4799-1805-8/15/$31.00 ©2015 IEEE

## II. Related Work

### A. Sudoku image recognition

In 2012, A. Van Horn proposed a system to recognize and solve Sudoku puzzles [4], also based on Hough Transform. The four corners of the Sudoku are detected based on the intersections of the detected lines. The digits are then centered in their cells and passed to an Artificial Neural Network (ANN). From each digit image, 100 features are computed. Blank cells are also classified by the ANN and not detected a priori. The system was tested on a rather small set of images.

Simha et al. presented another Sudoku recognition system in 2012 [5]. Adaptive thresholding is applied and components connected to the borders are removed to reduce noise and improve the later character recognition steps. By using another Connected Components algorithm, the largest component area is identified as the grid. Digits inside the grid are then located by labeling the connected components. After that, a virtual grid is computed based on the enclosing box of the grid and each detected digit is assigned to a cell. Finally, the digits are classified using a simple template matching strategy.

None of these methods were thoroughly tested on a well defined dataset. For this reason, we gathered and published our dataset to ensure reproducible results. Moreover, to the best of our knowledge, no research has been attempted on classifying Sudoku with mixed printed and handwritten digits.

### B. Camera-based OCR

Text detection and recognition in images acquired from scanners have been studied for a long time with very efficient solutions proposed [6]. On the other hand, camera-based computer vision problems remain challenging for several reasons. While scanners generally produce similar results, cameras are of various qualities and two cameras may produce different pictures for the same scene. Focus in such devices is rarely perfect and optical zoom is often of poor quality. Pictures are often taken with varying light conditions either natural or artificial and presents shadows and gradient of illumination. While text in a scanner is generally well aligned, images taken with a camera are more likely to be rotated or skewed.

When considering pictures taken from newspaper, several other sources of variability have to be taken into consideration. A newspaper page is rarely completely flat, resulting in distorted images. The font styles and sizes used by different newspapers can also differ. Moreover, the surroundings of the object of interest can be of different nature due to the layout strategy (images, text, margins, ...).

In 2005, Liang et al. published a complete survey of Camera-based analysis of text and documents [7]. The various challenges of this problem are studied in details. The standard steps of image processing (text localization, normalization, enhancement and binarization) are analyzed and different solutions are compared. Although there are many solutions, they show that many problems remain open. In 2013, Jain et al. thoroughly explored the different challenges arisen by Mobile Based OCR [8]. The solutions adopted by standard systems to overcome these challenges are analyzed and compared. They focus on the processing steps allowing later traditional feature extraction and recognition techniques to work as usual. They have shown that even if solutions are getting better, there is still room for improvement.

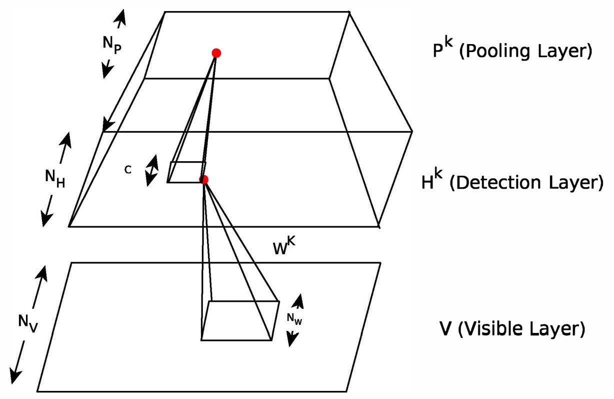


Fig. 2: Convolutional RBM with max pooling

### C. Convolutional DBN

Deep Belief Network (DBN) were first introduced by G. Hinton and R. Salakhutdinov in 2006 [9]. It is a novel way of training deep neural network efficiently and effectively. A DBN is a deep neural network composed of several layers of hidden units. There are connections between the layers, but not between units of the same layer. DBNs are typically implemented as a composition of simple networks, generally Restricted Boltzmann Machine (RBM). Using RBMs for each layer leads to a fast, unsupervised, layer by layer, training method. For that, Contrastive Divergence is applied to each layer in turn. RBMs and DBNs can then be used to learn a feature extractor in a data-driven way on a given set of observation data. This is an important property that removes the burden of finding and tuning *hand-crafted* feature extraction algorithms. To then turn the network into a classifier, fine-tuning strategy can be applied to the whole network to finalize the training [1]. This is comparable to the backpropagation algorithms used to optimize a standard neural network.

In 2009, Lee et al. presented their Convolutional Deep Belief Network (CDBN) [2]. This solution allows to scale the network up to handle larger image sizes. Moreover, the learned features are usually more robust to scaling variabilities and the convolution brings translation invariance to the DBN. They demonstrated excellent performance on visual recognition tasks. The authors also introduced the notion of Probabilistic Max Pooling leading to probabilistic networks such as Convolutional Restricted Boltzmann Machine (CRBM). Moreover, their network achieved state of the art performances on the MNIST dataset and showed excellent results in learning deep features for object recognition. Figure 2 shows an example of such a CRBM with Probabilistic Max Pooling. Krizhevsky [10] developed a fine-tuned CDBN for the task of object recognition with special capped Rectified Linear Unit (ReLU) in the hidden layer and an alternative model for the biases.

Since their discovery, Deep Belief Networks and Deep Architectures in general have been used in several domains (Face Recognition, Reinforcement Learning, Handwritten Characters Recognition, etc.). They have proved very successful, often achieving state of the art results.

## III. Dataset

We gathered and compiled the Sudoku Recognition Dataset (SRD) that we used to thoroughly test the proposed approach. This dataset is freely available online[1]. To release a significant number of *filled* Sudoku images, the dataset has been mostly synthesized using handwritten digits from the MNIST dataset. Randomly selected MNIST digits were automatically integrated in the empty cells of the Sudoku puzzles, using the ground truth information about the location of filled and empty cells. The procedure involved resizing and centering the digits in the cell, using transparency to keep parts of the initial artefacts of the grid and filtering to obtain realistic results.

The dataset contains 200 Sudoku images, taken from various cell phones and from different Swiss newspapers. The dataset is separated in a training set of 160 images and 40 images for testing. The images have been taken with 11 different phones. The images are coming from old phones (three years old) and modern smartphones (less than one year old). The pictures are generally centered on the Sudoku, but include text, images and sometimes even other partial Sudoku puzzles. The conditions of the images vary greatly from one to another, for example showing blurred parts, shadows, illumination gradients etc. Several images were taken on newspaper pages that were not perfectly flat, resulting in distorted puzzles.

## IV. Preprocessing

The digit detection step follows the approach of our previous work [3]. The detection procedure had to be tuned in order to handle handwritten digits with thinner strokes and the fact that there are no empty cells. The detection steps are:

1) Edges of the binary image are detected using the Canny algorithm [11]. Segments of lines are then detected using a Progressive Probabilistic Hough Transform [12]. The Hough transform is a standard computer vision technique designed to detect lines. The probabilistic version of this algorithm detects segments rather than complete lines.
2) The Hough algorithm detects many segments on the same line. Therefore, a simple Connected Component Analysis [13] is performed to cluster segments together and find the group that is the most likely to form a Sudoku.
3) If the cluster and its segments are correctly detected, there will be 100 intersections between its segments and these points will be taken as forming the grid. Otherwise, a Contour Detection algorithm [14] is used to find the largest contour inside the image. In which case, the outer points of the contour are considered. A quadrilateral computed from these outer points is then considered as the final Sudoku grid.
4) Once all the cells have been properly detected, the digits are isolated using another Contour Detection to find the best enclosing rectangle of the digit.

Since the feature extractor expects equally-sized squares, the final rectangle is enlarged to a square and resized to $32 \times 32$. More information on these steps is available in [3].

## V. Feature Extraction

Features are extracted from the Sudoku puzzles using a Convolutional DBN that is trained in an unsupervised manner.

### A. Convolutional RBM

A Convolutional RBM (CRBM) with Probabilistic Max Pooling is made of three layers. The input layer is made of $N_V \times N_V$ binary units ($v_{i,j}$). There are $K$ groups (or “bases”) in the hidden layer and each group is an array of $N_H \times N_H$ binary units ($h_{i,j}$). There are $K$ convolutional filters ($W^k_{i,j}$) of shape $N_W \times N_W$ ($N_W \triangleq N_V - N_H + 1$), connecting the layers together. The filter weights are shared by all hidden units of a group. There is a bias $b_k$ for each hidden group and every visible unit share a single bias $c$. The pooling layer has $K$ groups of binary units, each group of size $N_P \times N_P$. Pooling shrinks the hidden representation by a $C$, usually small ($N_P \triangleq N_H/C$).

Sampling each unit can be done as follows:

$$P(v_{i,j} = 1|h) = \sigma(c + \sum_{k}^{K}(W^k *_f h^k)_{i,j}) \tag{1}$$

$$B_\alpha \triangleq (i,j) : h_{i,j}\text{belongs to block } \alpha \tag{2}$$

$$I(h^k_{i,j}) \triangleq b_k + (\tilde{W}^k *_v v)_{i,j} \tag{3}$$

$$P(h^k_{i,j} = 1|v) = \frac{exp(I(h^k_{ij}))}{1 + \sum_{i',j' \in \beta_\alpha} exp(I(h^k_{i',j'}))} \tag{4}$$

$$P(p^k_\alpha = 0|v) = \frac{1}{1 + \sum_{i',j' \in \beta_\alpha} exp(I(h^k_{i',j'}))} \tag{5}$$

This results in a network with the given energy function:

$$E(v,h) = -\sum_{k}^{K}\sum_{i,j}(h^k_{i,j}(\tilde{W}^k * v)_{i,j} + b_k h^k_{i,j}) - c\sum_{i,j} v_{i,j} \tag{6}$$

This network can be trained with standard Contrastive Divergence with the weight gradients obtained using convolution.

### B. Feature Extraction

A Convolutional DBN (CDBN) is used to extract higher-level features from the digits. The input of the CDBN is a $32 \times 32$ grayscale image ($N_V = 32$). Different experiments have been run to compare the accuracy with binary, grayscale and RGB inputs. Binary inputs were leading to decreased performance in comparison with grayscale and RGB which performed equally well. Considering this, grayscale images have been used.

Our CDBN has two layers, each being a CRBM with Probabilistic Max Pooling. The first layer uses Gaussian visible units and binary hidden units and has 40 bases of $11 \times 11$ pixels

[1] https://github.com/wichtounet/sudoku_dataset

TABLE I: Training parameters for each layer of the CDBN

| Layer | Learning rate | Sparsity Target | Momentum[2] | Weight decay |
|---|---|---|---|---|
| First layer | $1 \times 10^{-5}$ | 0.08 | $0.5 \rightarrow 0.9$ | $2 \times 10^{-4}$ |
| Second layer | $2 \times 10^{-3}$ | 0.06 | $0.5 \rightarrow 0.9$ | $2 \times 10^{-4}$ |

($K = 40$, $N_H = 11$). The second layer uses binary visible and hidden units and has 40 bases of size $6 \times 6$ ($K = 40$, $N_H = 6$). The pooling ratio $C$ is set to 2 in both layers. Both CRBM have been trained in an unsupervised manner using Contrastive Divergence (CD), for 100 epochs. Although several steps of CD may improve the features learned by the network, one step is generally enough [15]. Momentum and weight decay were applied on the CD updates on weights and biases.

A CDBN model is highly overcomplete, i.e. the size of the output representation is larger than the size of its input. With a small convolutional filter, the model is overcomplete roughly by a factor of $K$ since the first layer contains $K$ bases, each roughly the size of the input image. In practice, overcomplete models have the risk of learning trivial solutions, such as pixel detectors. The most common solution to this problem is to enforce the output representation to be sparse in that for a given stimulus in the input, only a small fraction of the output is activated. The proposed system follows Lee at al. regularization method [16]. The following update (applied before weight updates) has been used during training:

$$\Delta b_k^{\text{sparsity}} = p - \frac{1}{N_h^2} \sum_{i,j} P(h_{i,j}^k = 1|v) \qquad (7)$$

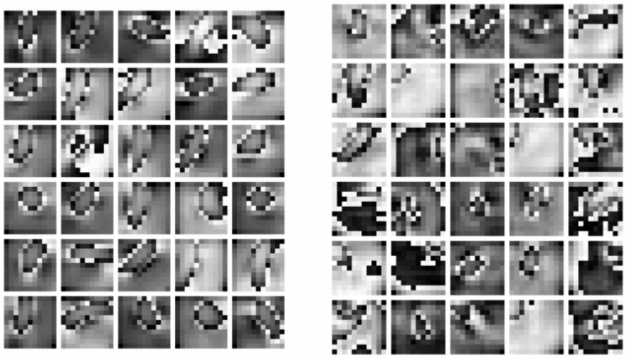
Where $p$ is the target sparsity. This update is applied to the visible biases with a specific learning rate. The sparsity learning rate has to be chosen so that the target sparsity is reached while still allowing the reconstruction error to diminish over the epochs. Table I synthesizes the parameters used for training.

## VI. Classification results

A multiclass Support Vector Machine (SVM) [17] classifier with a Radial Basis Function (RBF) kernel has been used for classification. The parameters of the kernel ($C$,$\gamma$) have been selected using a grid search with cross-validation. The input feature vectors are computed by concatenating the activation probabilities of the first and second pooling layers.

The classifier is trained on the training set (160 images, 12960 digits) and tested on the test set (40 images, 3240 digits). The overall digit recognition rate, mixing handwritten and printed inputs, is 91.98%. This results shows that the task remains difficult but with an interesting capability of the system to cope with mixed input natures. The Sudoku “grid-level” accuracy is 62.5%, i.e. 25 of the 40 grids have the 81 digit inputs perfectly recognized.

Some parameters can be tuned independently of each others, such as weight decay and momentum. However, some parameters need to be considered together, such as the learning rate of the gradients and the target sparsity. It is important to enforce sparsity in order to learn high-level features, without being detrimental to the accuracy of these features. While an higher learning rate may speed up the training, it may be decreasing the quality of the features learned. On the contrary, a too small learning rate may never converge to an acceptable solution. Moreover the sparsity target must also be considered alongside the number of hidden units and the number of bases. Generally, the parameters of one layer of the CDBN can be tuned independently of other layers. SVM parameters are highly dependent on the features learned by the CDBN and must be tuned again after each change of the CDBN parameters. Figure 3 shows that a slight change in sparsity may lead to entirely different filters.



(a) $sparsity\_target = 0.08$ (b) $sparsity\_target = 0.10$

Fig. 3: Weights learned by the first layer using different sparsity target. Both networks were trained for the same number of epochs and only the sparsity target changed.

The errors mostly come from two causes. First, many misclassifications are caused by imperfect detection of the digit, leading to only a small part of the digit to be detected or a large part of the background being included. Secondly, some images are too blurry or have significant noise which hinders the accuracy of the classifier if it was not trained on images exhibiting similar conditions.

The classifier is trained and tested on digits that were detected by the digit detector (see Section IV) and these results are not always perfect. Sometimes, parts of the Sudoku grids are detected alongside the digits. Moreover, several images are very noisy. For these reasons, performance of the system were also evaluated using a perfect digit detector. This detector could be easily built relying on the ground-truth meta-data of the grids. The performance at the digit level is then increased to 97.72% (error rate of 2.28%) and the grid-level accuracy goes up to 80%. In this case, two types of error remain. First, classification errors between visually similar digits such as 1 and 7. Secondly, classification errors likely induced by a lack of genericity in the learned features or overfitting during training. About 70% of the errors are found on handwritten digits that have higher inherent variability. These results cannot be directly compared to the state of the digit recognition results. Indeed, digit database have more samples and much less variations. For instance, MNIST digits are perfectly centered and binarized. Moreover, these recognition results are depending on the quality of all the previous passes.

[2] Momentum is increased after 10 epochs

## VII. Conclusion and Future Work

We designed and implemented a complete solution to detect and recognize a Sudoku grids containing both printed and handwritten digits. The grid and cell detection part is using various image processing techniques including Hough Transform and Contour Detection. The recognition part is based on a feature-driven CDBN trained in a unsupervised way on mixed printed and handwritten inputs, and on a SVM classifier. While improvements can certainly be brought to the detection part, the overall system is offering good overall performance in spite of the difficulties inherent to camera-based inputs including variabilities of illumination, autofocus artefacts (blurred parts), skewed and rotated inputs. An interesting result is in the capability of CDBN to handle mixed inputs and extract relevant features on both handwritten and printed inputs. A result of our work is also in the dataset of 200 images of Sudoku puzzles, containing both unfilled and filled grids that we made available for the community to perform their own experiments.

While our system gives promising results, we foresee several extensions for this work:

- The detection part could be improved, either tuning further the front-end algorithms, either applying also a data-driven method. In this last direction, a CDBN coupled to a scaling sliding window detection system could probably be used to detect digits in the newspaper pictures. This method would be more generic than our detection system (or at least requiring less hand-tuning) and may improve our results.
- Overfitting could remain a problem in our case. More care should be put in ensuring that the weights of the CRBMs are not too tightly coupled to the training set. Generating more training examples could help limit overfitting.
- The SVM training system is rather slow and requires also a good deal of tuning. We believe that using fine-tuning on the Convolutional DBN would be faster and may lead to better results. For this to work, it would be necessary to develop a variant of Stochastic Gradient Descent (SGD) or Conjugate Gradient (CG) for CDBNs [18].
- There are several variations of training methods for CDBN and CRBM, while only CD was considered in this work. Other CDBN training procedures could be used to compare them and possibly improve the current results.

## Acknowledgment

The authors would like to thank all the people who contributed to the Sudoku image collection necessary for this work. The work was also supported by internal grants from the HES-SO, University of Applied Science Western Switzerland.

## Implementation

The C++ implementations of our recognizer[3] and our CDBN library[4] are freely available on-line.

[3] https://github.com/wichtounet/sudoku_recognizer/tree/paper_v2

[4] https://github.com/wichtounet/dbn

## References

[1] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006. [Online]. Available: http://dx.doi.org/10.1162/neco.2006.18.7.1527

[2] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: ACM, 2009, pp. 609–616. [Online]. Available: http://doi.acm.org/10.1145/1553374.1553453

[3] B. Wicht and J. Hennebert, "Camera-based sudoku recognition with deep belief network," in *Soft Computing and Pattern Recognition (SoCPaR), 2014 6th International Conference of*, Aug 2014, pp. 83–88.

[4] A. Van Horn, "Extraction of sudoku puzzles using the hough transform," University of Kansas, Department of Electrical Engineering and Compute Science, Tech. Rep., 2012.

[5] P. Simha, K. Suraj, and T. Ahobala, "Recognition of numbers and position using image processing techniques for solving sudoku puzzles," in *Advances in Engineering, Science and Management (ICAESM), 2012*. IEEE, 2012, pp. 1–5.

[6] S. Impedovo, L. Ottaviano, and S. Occhinegro, "Optical character recognition—a survey," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, no. 01n02, pp. 1–24, 1991.

[7] J. Liang, D. Doermann, and H. Li, "Camera-based analysis of text and documents: a survey," *International Journal of Document Analysis and Recognition (IJDAR)*, vol. 7, no. 2-3, pp. 84–104, 2005.

[8] A. Jain, A. Dubey, R. Gupta, and N. Jain, "Fundamental challenges to mobile based ocr," vol. 2, no. 5, May 2013, pp. 86–101.

[9] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006. [Online]. Available: http://www.ncbi.nlm.nih.gov/sites/entrez?db=pubmed&uid=16873662&cmd=showdetailview&indexed=google

[10] A. Krizhevsky, "Convolutional deep belief networks on cifar-10," 2010.

[11] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, pp. 679–698, Jun. 1986. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.1986.4767851

[12] J. Matas, C. Galambos, and J. Kittler, "Robust detection of lines using the progressive probabilistic hough transform," *Comput. Vis. Image Underst.*, vol. 78, no. 1, pp. 119–137, Apr. 2000. [Online]. Available: http://dx.doi.org/10.1006/cviu.1999.0831

[13] C. Ronse and P. A. Devijver, *Connected Components in Binary Images: The Detection Problem*. New York, NY, USA: John Wiley & Sons, Inc., 1984.

[14] S. Suzuki and K. Abe, "Topological structural analysis of digitized binary images by border following." *Computer Vision, Graphics, and Image Processing*, vol. 30, no. 1, pp. 32–46, 1985. [Online]. Available: http://dblp.uni-trier.de/db/journals/cvgip/cvgip30.html#SuzukiA85

[15] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Comput.*, vol. 14, no. 8, pp. 1771–1800, Aug. 2002. [Online]. Available: http://dx.doi.org/10.1162/089976602760128018

[16] H. Lee, C. Ekanadham, and A. Y. Ng, "Sparse deep belief net model for visual area V2," in *Advances in Neural Information Processing Systems 20*, 2008, pp. 873–880.

[17] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[18] Q. V. Le, J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, and A. Y. Ng, "On optimization methods for deep learning." in *ICML*, L. Getoor and T. Scheffer, Eds. Omnipress, 2011, pp. 265–272. [Online]. Available: http://dblp.uni-trier.de/db/conf/icml/icml2011.html#LeNCLPN11