https://doi.org/10.2352/ISSN.2470-1173.2017.7.MWSF-337

Deciphering Severely Degraded License Plates

Shruti Agarwal, Du Tran, Lorenzo Torresani, Hany Farid; Dartmouth College; Hanover, NH, USA

Abstract

Extremely low-quality images, on the order of 20 *pixels in width, appear with frustrating frequency in many forensic investigations. Even advanced de-noising and super-resolution technologies are unable to extract useful information from such low-quality images. We show, however, that useful information is present in such highly degraded images. We also show that convolutional neural networks can be trained to decipher the contents of highly degraded images of license plates, and that these networks significantly outperform human observers.*

Introduction

Recognizing text in images is a well-studied problem [1]. Text recognition can either be done by recognizing individual characters or by recognizing the full word. For degraded images, however, it is difficult to localize and recognize individual characters in an image. Word recognition, therefore, has become central to text recognition in degraded images. Deep convolutional neural networks [2] have been used for text recognition in natural images. Goodfellow et al. [3] used a deep neural network to localize, segment and recognize multiple digits on street view images. Jaderberg et al. [4] also proposed an end to end text recognition system for natural images using a deep neural network. Jaderberg et al. used a large word dictionary and formulated the text recognition task as a large scale classification problem. Recently, Svoboda et al. [5] used CNN to remove motion blur from images of license plate blurred with a blur kernel of various directions and lengths. Although this approach is able to deblur highly blurred images, it does not contend with extremely low resolution and noisy images.

Unlike much of this previous work we focus on extracting text from highly degraded images on the scale of only a few pixels per character. Hsieh et al. [6] were the first to show that information can be extracted from highly degraded license plates. In this work the authors assume a known font type, font size, and character layout, and assume that the degraded image is blurry and perspectively distorted, but does not necessarily contain additive noise. Although the authors only show results on a small set of images, they do show that information is present in license plates as small as 20 pixels in width. Building on these ideas, in this paper we propose to train a CNN for recognizing highly degraded license plates with an unknown background template, font type, size and character location. We also explicitly work in the presence of high amounts of additive noise – a common occurrence in real-world imagery.

Recognition by Human Observers

We begin by performing a perceptual study to determine how well human observers can decipher degraded license plates. This study provides a baseline against which to compare our computational approaches. Observers were shown images of synthetically generated license plates with 7 characters and asked to determine the identity of a single randomly selected character. Observers saw license plates ranging in width from 15 to 55 pixels (corresponding to approximately 1.9 to 6.9 pixels per character) with signal-to-noise ratios (SNR) ranging from -3.0 to 20.0 db, Figure 3.

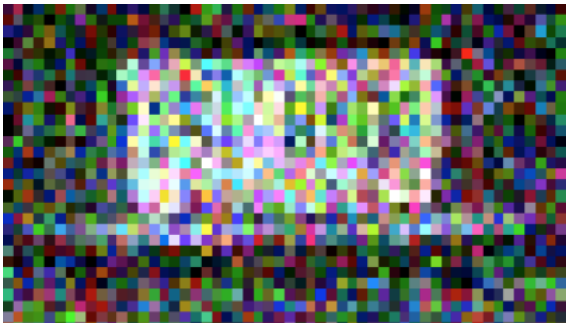

Figure 1. *An example of the type of degraded license plate that we seek to decipher.*

The average accuracy with which 12 observers were able to correctly identify a character is shown in Figure 2. Observers are fairly accurate at resolutions of 35 pixels and larger with SNR greater than 3.0 db. Observer performance falls precipitously at a resolution less than 25 pixels, almost regardless of SNR.

width	noise (SNR)				
(pixels)	-3.0	0.0	3.0	7.0	20.0
55	52.8	88.9	97.2	100.0	97.2
45	50.0	75.0	86.1	91.7	100.0
35	33.3	63.9	80.6	80.6	97.2
25	0.0	13.9	33.3	52.8	77.8
15	2.8	2.8	5.6	11.1	2.8
12	-	-	-	-	-

Figure 2. *Human accuracy (in percent) of identifying a single character in a* 7*-character license plate. Chance performance is* $1/36 = 2.8\%$.

Recognition using Correlation

Why were our observers not able to decipher license plates below a certain resolution and signal-to-noise ratio? Is there information in the degraded images that observers cannot extract or does the degradation destroy any distinct information? To find out, we performed large-scale simulations to determine if distinctive information survives severe image degradation.

We synthesized a target image of a license plate with 7 characters and degraded this image to a resolution ranging in width from 12 to 55 pixels (corresponding to approximately 1.5 to 6.9

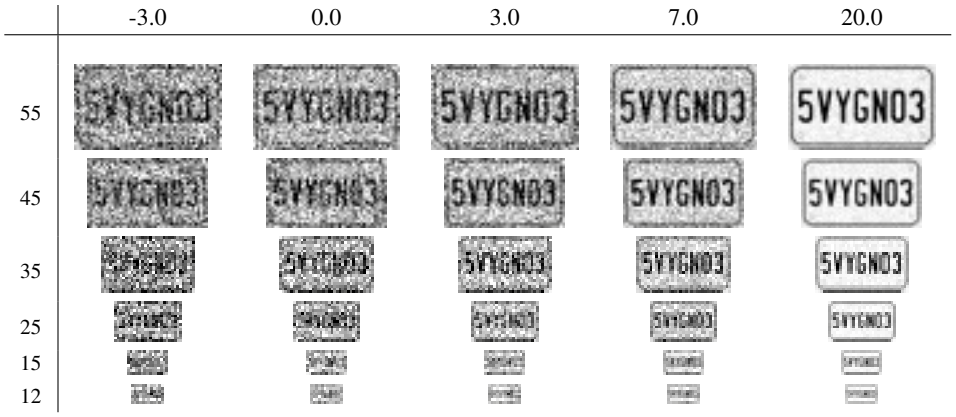

Figure 3. *Shown from top to bottom are images of decreasing resolution (in pixels) and shown from left to right are images with increasing SNR (in db).*

pixels per character) with SNR ranging from -3.0 to 20.0 db, Figure 3. We then synthesized 36 test images in which one of 7 character positions takes on each of 36 possible values (26 alphabetic or 10 numeric characters), and the remaining characters are the same as in the target image. The location and size of this character was isolated in each of the 36 test images and compared–using a 2-D correlation–to the target image. The character with the largest correlation was taken to be the identity of the character in the target image. The classification accuracy, averaged over $20,000$ randomly generated images, is shown in Figure 4. These results show that information is present in highly degraded images that our human observers (Figure 2) were unable to extract. For example, at resolution of 25 pixels and a noise level of 3.0 db, the correlation accuracy is 53.8% as compared to the human's 33.3%. Similarly, at resolution of 15 pixels and a noise level of 20.0 db, the correlation accuracy is 98.4% as compared to the human's 2.8%.

This correlation-based approach, however, assumes perfect knowledge of font type and size, character alignment and background template. We next explore the performance of a convolutional neural network (CNN) in the absence of this assumed knowledge.

width	noise (SNR)				
(pixels)	-3.0	0.0	3.0	7.0	20.0
55	60.9	84.1	95.2	99.0	100.0
45	49.2	74.9	91.6	98.1	100.0
35	33.9	59.2	81.8	95.4	100.0
25	16.4	31.2	53.8	82.5	100.0
15	5.7	8.8	15.3	32.1	98.4
12	5.0	7.1	11.1	22.2	88.7

Figure 4. *Correlation-based accuracy (in percent) of identifying a single character in a 7-character license plate. Chance performance is* $1/36 = 2.8\%$.

Recognition using Deep Learning

There are two basic approaches to recognize license plates. In the first approach, each character is individually isolated in the image and recognized. This approach has the drawback that it can be difficult to localize individual characters in extremely low resolution images, and by considering only single characters the interaction of neighboring characters at low resolutions is ignored. In the second approach, all characters are considered and recognized as a single entity. This approach does not suffer from the shortcomings of the first approach, but needs a prohibitively large training dataset (e.g., there are $36^6 \approx 2$ billion unique 6-character license plates). We therefore adopt a hybrid approach in which, using the whole image as input, we recognize the license plate in two parts. For simplicity of exposition, we will begin by assuming a license plate with 6 characters and separately recognize the first 3 and the second 3 characters (this basic approach will generalize to an arbitrary number of characters).

We describe the training of the first set of three characters; the training for the second set of three characters is identical. The training set consists of 600 variants of each of $36^3 = 46,656$ alpha-numeric character combinations in the first three character positions (with a random set of characters in the fourth through sixth positions), for a total of $27,993,600$ training images. The license plates were constructed using characters from one of five different font styles. The characters were then scaled to a random aspect ratio ranging from 0.4 to 0.6 and a width of 42 to 52 pixels. The contrast of the characters relative to the background was also chosen randomly and uniformly in the range of 0.15 to 1.0. These characters were then placed atop a background of random color and random texture meant to simulate the markings on real-world license plates, Figure 5 (top panel). The characters were arranged on the background with a random gap between the third and fourth character position. The width of the gap was selected uniformly from a range of 6 to 90 pixels. All of these values were selected based on qualitative measurements taken from real-world license plates.

The training images, synthesized at a resolution of 400×200 pixels, were then degraded to a resolution ranging in width from 12 to 55 pixels (corresponding to approximately 1.5 to 6.9 pixels per character) with SNR ranging from -3.0 to 20.0 db. Each degraded image was then resized to a resolution of 100×50 pixels. These images are the input to the CNN.

Figure 5. *Synthetic training images (top), synthetic testing images (middle), and real-world photographic test images (bottom).*

CNN Architecture

The inputs to our CNN are synthetically generated grayscale images of size 100×50 pixels. Our CNN has the following architecture: 8 convolutional layers, 3 fully connected layers, followed by one layer containing 3 separate softmax function (one per character). Each softmax function has 36 outputs corresponding to a probability distribution over 36 alpha-numeric characters. One softmax function predicts the identity of one character in the license plate, yielding a total of 36×3 outputs.

The kernel size and number of filters used in each of the 8 convolution layers were: $(3,64)$, $(3,64)$, $(3,128)$, $(3,128)$, $(3,256)$, $(3,256)$, $(3,512)$, $(3,512)$. Small size kernels were used as suggested by Simonyan et al. [7] for better convergence during training. The input to all convolution layers was spatially padded by 1 pixel to avoid any reduction in input size after convolution. A fixed stride of 1 was used in all convolutional layers. Spatial pooling was done using 5 max pooling layers after the second, fourth, sixth, seventh, and eight convolutional layer. The size of the image was reduced three times by using a stride of 2 in every other max pooling layer. The size of the first two fully connected layers were 1024 and 2048. A dropout ratio of 0.5 was used after the first two fully connected layers. A mini-batch of size 250 was used with momentum set to 0.9. In addition to the dropout regularisation used with the first two fully connected layers, training was also regularised by a weight decay of 0.0005. A fixed learning rate of 0.01 was used throughout the training. We stopped the training after 20K iterations (107 epochs) and the size of the epoch is $36^3 = 46,656$. Our CNN was implemented using the Caffe toolbox. [8]

Experiments and Results

We tested our CNN on $20,000$ synthetically generated images and on 132 perspectively corrected real-world images. The synthetic test images were generated in a similar fashion but superimposed atop actual license plate templates, Figure 5 (middle panel). This more realistic background was used to more closely model real-world images, but were not used in training to avoid over-fitting to a particular license plate template. The real-word images were from states that contain 6 characters, Figure 5 (bottom panel).

The probability of any 3-character combination can be computed from the output of the 3 softmax functions. We can, therefore, rank order all 3 character combinations in order of likelihood. Shown in the left column of Figure 6 is the accuracy that the correct 3 characters is the top-rated combination (averaged over two CNNs trained to recognize the first 3 and last 3 characters). Also shown in Figure 6 are the accuracies with which the correct 3 characters appeared in the top 5 and top 10 most likely combinations.

Note that, unlike the previous results, where chance performance is $1/36 = 2.8\%$, chance performance is now $1/(36^3) = 0.002\%$. To make a direct comparison between the human observers (Figure 2) and correlation (Figure 4) we will assume that the accuracy of detecting individual characters is statistically independent. Under this assumption, at a resolution of 25 pixels and with an SNR of 3.0 db, human performance is 3.6%, correlation-based is 15.7%, and the CNN is 42.9% (training), 40.9% (testing, synthetic), and 31.0% (testing, real), Figure 7. At the highest-resolution (55 pixels) and lowest-noise (20.0 db) humans and correlation perform approximately as well as the CNN. In the critical cases with low-resolution and high-noise levels, the CNN performs much better and has the advantage that it makes no assumptions about the font style, size, or character layout. Finally, we note that the accuracy for the synthetic testing condition is better than the real testing condition. This suggests that we can improve the training to better generalize to real-world images.

Show in Figure 8 is the nature of the mistakes made by the CNN. In particular, for each character, we computed the top-5 most common matched characters at a resolution of 25 pixels and an SNR of 3.0 db. For example, the number "0" was correctly matched 19.8% while it was confused with a "O" 26.7%, a "D" 17.6%, a "U" 11.1%, and a "Q" 10.9%. These mistakes are intuitive given the common structure of these characters. We find that throughout each character, the mistakes follow a similar pattern that results from a common character structure.

Comparison to SVM

It is natural to wonder if more standard machine learning approaches would yield similar results to the CNN. We therefore trained a non-linear support vector machine (SMV) [9, 10] on a similar task as described above.

To simplify the task, we trained an SVM to recognize a single character. All images were down sampled to a fixed resolution of 25 pixels and with each of 5 SNR levels. For each of 36 characters, we created 600 images with random neighboring characters, backgrounds, font size, font type, and contrast, yielding a total of $21,600$ images. Each image was then degraded by 1 of 5 noise levels yielding $108,000$ training images. Testing of the SVM was performed on the same $20,000$ test images described in Section . The single character to be recognized was isolated and extracted from the training and testing images. While this clearly simplifies the task as compared to the CNN training, the SVM still had to

		Top 1					Top 5					Top 10				
	width	noise (SNR)					noise (SNR)					noise (SNR)				
	(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
(a)	55	61.8	82.0	89.4	91.7	92.8	90.8	98.8	99.7	99.9	100.0	92.0	99.2	99.9	100.0	100.0
	45	43.6	72.4	86.0	90.4	92.7	74.7	95.5	99.4	99.8	100.0	78.5	97.1	99.8	100.0	100.0
	35	20.8	52.5	76.1	87.7	92.5	45.8	83.8	96.5	99.6	100.0	50.4	87.0	98.0	99.8	100.0
	25	4.3	18.0	42.9	69.4	89.3	12.5	40.1	73.6	94.4	99.7	15.6	46.0	78.5	95.9	99.8
	15	0.3	0.8	2.5	8.2	35.7	0.9	2.8	7.6	22.7	66.9	1.2	3.5	9.7	25.8	68.0
	12	0.0	0.1	0.7	1.5	5.1	0.3	0.6	2.0	4.9	15.2	0.4	1.1	2.6	6.3	19.0
(b)	55	64.2	83.9	89.7	92.2	93.8	91.7	99.1	99.7	99.9	100.0	96.1	99.8	100.0	100.0	100.0
	45	44.5	75.5	87.2	91.7	93.7	76.3	97.0	99.6	99.8	100.0	84.9	99.0	100.0	100.0	100.0
	35	18.3	53.2	77.5	88.7	93.1	41.4	83.8	97.7	99.8	100.0	51.6	90.8	99.3	100.0	100.0
	25	2.6	14.8	40.9	71.0	88.6	8.5	35.0	73.0	95.8	99.8	12.8	45.2	82.3	98.4	100.0
	15	0.1	0.3	1.1	4.2	31.1	0.3	1.1	3.7	12.6	60.4	0.7	2.0	5.8	18.5	70.6
	12	0.1	0.2	0.5	1.2	4.7	0.3	0.6	1.6	4.2	13.2	0.6	1.1	2.6	6.4	18.7
(c)	55	53.8	71.5	81.4	85.4	91.2	81.4	94.4	98.6	98.8	99.7	90.2	96.0	98.6	99.0	99.8
	45	27.7	58.3	75.9	88.2	90.2	56.4	85.5	95.2	98.8	98.8	67.1	91.8	98.2	99.8	99.8
	35	10.4	40.7	57.4	77.1	87.2	28.0	63.1	83.6	95.1	98.7	32.6	74.3	87.0	96.2	98.8
	25	2.5	10.3	31.0	52.7	75.2	6.9	26.1	51.7	75.8	94.0	9.0	35.6	60.1	82.2	96.8
	15	0.0	0.0	1.5	2.8	23.6	0.0	0.9	3.7	8.8	49.1	1.0	1.1	5.9	9.1	54.9
	12	0.2	0.5	0.0	0.0	1.2	1.2	0.7	1.0	1.6	9.5	1.2	0.7	1.9	3.6	10.7

Figure 6. *Top* 1*, top* 5 *and top* 10 *accuracy of (a) training set, (b) testing on synthetic license plate images and (c) testing on real license plate images. Chance performance is* $1/46,650 = 0.002\%$*. See also Figure 7.*

contend with varying font size, style, and contrast.

We used the publicly available implementation of libsvm [10]. The training parameters of the radial basis function, c and γ, were determined by performing a grid search to maximize the accuracy on the testing image set. This obviously gave the SVM an advantage over the CNN, but we found that it was necessary to avoid over-fitting to the training data set.

In order to directly compare to the CNN results, we again assume that the probability of recognizing individual characters p is statistically independent and therefore the probability of recognizing 3 neighboring characters is p^3. The SVM training accuracy for each of 5 noise levels is 47.4%, 49.3%, 61.4%, 75.3%, and 85.7%. The testing accuracy is 1.7%, 6.7%, 15.8%, 29.5%, and 52.9%, Figure 7. In contrast, the testing accuracy for the CNN is 2.6%, 14.8%, 40.9%, 71.0%, and 88.6%, corresponding to an average improvement of more than a factor of two.

We trained a second SVM on SIFT features [11] as opposed to the raw image pixels. The results were nearly identical. Despite having the advantage of only recognizing a single character and having the position of that character localized, the CNN significantly outperformed the SVM.

Discussion

We have shown that observers are not able to extract useful information that remains in highly degraded images of license plates. We have also shown that information remains in images of highly degraded license plates of resolution as low as 1.9 pixels per character and with noise levels as low as −3.0 db. We have shown how to train a deep convolutional neural network using synthetically generated and degraded images to recognize characters on real world license plate images. We were able to relax the assumption of known font style, font size, background template, contrast and location of characters. The accuracies obtained from our CNN are significantly better than humans, a simple correlation-based method, and a traditional non-linear support vector machine.

One limitation of our approach is that we assume a perspectively corrected license plate. In real-world settings, however, this may not always be the case. It is possible to remove such distortions [12]. Given the nature of our highly degraded images, however, it may be difficult to accurately isolate the corners of the license plate necessary to remove perspective distortion. A second limitation is our assumption that the license plate contains only 6 character. This assumption only holds for some US states, with other states having between 4 and 8 characters. We propose to handle this variation by training different CNNs to handle different character configurations. And finally, our CNN may benefit from a denoising [13] and/or deblurring [14] pre-processing stage that improves the image quality.

Acknowledgment

This work was supported by a grant from the Department of Justice, NIJ (2016-R2-CX-0012), the National Science Foundation (CNS-1205521), and an equipment grant from NVIDIA.

References

[1] Q. Ye and D. Doermann, "Text detection and recognition in imagery: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 7, pp. 1480–1500, 2015.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, 2012, pp. 1097–1105.

[3] 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," *International Conference on Learning Representations*, vol. abs/1312.6082, 2014.

[4] 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, 2016.

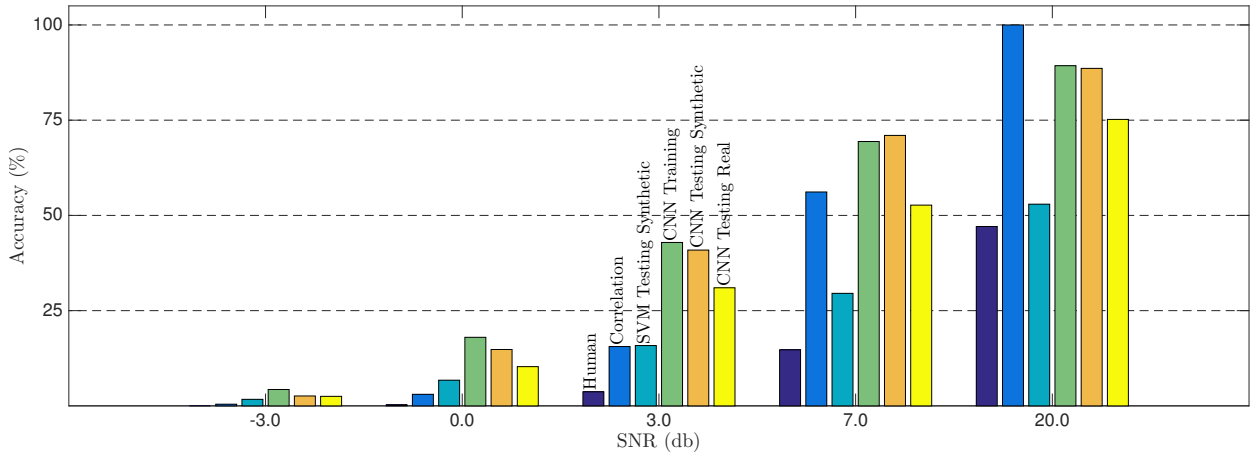

Figure 7. *Performance comparison of humans, correlation, SVM, and CNN (training, synthetic testing and real-world testing) at resolution of* 25 *pixels and at five different SNR levels. See also Figure 6.*

[5] P. Svoboda, M. Hradis, L. Marsik, and P. Zemcík, "CNN for license plate motion deblurring," *CoRR*, vol. abs/1602.07873, 2016.

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

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[8] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22Nd ACM International Conference on Multimedia*, 2014.

[9] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, 1995.

[10] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, 2011.

[11] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, 2004.

[12] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[13] J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli, "Image denoising using scale mixtures of Gaussians in the wavelet domain," *IEEE Transactions on Image Processing*, vol. 12, no. 11, pp. 1338–1351, 2003.

[14] W. T. Freeman, T. R. Jones, and E. C. Pasztor, "Example-based super-resolution," *IEEE Computer Graphics Applications*, vol. 22, no. 2, pp. 56–65, 2002.

Author Biography

Shruti Agarwal *is a 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 Digital Image Processing, Computer Vision and Graphics.*

Du Tran *is a research scientist at Facebook AI Research and Applied Machine Learning. He graduated with a Ph.D. degree in computer science from Dartmouth College. He received an M.S. degree in computer science from University of Illinois at Urbana-Champaign. Before coming to Dartmouth, he was a research staff at Nanyang Technological University. His research interests are computer vision, machine learning, and computer graphics, with specific interests in human activity and video event analysis.*

Lorenzo Torresani *is an Associate Professor in the Computer Science Department at Dartmouth College. He received a Laurea Degree in Computer Science from the University of Milan (Italy) in 1996, and an M.S. and a Ph.D. in Computer Science from Stanford University in 2001 and 2005, respectively. In the past, he has worked at several industrial research labs including Microsoft Research Cambridge, Like.com and Digital Persona. His research interests are in computer vision and machine learning. He is the recipient of a CVPR best student paper prize, a National Science Foundation CAREER Award, and a Google Faculty Research Award.*

Hany Farid *is the Albert Bradley 1915 Third Century Professor and Chair of Computer Science at Dartmouth. His research focuses on digital forensics, image analysis, and human perception. He received his undergraduate 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.*

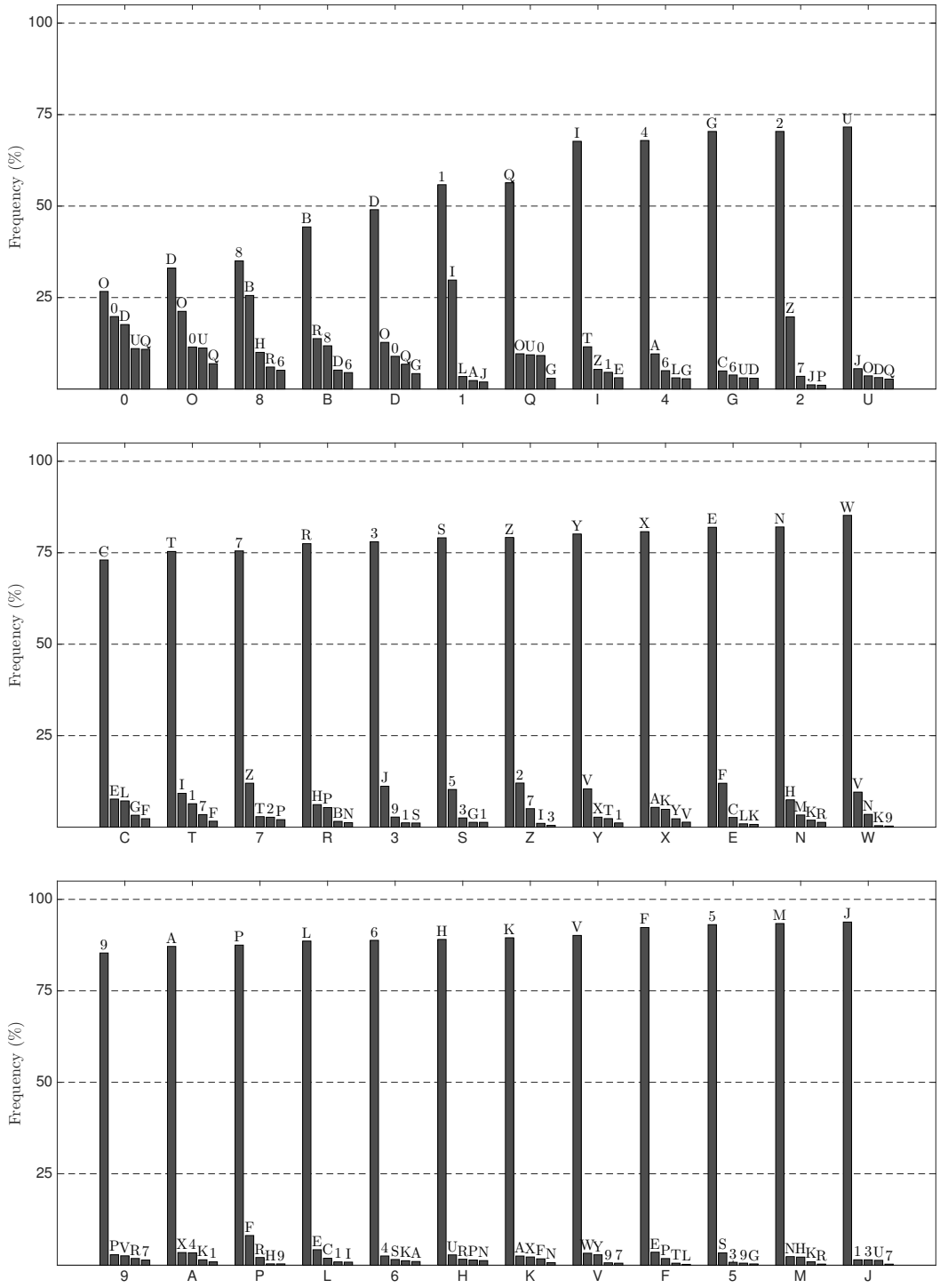

Figure 8. *Each bar graph corresponds to one of each* 36 *alpha-numeric characters (horizontal label). Shown in each graph is the* 5 *most frequently matched characters for a given character (annotated above each bar).*