Forensic Reconstruction of Severely Degraded License Plates

Benedikt Lorch; Friedrich-Alexander University; Erlangen, Germany
Shruti Agarwal, Hany Farid; Dartmouth College; Hanover, NH, USA

Abstract

Forensic investigations often have to contend with extremely low-quality images that can provide critical evidence. Recent work has shown that, although not visually apparent, information can be recovered from such low-resolution and degraded images. We present a CNN-based approach to decipher the contents of low-quality images of license plates. Evaluation on synthetically-generated and real-world images, with resolutions ranging from 10 *to* 60 *pixels in width and signal-to-noise ratios ranging from* −3.0 *to* 20.0 *dB, shows that the proposed approach can localize and extract content from severely degraded images, outperforming human performance and previous approaches.*

Introduction

Although not without controversy, public surveillance cameras for monitoring public areas and traffic have become commonplace. Because of bandwidth and storage limitations, most of these video surveillance systems consist of relatively low-resolution cameras. Additionally, surveillance cameras are typically configured to maximize the field of view, thus further limiting the effective resolution of objects in the video. As a result, footage from public surveillance cameras are often of such low resolution or quality that important identifying details are obscured.

We describe a system for recognizing license plates from extremely degraded images. We evaluate the system's efficacy with respect to a wide range of license plate configurations, resolutions, and degradations.

Conventional methods for automatic license plate recognition divide the task into four stages: image acquisition, license plate extraction, character segmentation, and character recognition [11]. The performance of each stage relies on the robustness of the previous stage [3]. Low-resolution images hamper the ability to isolate individual characters, making it difficult to identify the boundaries between neighboring characters. It seems beneficial, therefore, when dealing with low-resolution images, to combine character segmentation and recognition rather than separating them.

With the recent success of convolutional neural networks (CNN), a variety of segmentation-free recognition methods have emerged. Jaderberg *et al*., for example, presented an end-to-end system for automatic detection and recognition of text in natural-scene images [7]. To transcribe house numbers from street-view imagery, Goodfellow *et al*. presented a CNN architecture capable of recognizing arbitrary digit combinations of bounded length [5]. Špaňhel *et al*. adopted a similar method for the purpose of reading European license plates [12]. Even though we use a similar approach to these previous works, our work focuses on extremely degraded images of U.S. license plates with a large diversity in format.

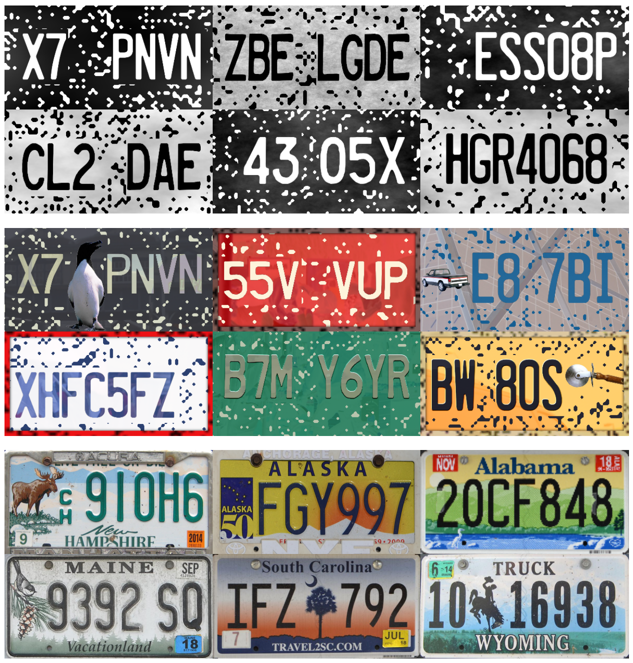

Figure 1. *Synthetically-generated (top/middle) and real-world images (bottom).*

Similar to our work, Hsieh *et al*. presented a method to decipher low-resolution and low-quality license plates [6]. Their approach, however, required explicit knowledge of the character font style and size as well as precise character placement. These assumptions make it challenging to analyze U.S. license plates with their large variation in appearance and format. Building on Hsieh *et al*.'s work, Agarwal *et al*. presented a CNN architecture for deciphering low-resolution and low-quality images of license plates with fewer assumptions regarding font style and size and character placement [1]. Despite showing the ability to generalize to a wider range of formats, this work still assumed a plate with six characters consisting of two groups of three characters separated by a small gap.

Building on this earlier work, we describe a more flexible CNN-based method for deciphering license plates. This approach places fewer limitations on the configuration of the license plate. We only consider license plates containing five to seven characters, but place no other constraints on the character format or placement. We evaluate our approach's efficacy on extremely degraded images with a resolution ranging in size between 12×6 and 55×27.5 pixels and with a signal-to-noise ratio (SNR) rang-

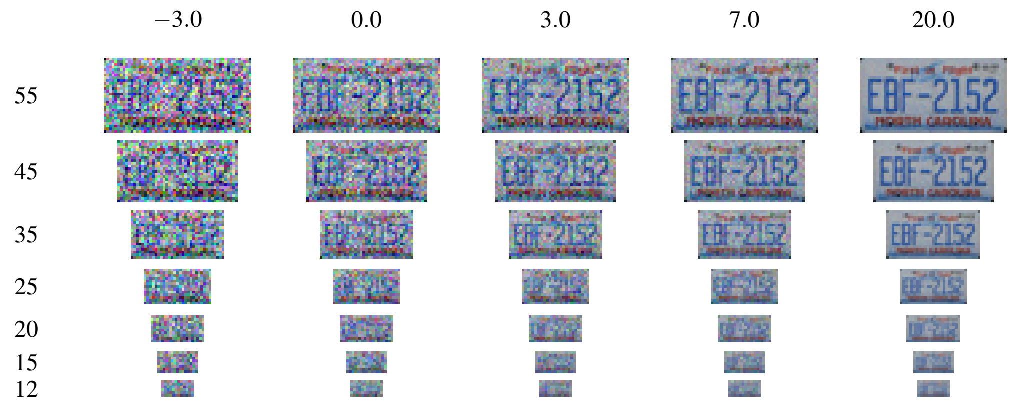

Figure 2. An example license plate (from North Carolina) at different degradation levels. The rows correspond to the image width (pixels) and the columns correspond to the amount of additive Gaussian noise (SNR in dB).

ing between −3.0 and 20.0 dB (at this low resolution the width of a single character ranges between 1.3 and 6.0 pixels). We assume that an investigator corrects the image in question for perspective distortion (although this need not be perfect) and that the image is tightly cropped at the boundary of the license plate.

Datasets

Since we were not aware of any large-scale data sets of labeled license plate images, we created and collected a large set of synthetically-generated and real-world images for the purpose of training our CNN.

Synthetic

The first data set was created similar to that described in [1]. We measured the font size and character placement from 259 real license plates containing between five and seven characters. These measurements included the character width and height, the character offset from the top left of the plate, and the position and width of any gap between characters. Alpha-numeric strings of length five to seven were randomly generated and rendered with character size and spacing drawn from these distributions of real-world measurements. The characters were rendered in one of four typical license plate fonts and with a randomly selected brightness ranging from black to white set against a random grayscale background selected to ensure a minimum contrast. The background was speckled with random patterns to make it more likely that the CNN would generalize to real-world images. The grayscale images were synthesized at a resolution of 400×200 pixels. Six representative synthesized plates are shown in the top portion of Figure 1.

In early experiments we found that although these synthetically-generated images allowed for some degree of generalization to real-world images, accuracy dropped for many plates with more unusual formats shown in the bottom portion of Figure 1. To this end, we generated a second set of images containing a variety of new more realistic and varied features. These features include: (1) Addition of randomly selected objects drawn from the Hemera Photo Objects collection [2]; (2) Addition of embossing to the characters to simulate how most U.S. plates are created. This embossing introduces realistic shading at the edges of the characters; (3) Addition of a license plate frame that is often seen in real-world images; (4) The images were rendered in color where the colors were extracted from randomly selected real-world license plates, similar to [7]; and (5) The uniformly colored background, characters, and frame were blended with random crops from images downloaded from Flickr [8] to simulate texture. The final color 400×200 pixel images were rendered by randomly toggling the visibility of each these effects. Six representative synthesized plates are shown in the middle portion of Figure 1.

Real-world

We downloaded 6,196 images from plateshack.com, a site which hosts a large and diverse collection of license plate images. These images were drawn from the *y2k* subcategory containing license plates issued after the year 2000. Each image was tightly cropped around the license plate frame and contained no perspective distortion. We manually reviewed all the downloaded images, retaining plates with five to seven characters and removing low-quality images (blurry or in which the characters were somehow obscured).

We also collected 1,771 photographs of real license plates by walking through a variety of parking lots in several different U.S. states. Each image was again tightly cropped around the license plate and any residual perspective distortion was manually removed [4].

Degradation

To simulate low-resolution and low-quality images, the synthetic and real-world images described above were subjected to various levels of image degradation. Each image was downsampled from their original width of 400 pixels to a width ranging from 60 to 10 pixels. Each image was also corrupted with additive zero-mean Gaussian noise yielding images with an SNR in the range of −3.0 to 20.0 dB. Testing images were degraded to a fixed set of resolutions and noise levels within these ranges. Shown in Figure 2 is a representative example at each testing resolution and SNR.

CNN Architecture

Agarwal *et al.* have shown good results in deciphering highly degraded license plates containing exactly six characters [1]. Because their approach explicitly assumed that the license plate consisted of two sets of three characters (with a small gap separating them), it is not straightforward to extend their CNN architecture to work with license plates of variable lengths. In contrast, Goodfellow *et al.* demonstrated how to transcribe house numbers, from Google Street View images, with a varying but bounded number of characters [5]. We merge certain aspects of Goodfellow *et al.*'s CNN architecture with that of Agarwal *et al.*'s.

Our proposed CNN architecture consists of eight convolutional layers and five max-pooling layers. The convolutional layers use kernels of size 3×3 and padding of one pixel. The number of filter kernels are $\{64, 64, 128, 128, 256, 256, 512, 512\}$. The max-pooling layers follow the second, fourth, sixth, seventh, and eighth convolutional layer. Every odd max-pooling layer reduces the spatial size by two. The last pooling layer is followed by two fully-connected layers consisting of 1024 and 2048 units, respectively. The output of the second fully-connected layer is fed into seven output layers, each of which is a fully-connected layer with 37 units followed by softmax activation. All convolutional layers, and the first two fully-connected layers, use ReLU activation. Dropout is applied after the activation function of the first two fully-connected layers. As suggested in [13], the dropout probability is set to 0.5. Xavier weight initialization is used for all the convolutional layers and the seven output layers. The first two fully-connected layers are initialized using a truncated normal distribution with zero-mean and 0.005 standard deviation. The biases in the convolutional layers, the first two fully-connected layers, and the seven output layers are initialized with a fixed value of 0.1, 0.1, and 0.

We used the sum of cross-entropy losses of all seven output layers as cost function and stochastic minibatch gradient descent with a batch size of 32 and a learning rate of 0.01. Training was stopped after the accuracy on the validation set had not improved for 100k iterations. Our CNN was implemented using TensorFlow v1.4 and all experiments were performed on a GeForce GTX 1080 Ti GPU.

Our network has seven output layers, each of which corresponds to one character, thus this architecture can read license numbers up to length seven. Each output layer consists of 37 units corresponding to 26 possible letters, 10 possible digits, and a special *null character* "$\diamond$". Similar to [12], we introduced the null character to allow for variable length plates. For example, the five-character plate "ABC12" is represented as "ABC12$\diamond\diamond$", the six-character plate "ABC123" is represented as "ABC123$\diamond$", and the seven-character plate "ABC1234" is represented as "ABC1234". The predictions of each output layer can be interpreted as a discrete probability distribution over the 37 possible characters.

Evaluation

In contrast to Goodfellow *et al.*, who do not award any credit for partially correct house numbers [5], even a partial match, combined with additional information like the car make and model, can be extremely helpful in narrowing the list of suspects in forensic investigations. Therefore, it is beneficial to evaluate the accuracy of our model on a per-character basis, awarding partial credit for correctly identifying individual characters. We describe two such metrics that we use to evaluate our system.

Because we are contending with variable-length license plates with five to seven characters, we pad the ground truth labels with null characters ($\diamond$) to a fixed length of seven. The ground truth and the predicted output seven-character strings are then compared character-by-character. We report both the "Top-1" and "Top-5" accuracy. The Top-1 category corresponds to the case when the correct character is the most likely predicted character. The Top-5 category corresponds to the case when the correct character is ranked in the top five of the most likely predicted characters. This Top-5 metric is particularly useful in light of the often confused "0" and "O" or "1" and "I".

One drawback of this evaluation metric is that, for example, a prediction of "XABC123" will be scored as $0/7$ against the string "ABC123$\diamond$", despite the prediction being wrong in effectively only one character. To this end, we also evaluate our system based on the Levenshtein distance [9]. Given two strings, the Levenshtein distance counts the minimum number of characters to insert, delete, and substitute to transform one string into the other.

Results

Shown in Table 1 are the results from three separate experiments corresponding to training and fine-tuning using the datasets described above. These accuracies correspond to the testing accuracy against 1,000 real-world images of our recording and not previously seen by the network during training.

Synthetic-I

In the first experiment, we trained our CNN on 10M synthetic grayscale images (as shown in the top portion of Figure 1) with 2k images used for validation. The CNN was then tested on 1k real-world images (converted to grayscale) from our own collection. Shown in the top portion of Table 1 are the Top-1 and Top-5 accuracies (%) and the Levenshtein distances as a function of image resolution and noise level. As expected, the accuracy decreases at lower resolutions and with higher amounts of noise. The most precipitous drop in accuracy occurs for resolution below 25 pixels (or approximately 3 pixels per character for a seven-character plate). Chance prediction for the Top-1 metric is $1/37 = 2.7\%$ while chance prediction for the Levenshtein distance (averaged over five to seven character long strings with the same distribution as license numbers in the testing set) is 6.5. Although not very high, accuracies at the lowest resolution and highest noise level are still above chance. The significant improvement between the Top-1 and Top-5 categories is due to the disambiguation of easily confused characters like "0" and "O" or "1" and "I". For example, at a resolution of 25 pixels and with an SNR of 3.0 dB, the Top-1 accuracy is 39.8% as compared to a Top-5 accuracy of 67.5%.

Synthetic-II

In the above experiment there was a significant gap between the accuracy on synthetically-generated testing images (complete results not included) and the real-world testing images. At a resolution of 25 pixels and with an SNR of 3.0 dB, for example, the Top-1 and Top-5 accuracies on unseen synthetically-generated images are 82.8% and 98.1%, with a Levenshtein distance of 1.2.

		(a) Top-1					(b) Top-5					(c) Levenshtein distance				
	width	SNR (dB)					SNR (dB)					SNR (dB)				
	(pixels)	-3.0	0.0	3.0	7.0	20.0	-3.0	0.0	3.0	7.0	20.0	-3.0	0.0	3.0	7.0	20.0
Synthetic-I	55	71.3	80.1	84.0	85.7	85.8	91.0	95.1	96.2	96.4	96.1	1.8	1.2	1.0	0.8	0.8
	45	60.8	74.8	80.6	83.3	85.0	85.0	92.3	95.1	95.6	95.9	2.5	1.6	1.2	1.0	0.9
	35	43.8	57.8	68.1	74.8	79.0	71.5	83.1	89.3	92.0	93.4	3.7	2.7	2.0	1.6	1.3
	25	21.3	30.9	39.8	50.3	59.9	46.0	57.5	67.5	76.5	83.6	5.3	4.7	4.0	3.3	2.5
	20	15.2	19.1	24.6	31.3	40.6	35.3	43.6	51.3	59.4	69.4	5.8	5.5	5.1	4.6	3.9
	15	11.1	12.6	14.3	16.5	19.9	26.9	29.9	34.5	38.8	44.3	6.1	6.0	5.9	5.7	5.5
	12	10.5	11.3	11.9	13.0	14.4	23.9	26.2	28.2	31.3	35.1	6.2	6.1	6.0	6.0	5.9
Synthetic-II	55	76.8	83.1	85.6	87.1	88.1	93.8	96.0	97.0	97.4	97.5	1.5	1.0	0.8	0.7	0.7
	45	69.6	80.1	84.8	86.9	88.1	90.2	95.1	96.6	97.1	97.6	2.0	1.2	0.9	0.8	0.7
	35	53.8	68.3	77.2	83.5	86.5	80.5	89.8	94.2	95.9	96.7	3.1	2.1	1.5	1.0	0.8
	25	29.3	42.5	55.1	66.9	77.8	56.7	70.1	80.9	88.5	93.7	4.8	3.9	3.0	2.2	1.4
	20	19.6	27.3	37.3	47.0	60.5	43.6	55.0	65.7	74.9	84.6	5.5	5.0	4.3	3.6	2.6
	15	12.5	15.2	19.1	22.9	31.4	30.1	36.2	42.4	48.9	58.4	6.0	5.8	5.6	5.3	4.7
	12	10.5	12.2	13.6	16.3	19.9	26.9	30.6	33.2	39.4	46.3	6.2	6.1	6.0	5.8	5.5
Real-world	55	89.2	93.9	95.5	96.0	96.2	98.4	99.3	99.6	99.6	99.7	0.7	0.4	0.3	0.2	0.2
	45	82.1	91.8	94.9	95.8	96.3	96.5	98.9	99.5	99.6	99.7	1.2	0.5	0.3	0.3	0.2
	35	67.8	82.7	90.9	94.3	95.6	90.6	96.7	98.9	99.4	99.6	2.2	1.1	0.6	0.3	0.3
	25	40.3	57.8	72.9	84.7	92.2	73.0	85.9	93.3	97.3	99.1	4.1	2.8	1.8	1.0	0.5
	20	26.4	38.8	52.5	65.9	81.8	60.3	72.0	83.3	90.8	96.5	5.0	4.1	3.2	2.3	1.2
	15	16.7	21.5	27.9	38.2	52.9	46.7	53.8	62.2	72.2	83.2	5.6	5.3	4.9	4.2	3.2
	12	13.5	15.9	18.8	24.3	32.9	41.0	45.8	51.0	58.4	68.8	5.8	5.6	5.4	5.1	4.6

Table 1. *Prediction accuracy for real-world testing images reported as (a)-(b) per-character accuracy (%) where chance prediction is* 2.7%*; and (c) Levenshtein distance (number of insertions, deletions, and substitutions needed to transform the predicted string to correct string) where chance prediction is* 6.5*. The rows and columns correspond to the accuracy for images of varying resolution (pixels) and noise level (dB). The accuracies/distances, from top to bottom, correspond to training on the two sets of synthetically-generated images, and fine-tuning with real-world images (see Figure 1).*

These accuracies drop to 39.8% and 67.5% and 4.0 on real-world images. To close this gap, we retrained the network on the more realistic synthetic color images (middle portion of Figure 1). As before, we generated 10M training images with 2k images used for validation.

Shown in the middle portion of Table 1 are the Top-1 and Top-5 accuracies and the Levenshtein distances. Accuracy improved across almost all resolutions and noise levels. For example, at a resolution of 25 pixels and with an SNR of 3.0 dB, the Top-1 accuracy increased from 39.8% to 55.1%, the Top-5 accuracy increased from 67.5% to 80.9%, and the Levenshtein distance decreased from 4.0 to 3.0.

Real-world

In this third and final experiment we fine-tuned the CNN trained in the previous section with the real-world images downloaded from plateshack. Specifically, we randomly selected one of the plateshack images (with replacement), degraded it to the same range of resolutions and noise levels as before, and repeated these steps to obtain 1M degraded images used as training data. Similarly, a validation set of 10k images was created from the 771 remaining real-world images of our collection which were not used for testing. Starting with the CNN from the previous section, these training/validation images were used to fine-tune the network with a reduced learning rate of 0.0001.

As shown in the bottom portion of Table 1, this fine-tuning improved accuracy for all resolutions and noise levels. For example, at a resolution of 25 pixels and with an SNR of 3.0 dB, the Top-1 accuracy increased from 55.1% to 72.9%, the Top-5 accuracy increased from 80.9% to 93.3%, and the Levenshtein distance decreased from 3.0 to 1.8. In addition, accuracies at the lowest resolution of 12 pixels increased significantly.

The continued improvement in performance with increased realism of training and validation images shows the importance of the selection of the training images and provides hope that even further dataset refinements will lead to even better recognition rates.

Comparison

Our fine-tuned CNN outperforms human performance on a similar recognition task. In particular, as compared to the human detection accuracies reported in Agarwal *et al.* [1], our Top-1 detection accuracy is more accurate at nearly all resolutions and noise levels. For example, at a resolution of 25 pixels and with an SNR of 3.0 dB, our accuracy is 72.9% as compared to the human's 33.3%. Similarly, at a resolution of 15 pixels and with an SNR of -3.0 dB, our accuracy is 16.7% as compared to the human's near chance performance of 2.8%.

Our fine-tuned CNN also outperforms the approach of Agarwal *et al.* In order to facilitate a direct comparison between the six-character network of Agarwal *et al.* and our more format-flexible network, we limited our comparison to 409 six-character real-world license plates of our collection. In making this comparison, we consider the outputs of only the first six characters (this

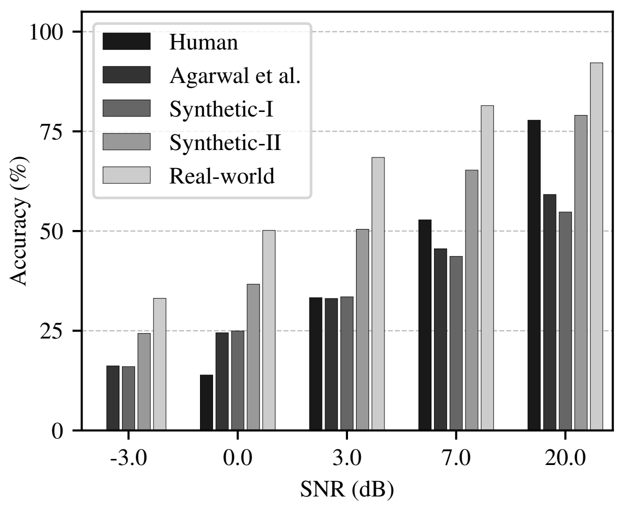

Figure 3. *Comparison of human accuracy and Agarwal* et al. *to our CNN trained on synthetically-generated images (Synthetic-I, Synthetic-II) and fine-tuned on real-world images. All test images contained six characters plates and had a horizontal resolution of* 25 *pixels.*

slightly reduces the overall accuracies of our network because the network is not rewarded for correctly classifying the seventh character as a null character). As shown in Figure 3, our Synthetic-I network performs on par with Agarwal *et al.* Our Synthetic-II and Real-world networks, however, significantly outperform Agarwal *et al.* (and humans) at all resolutions and noise levels. For example, at a resolution of 25 pixels and with an SNR of 3.0 dB, Agarwal *et al.*'s network achieves an accuracy of 33.1% while our Synthetic-II and Real-world networks achieve an accuracy of 50.4% and 68.5%, respectively.

Analysis

Despite recent efforts towards interpreting the representations learned by a neural network [10, 14], it still remains difficult to fully comprehend how or why deep-learning methods work as well as they do. To this end, we provide two analyses that indirectly suggest that the network is in fact performing the recognition task in a meaningful way.

Shown in Figure 4 is the frequency with which each alpha-numeric character is recognized and confused with other characters. The alpha-numeric characters are sorted in increasing level of overall accuracy with "O" being the least accurate character. The fact that similarly appearing characters are most frequently confused gives us confidence that the network is using meaningful information to perform the recognition task. The letter "O", for example, is most often confused with the visually similar letters "0", "B", and "D", and the letter "Q" is most often confused with "0", "D", and "6".

To further support the hypothesis that our CNN learned a meaningful representation, we analyzed which parts of the input image the CNN relies on to predict each character position. Specifically, we occluded a part of the license plate by placing a uniform gray patch over each character position (see, e.g., [14]). This process was repeated for each character position. The CNN failed to recognize the character when it was occluded, which indicates that the network was not latching onto some secondary or tertiary artifact to perform recognition.

Conclusion

Despite not being visually apparent, highly-degraded images of license plates contain distinctive information. We have shown that with the proper training dataset, a CNN can accurately decipher even the most degraded images of license plates, significantly outperforming human recognition rates.

As compared to previous work, our network can decipher a wider range of license plate formats, from five to seven characters and with a wide range of configurations. Our network, however, would still benefit from even more flexibility to contend with, in the U.S. at least, an enormous range of formats including vanity plates. Given the continued improvement that we saw with more realistic training data, we expect that an increasingly diverse dataset will lead to more flexibility and even higher recognition rates. In future work we would like to investigate more sources of degradation, such as perspective distortion and motion blur.

Images of license plates are, of course, not the only type of content of interest for investigators. Our expectation is that similar approaches to that described here can be used to, for example, recognize people in highly degraded images.

Acknowledgments

The authors would like to thank Christian Riess for his constructive feedback.

References

[1] S. Agarwal, D. Tran, L. Torresani, and H. Farid, "Deciphering severely degraded license plates," *Electronic Imaging*, vol. 2017, no. 7, pp. 138–143, Jan. 2017.

[2] M. J. Bravo and H. Farid, "Object recognition in dense clutter," *Perception & Psychophysics*, vol. 68, no. 6, pp. 911–918, Aug. 2006.

[3] S. Du, M. Ibrahim, M. Shehata, and W. Badawy, "Automatic license plate recognition (ALPR): A state-of-the-art review," *IEEE Transactions on Ciruits and Systems for Video Technology*, vol. 23, no. 2, pp. 311–325, Feb. 2013.

[4] H. Farid, *Photo Forensics*. The MIT Press, 2016.

[5] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnoud, and V. D. Shet, "Multi-digit number recognition from street view imagery using deep convolutional neural networks," *ArXiv e-prints*, Dec. 2013. arXiv: `1312.6082 [cs.CV]`.

[6] P.-L. Hsieh, Y.-M. Liang, and H.-Y. M. Liao, "Recognition of blurred license plate images," in *2010 IEEE International Workshop on Information Forensics and Security*, IEEE, Dec. 2010.

[7] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Reading text in the wild with convolutional neural networks," *International Journal of Computer Vision*, vol. 116, no. 1, pp. 1–20, May 2015.

[8] E. Kee, M. K. Johnson, and H. Farid, "Digital image authentication from JPEG headers," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 1066–1075, Sep. 2011.

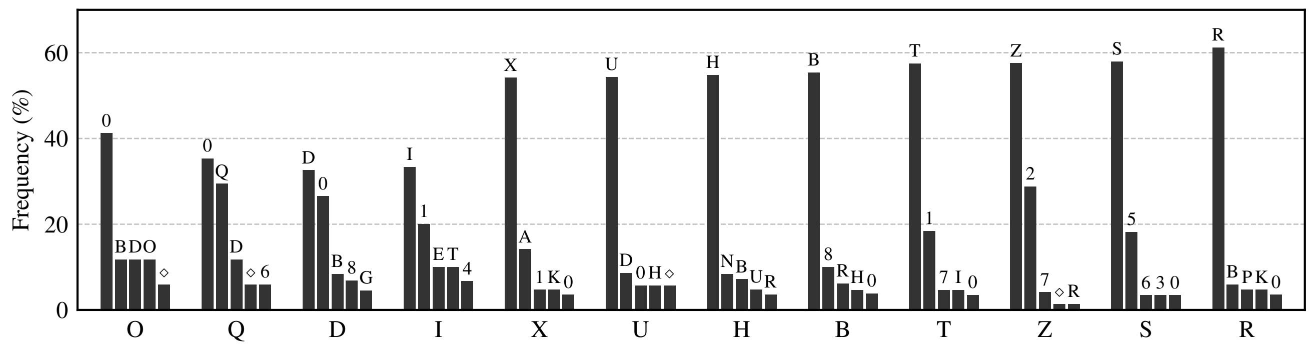

Figure 4. *Each group of five bars corresponds to a subset of the* 36 *alpha-numeric characters (ranked in order of least to most accurate). The height of each bar corresponds to the frequency with which the corresponding character was recognized (at a resolution of* 25 *pixels and an SNR of* 3.0 dB*). The character "O", for example, was recognized as "0" about* 40% *of the time followed by "B", "D" and "O" with equal frequency, and occasionally as the null character "◇".*

[9] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Soviet Physics Doklady*, vol. 10, p. 707, Feb. 1966.

[10] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 5188–5196.

[11] M. Sarfraz, M. J. Ahmed, and S. A. Ghazi, "Saudi Arabian license plate recognition system," in *2003 International Conference on Geometric Modeling and Graphics, 2003. Proceedings*, Jul. 2003, pp. 36–41.

[12] J. Špaňhel, J. Sochor, R. Juránek, A. Herout, L. Maršík, and P. Zemšík, "Holistic recognition of low quality license plates by CNN using track annotated data," in *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Aug. 2017, pp. 1–6.

[13] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European Conference on Computer Vision (ECCV)*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., Sep. 2014, pp. 818–833.

Author Biography

Benedikt Lorch *is Ph.D. student in the Security Research Group at the Friedrich-Alexander University (FAU). He received his B.S. and M.S. degree in Computer Science from the FAU in 2016 and 2018, respectively. During his studies he was a visiting student at Imperial College London and Dartmouth College. His research interests focus on image forensics, image processing, and computer vision.*

Shruti Agarwal *is Ph.D. student in the Department of Computer Science at Dartmouth College. Previously, she worked as a software developer in Adobe Illustrator team at Adobe, India. She received her M.S. and B.S. degree in Computer Science from Indian Institute of Technology (IIT) Delhi, India, and Harcourt Butler Technology Institute (HBTI), India, respectively. Her primary research interests lies in image processing, computer vision, and graphics.*

Hany Farid *is the Albert Bradley 1915 Third Century Professor of Computer Science at Dartmouth. His research focuses on digital forensics, image analysis, and human perception. He received his under-graduate degree in Computer Science and Applied Mathematics from the University of Rochester in 1989 and Ph.D. in Computer Science from the University of Pennsylvania in 1997. Following a two year post-doctoral fellowship in Brain and Cognitive Sciences at MIT, he joined the faculty at Dartmouth in 1999. He is the recipient of a Alfred P. Sloan Fellowship, a John Simon Guggenheim Fellowship, and is a fellow of the National Academy of Inventors.*