https://doi.org/10.2352/ISSN.2470-1173.2018.10.IMAWM-337

Logo detection and recognition with synthetic images

Daniel Mas Montserrat; School of Electrical and Computer Engineering; Purdue University; West Lafayette, Indiana, USA
Qian Lin; HP Labs, HP Inc; Palo Alto, California, USA
Jan Allebach; School of Electrical and Computer Engineering; Purdue University; West Lafayette, Indiana, USA
Edward J. Delp; School of Electrical and Computer Engineering; Purdue University; West Lafayette, Indiana, USA

Abstract

During recent years, deep learning methods have shown to be effective for image classification, localization and detection. Convolutional Neural Networks (CNN) are used to extract information from images and are the main element of modern machine learning and computer vision methods. CNNs can be used for logo detection and recognition. Logo detection consist on locate and recognize commercial brand logos within an image. These methods are useful in the areas of online brand management or ad placement. The performance of this methods is closely related on the quantity and the quality of the data, typically image/label pairs, used to train the CNNs. Collecting the pair of images and labels, commonly referred as ground truth, can be expensive and time consuming. Multiple techniques try to solve this problem by either transforming the available data using data augmentation methods or by creating new images from scratch or from other images using image synthesis methods. In this paper, we investigate the latter approach. We segment background images, extract depth information and then blend logo images accordingly in order to create new real looking images. This approach allows us to create an indefinite number of images with a minimum manual labeling effort. The synthetic images can later be used to train CNNs for logo detection and recognition.

Introduction

Logo detection is an important part in validation of product placement, online brand management and contextual ad placement (placing relevant ads on webpages, images, and videos) [1]. The same techniques used for object detection and recognition methods can be used for detecting logo within images. Several API services are capable to detect commercial logos [2, 3, 4, 5]. Logo detection can be used as a tool for brand analysis in social media like presented in [6], where beer brand logos are detected in images extracted from Twitter [7] and are combined with male/female face detection to study the presence of the brands on the internet and its relation with the gender of the consumers.

Available datasets containing logo images are usually limited in number of logos (classes) and number of images [8, 9, 10]. This can be a problem as methods based on deep learning typically require large amounts of training images (around 5,000 training samples per class [11]) to have a good performance. Also, real world applications require to work with a large variety of logos (classes) and the possibility of easily adding new ones. Data augmentation and image synthesis methods can provide a useful and scalable alternative to the tedious and expensive task of manually labeling large amounts of images.

Data augmentation methods typically apply linear and non-linear transformations on the training data to create new samples. Transformations can include color changes, spatial rotations, warping and other deformations. This set of transformations do not change the labels of the training samples.

Image synthesis methods consist on creating new images from scratch or by combining other images. One or multiple labels can be assigned to the generated images. Several methods have been presented to synthesize images to train object detectors [12, 13]. In this paper we present a method to automatically create new images containing logos. Our method is based on the technique used in SynthText [14]. SynthText is a dataset containing images with text created by blending rendered text into background images. In our work, images are created by applying transformations to images containing logos and then blending them into background images using their depth information. Figure 1 presents an overview of the synthesis process. The pipeline is described in more detail in the following sections.

Finally, we use the synthesized images to train the Faster R-CNN (Region-based Convolutional Neural Network) [15] and a variant named PVANet [16]. Both networks are composed of three parts: a feature extractor, a region proposal network, and a classifier. The networks are able to locate and classify multiple logos in an image. A more detailed description is presented in the following section.

The main contribution of this paper is to use image synthesis techniques to generate an arbitrary number of images and then use them to train CNN based methods for logo detection.

This paper is organized as follows, in Section 2, we present an overview of related work in object and logo recognition, data augmentation, image synthesis and common logo datasets. In Section 3, we propose our method for image synthesis. In Section 4, we present the experimental evaluation. We finish with Section 5, by presenting conclusions and future improvements.

Overview Of Related Work

Typically, logo detection is performed by adapting object detection methods in the domain of commercial logos [8, 1] (i.e. treating each logo as a different object or class). Before the popularization of deep learning, the main approach for object detection and other image processing and computer vision tasks was the use of hand crafted visual features like SIFT [17] and texture descriptors [18], combined with statistical classifiers, such as Nearest Neighbor (NN) [19] or Support Vector Machines (SVM) [20].

Recent advances in hardware and increasing availability of large labeled datasets (e.g. ImageNet [21]) have made possible an improvement on deep learning methods. This new methods provide promising results in computer vision tasks such as object recognition or image classification [15, 21, 22, 23, 24] and in other domains such as speech recognition, economics, neuro-

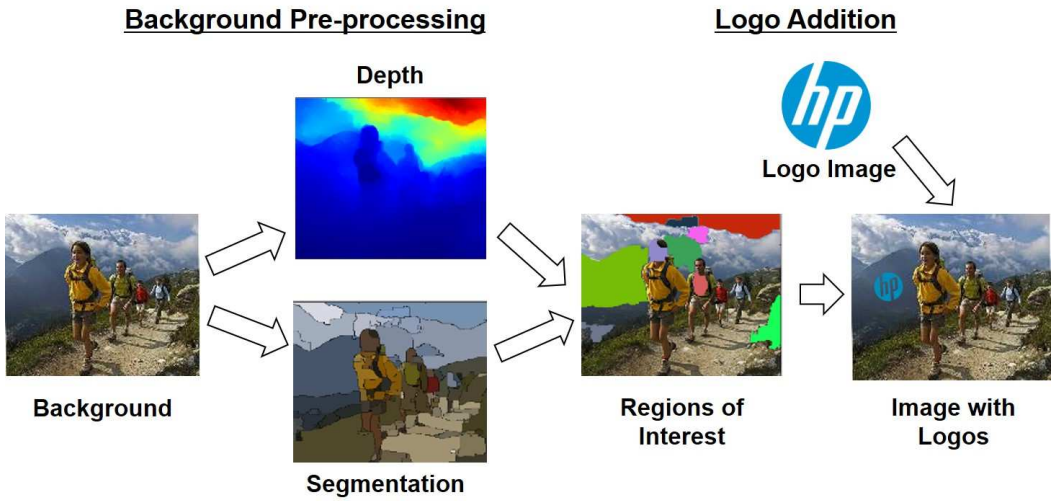

Figure 1. *Image synthesis pipeline*

science, chemistry or genomics [25, 26].

Convolutional Neural Networks (CNN) [25, 11, 24] are the core element of deep learning methods. CNNs combine convolutional filters with linear and non-linear operations to learn and extract visual information from images. Methods using CNNs are the leading approaches in image classification competitions (i.e. ImageNet [21]) and object detection competitions (i.e. Pascal VOC [27] and MS COCO [28]).

Convolutional Neural Networks are typically formed by a feature extraction subnet with several convolutional filters and a decision subnet with one or multiple fully connected layers [29, 24]. The filters of the initial convolutional layer of the feature extraction subnet learn to detect simple features such as color or edges, while the deeper layers learn to detect more complex features (e.g. complex shapes, faces, or animals). Non-linear operators can be included between convolutional layers. Some of these operators include Rectified Linear Unit (ReLU), which are layers that output the positive part of its input, or Max-pooling layers, which perform a non-linear down-sampling by partitioning the input image into non-overlapping rectangles and selecting the maximum value inside each rectangle. The fully connected layers that form the decision subnet consist of a set of affine or linear transformations followed by non-linear operations (typically a ReLU). Commonly, the last fully connected layer outputs a probability value or confidence score for each class.

The multiple layers that form a CNN have a set of weights and parameters that are learned from training samples using Backpropagation [25] and gradient-based optimization methods (e.g. Stochastic Gradient Descent (SGD) [26]). First, initial values are assigned to the weights and parameters of the network. The initial values can be randomly assigned, or fine-tuning can be used. Fine-tuning is a common practice that consist on training a CNN with a large generic dataset (e.g. ImageNet, MS COCO) and use the weights as an initialization. Then, gradient propagation and weight update stages are repeated over a fixed number of iterations. At each iteration, one or multiple input images are propagated through the network until they reach the output layer. The output of the network is compared with the ground truth value using a distance or loss function. As closer the predicted output is to the desired output, the smaller the error measure will be. The error value is backpropagated through the network and the weights are updated using a gradient-based method.

In our work, we use a common network named VGG16 (Visual Geometry Group) [30]. VGG16 contains 5 sets of layers, each formed by a pair of convolutional and ReLU layers followed by a max pooling layer. The number of filters in the convolutional layers are 65, 128, 256, 512 and 512 respectively. All the filters have a size of 3×3. VGG16 have 3 fully connected layers after the last convolutional layer.

Multiple methods for object detection using CNNs have been presented this recent years. The Region-Based Convolutional Neural Network (R-CNN) [31] is an architecture that locates and classifies multiple objects by combining a CNN and an external region proposal method. A region proposal method is an algorithm that outputs a set of regions of interest, typically defined with bounding boxes. A commonly used region proposal method is Selective Search [32]. This algorithm proposes regions of interest by using similarity measures based on color and visual features. R-CNN method crops and resize each region of interest and classifies them using a CNN. The original architecture uses a CNN with five convolutional layers and two fully connected layers, although any CNN classifier can be used.

Some more complex methods for object detection include Fast R-CNN [23] and, the method used in this work, Faster R-CNN [15]. Fast R-CNN is a method based on R-CNN in which the full image is processed by the convolutional layers and then, regions of the output of the last convolutional layer are cropped and classified. The network is formed by a set convolutional layers, fully-connected layers, an external region proposal method (typically Selective Search) and a Region of Interest (RoI) pooling layer. The RoI pooling layer applies max-pooling to each region of interest using a grid of a fixed size (typically 7×7). Fast R-CNN also introduces a bounding box regressor, a layer that outputs a fine-tuned location of bounding boxes.

Faster R-CNN is based on Fast R-CNN but substitutes the external region proposal methods by a Region Proposal Network

(RPN). RPN is a neural network that generates regions of interest using the features of the output of the last convolutional layer. RPN is formed by a 3×3 sliding window that outputs a set of bounding boxes (typically 9) with different sizes and aspect ratios and a fully connected layer that assigns a binary class (foreground or background) to each bounding box. Faster R-CNN can be trained end-to-end and is used in some 1st-place entries in ImageNet and MS COCO competitions [15] and in commercial systems like Pinterest [33].

In our work we also use PVANet [16]. PVANet is a lightweight version of Faster R-CNN. The model is smaller with 5 convolutional layers and 3 fully connected layers. While Faster R-CNN only uses the features of the last convolutional layer to localize and classify, this network combines the features of the last 3 convolutional layers. This method is faster than Fast R-CNN but can produce lower accuracy in some scenarios.

Many other object detection algorithms, including the previously ones described, output several overlapping bounding boxes. In order to merge them, Non-Maximum Suppression (NMS) algorithm [31] is used. NMS removes a bounding box if it largely overlaps with another bounding box of the same class with higher confidence score.

New methods for object detection based on deep learning are constantly appearing. Some of them include: Single Shot Multibox Detector (SSD) [34] or You Only Look Once (YOLO) [35] and YOLOv2 [36]. This methods typically provide a faster performance than Faster R-CNN but obtain a lower accuracy [34].

Several previous works have been presented for logo detection and recognition using hand crafted visual features [8] and based on deep learning [1, 37, 13]. The work presented in [1] makes use of Fast R-CNN with VGG16 and selective search for logo detection with and without localization obtaining a mean average precision of 74.4%. The work presented in [37] uses Faster R-CNN obtaining an 81.1% of accuracy. The work presented in [13] also uses Faster R-CNN with VGG16 and a smaller network named ZF [38] combined with data augmentation techniques obtaining a mean average precision of 85.4% on logo recognition with localization. The work presented in [39] uses an approach similar to R-CNN trained with a large number of images obtaining an accuracy of 96%.

Figure 2. *Image samples from FlickrLogos-32.*

There are several available labeled datasets with images containing logos in the wild. FlickrLogos-32 [8] dataset is the most used in recent works for training and testing. This dataset contains 32 different brands (classes), each with various versions. The dataset is composed by 8240 images mined from the Flickr image search engine [40]. The dataset is divided into training set, with 1280 images (40 per class) containing one or more logos and 3000 images with no logo content, and testing set, 960 images (30 per class) containing genuine logos and 3000 without logo content. The images without logo content are also referred as background images or distractors. The ground truth consists on a label (class) and a binary segmentation mask assigned to each image that contains a logo. A new version of the dataset, FlickrLogos-47, is available. FlickrLogos-47 contains the same images than FlickrLogos-32 but the labeling has been improved. In FlickrLogos-32, the logos composed by a symbol and text, only the symbol is treated as a logo, while in FlickrLogos-47 each part is treated as a different class (e.g. adidas-text, adidas-symbol). In our work, we use FlickrLogos-32 for evaluation purposes. Figure 2 presents some image examples of the dataset.

Other datasets include FlickrLogos-27 [41], which is composed of 27 different logos with a total of 810 annotated images (30 images per class) and 4207 distractor images. BelgaLogos [9] is a dataset with 37 different logos composed by 10,000 images with a binary label (1 if the logo is present and 0 otherwise) which 1,321 of them contain bounding boxes indicating the location of the logos. MICC-Logos [42] contains a total of 720 images with 13 different logos. TopLogo-10 [10] is a dataset containing 700 labeled images of 10 different clothing brands. Logos-32plus [43] is a dataset containing a total of 7,830 images with logos from the same corpus of FickrLogos-32. Logos in the Wild Dataset [44] is a new dataset containing the largest available number of training samples with a total of 11,054 images with 871 different brands. LOGO-Net [37] is a dataset with 160 different logos with a total of 73,414 labeled image. LOGO-Net is not currently available to the public. WebLogo-2M [45] is a dataset with 194 logos with 2,190,757 labeled images. This dataset does not contain bounding boxes but only the label of the logos in the image. It has been labeled in an unsupervised manner, so the labels might be incorrectly assigned.

Data augmentation techniques, such as random cropping, flipping or color changes [21], have been used for computer vision task using both hand crafted visual features [8, 46] and deep learning [12]. These techniques, usually help the network to avoid overfitting and to generalize better obtaining a higher accuracy. For example, the method presented in [46] obtains an increase of accuracy of 3.5% in 2010 ImageNet after applying data augmentation.

Image synthesis has been used to create new training samples for deep learning methods. One common approach is to use image compositing by adding foreground images (objects to be detected) into background images. This approach is used in [14] for text localization. The work presented in [13, 10] uses this approach for logo detection. FlickrBelgaLogos [47] is a public dataset synthetically created using image compositing with the logos extracted from BelgaLogos.

Synthesized images from 3D virtual simulations have been used for training neural networks for self-driving vehicles [48] or other tasks based on reinforced learning [49]. Methods such as [50] use Recurrent Neural Networks (RNN) to generate new training samples using information extracted from a training dataset. Generative Adversarial Networks (GANs) are networks able to generate new images that resemble the images used in the training process. The work presented in [51] uses this approach to create new training samples.

Our Proposed Approach

In this section we present our method for synthesizing images. Because our approach is based on the synthesis pipeline used in SynthText [14], we name our synthetic dataset as SynthLogo. The image synthesis process consists on estimating the depth and segmentation information of background images and blend logo images accordingly. Each step is described in the following sections.

Logo Images Acquisition

The image synthesis starts by obtaining multiple logo images. Our dataset contains a total of 604 different logos. The 604 logos include the ones that form FlickrLogos-32, FlickLogos-27, BelgaLogos and MICC-Logos and some extra classes including logos from popular brands of food, drinks, clothing, technology, transportation, finance, etc. The images are obtained from Google Search. We search for PNG images with alpha layer so we can separate the part of the image that corresponds to the logo from the background. Between 3 to 10 different images are obtained for each logo. We manually check them to remove outliers or undesired images. The same brand can have different versions of logos and some can contain both text and symbols. We do not make distinction between logo versions or text or symbol part and we assign one unique class per brand.

Background Images

We use the same 8,000 background images as used in SynthText. The images are obtained from Google Search and manually checked so they not contain text and they do not contain any logo from our corpus. Figure 3 shows some example of background images used.

Then, depth information is estimated, and the background image is segmented as explained in following sections. Precomputed values of depth estimation and background segmentation are available together with the original background images and code [52]. We choose to use the pre-computed values.

Figure 3. *Examples of background images.*

Depth Estimation

Depth information is inferred using a CNN as described in [53]. This method starts by dividing the image in superpixels. A superpixel is a set of neighboring pixels whit similar RGB values. Then 244×244 patches are cropped for each superpixel and processed by a CNN. The CNN contains 5 convolutional layers and 2 fully connected layers. This CNN estimates the depth value of the superpixel located in the center of the crop. A regularization factor is added in the loss function to accomplish pair-wise smoothness in the depth prediction between neighboring superpixels.

Background Segmentation

The background images are segmented by detecting the contours of the elements forming the image as described in [54]. The contours are locally computed using brightness, color and texture gradients. Once the contours are detected, watershed is used to segment the image. Watershed is a method that clusters all the pixels located in a region defined by a contour. Then, small segments are merged using color similarities.

Image Blending

Next, a segment of a background image is selected. Small segments or segments with extreme aspect ratios are discarded. Then, a logo image is randomly selected and resized so the largest size is 0.9 times the largest side of the segment. Then, the logo image is randomly rotated with a probability of 0.3 with a degree randomly selected between -90, 180 or 90 degrees. A small color jittering is randomly applied with a probability of 0.5 in the HSV color space. A total of 3 random values are selected uniformly from -10 to 10 and added to the hue, saturation and value channels respectively. Color jittering aims to add small color variations that logos may present in the real world caused by different lighting conditions.

Then the logo is geometrically transformed using Random Sample Consensus (RANSAC) [55]. RANSAC is a method that allows to match and project a planar surface using matching points. A planar surface is estimated in the segment using the depth information and a homographic projection is estimated so the plane of the segment and the plane of the logo match.

Finally, the logo image is blended into the background. The blending is done by combining the pixels of the background image with the pixels of the transformed logo image. An alpha value, α, is selected randomly between 0.5 and 1 and alpha blending is performed as specified in equation 1. $I_{synthetic}$ is the generated image, I_{logo} is the logo image and $I_{background}$ is the background image.

$$I_{synthetic} = \alpha I_{logo} + (1-\alpha) I_{background} \tag{1}$$

The complete process is repeated 1 to 5 times per synthetic image, therefore an image will include one or more logos. A binary mask is used to check that there is no collision between logos.

SynthLogo

Using the process previously described, we create a total of 280,000 images. Each of the 8,000 background images is used several times. In Figure 4 some examples of synthetic images are presented.

This process allows us to create an arbitrary number of images and new logos can be easily added to the corpus by simply

obtaining images from any image search engine. In table 2 we compare our dataset with other existing datasets. One important aspect is how easy is to augment both the number of images and number of logos. In the table, we mark as "Scalable" the datasets that can add more logos and images with minor or none manual labeling efforts.

Figure 4. *Example of synthetic images.*

Experiments

We train Faster R-CNN+VGG16 [30] and PVANet [16] models, previously described, with the SynthLogo dataset. We start the training process using pre-trained models with MS COCO. FlickrLogos-32 is used as validation and testing set. We compute the Mean Average Precision (mAP), described later in this section, using the Pascal VOC 2010 [27] method. In this evaluation method, each ground truth bounding box is compared with all predicted bounding boxes. If predicted and true classes match, and the Intersection over Union (IoU) (Equation 2) between the predicted bounding box B_p and the ground truth bounding box B_{gt} is larger than 50%, the prediction is a True Positive (TP), otherwise is a False Positive (FP).

$$IoUOverlap = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} \tag{2}$$

Precision (Equation 3) (i.e. ratio between TP and the sum of TP) and Recall (Equation 4) (i.e. ratio between TP and total number of ground truth bounding boxes (N_{bbox})) values are computed to calculate the mAP.

$$Precision = \frac{TP}{TP+FP} \tag{3}$$

$$Recall = \frac{TP}{N_{bbox}} \tag{4}$$

The last layer of Faster R-CNN and PVANet provides a probability or confidence measure between 0 and 1. A prediction is discarded (i.e. consider as a background) if its confidence score is lower than a threshold and maintained otherwise. For each class, a Precision/Recall curve is obtained by varying the threshold parameter from 0 to 1. The Average Precision (AP) is the area under the curve. We average the AP value of each class obtaining the Mean Average Precision (mAP).

We train a Faster R-CNN+VGG16 and PVANet for 200,000 iterations and evaluate the model every 10,000 iterations with the training set of FlickrLogos-32. Then, we select the best set of weights for each model and evaluate on the testing set of FlickrLogos-32. Table 1 presents the best results for validation and testing of both models. We can observe that the mAP is lower than results reported in previous works where manually labeled images have been used [13, 44, 43] but in our work the manual effort is minimum and the number of logos can be easily increased. Some works presents similar results when using an open set of logos (arbitrarily large number of logos) and suggests on using Faster R-CNN as an initial stage of localization and then include CNN for classification [44]. A similar approach as ours is presented in [10] where a synthetic dataset of 463 different logos is created and then used to train Faster R-CNN obtaining a 25.0% mAP when only using synthetic data.

Table 1 Mean Average Precision (mAP) of logo detection methods trained with SynthLogo and tested on FlickrLogos-32

Model	train+val set	testing set
Faster R-CNN+VGG16	49.46%	47.66%
PVANet	40.44%	38.56%

Conclusions

In this paper, we adapted the SynthText technique to create a method for logo image generation. This technique allows us to generate an arbitrary number of images and to easily extend the number of logos in the dataset with minimum manual labeling. This new dataset is useful to train logo detection networks such as Faster R-CNN obtaining promising results.

In the future, we want to apply realistic color transformations and to create images with context consistency (e.g. place "Coca-Cola" logos in top of cans, or "Corona" logos in top of bottles) and to include bootstrapping techniques.

Acknowledgment

This work was supported by HP Labs. Address all correspondence to Edward J. Delp, ace@ecn.purdue.edu

References

[1] F. N. Iandola, A. Shen, P. Gao, and K. Keutzer, "Deeplogo: Hitting logo recognition with the deep neural network hammer," *arXiv:1510.02131*, 2015.

[2] "Pixel Intelligence," URL: www.developers.hp.com.

[3] "Meerkat," URL: www.meerkat.com.br.

[4] "Google Cloud," URL: www.cloud.google.com.

[5] "Logo Grab," URL: www.logograb.com.

[6] "A study on beer: logo detection and analysis on social media," URL: https://medium.com/@meerkat.cv/a-study-on-beer-logo-detection-and-analysis-on-social-media-9ab2dab0014c.

[7] "Twitter," URL: www.twitter.com.

[8] S. Romberg, L. G. Pueyo, R. Lienhart, and R. van Zwol, "Scalable logo recognition in real-world images," *Proceedings of the ACM International Conference on Multimedia Retrieval*, pp. 251–258, April 2011, Trento, Italy.

Table 2 Different datasets containing logo images.

Dataset	Number of brands	Number of annotated images	Type	Publicly Available	Scalable
FlickrLogos-32(47)[8]	32	2,240	Real	Yes	No
Logos-32plus [43]	32	7,830	Real	Yes	No
TopLogo10 [10]	10	700	Real	Yes	No
FlickrLogos-27 [41]	27	810	Real	Yes	No
MICC-Logos [42]	13	720	Real	Yes	No
BelgaLogos [9]	37	1,321	Real	Yes	No
FlickrBelgaLogos [47]	37	2,697	Synthetic	Yes	Yes
LOGO-Net [37]	160	73,414	Real	No	No
WebLogo-2M [45]	194	2,190,757	Bootstrap	Yes	Yes
Logos in the Wild [44]	871	11,054	Real	Yes	No
[13]	32	16,000	Synthetic	No	Yes
SynthLogo	604	280,000	Synthetic	No	Yes

[9] A. Joly and O. Buisson, "Logo retrieval with a contrario visual query expansion," *Proceedings of the ACM International Conference on Multimedia Retrieval*, pp. 581–584, October 2009, Beijing, China.

[10] S. G. Hang Su, Xiatian Zhu, "Deep learning logo detection with data expansion by synthesising context," *Proceedings of the IEEE Winter Conference on Applications of Computer Science*, pp. 20–25, March 2017, Santa Rosa, CA.

[11] I. Goodfellow, Y. Bengio, and A. Courville, "Introduction," *Deep Learning*. Cambridge, MA: MIT Press, 2016, vol. 1, p. 20.

[12] A. Rozantsev, V. Lepetit, and P. Fua, "On rendering synthetic images for training an object detector," *Computer Vision and Image Understanding*, vol. 137, pp. 24–37, August 2015.

[13] D. Mas Montserrat, Q. Lin, J. Allebach, and E. J. Delp, "Training object detection and recognition CNN models using data augmentation," *Proceedings of the IS&T International Symposium on Electronic Imaging*, January 2017, Burlingame, CA.

[14] 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, January 2016.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *Proceedings of the Advances in Neural Information Processing Systems*, pp. 91–99, December 2015, Montreal, Canada.

[16] S. Hong, B. Roh, K.-H. Kim, Y. Cheon, and M. Park, "PVANet: Lightweight deep neural networks for real-time object detection," *arXiv:1611.08588*, 2016.

[17] D. G. Lowe, "Object recognition from local scale-invariant features," *Proceedings of the International Conference on Computer Vision*, pp. 1150–1159, September 1999, Kerkya, Greece.

[18] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, no. 6, pp. 610–621, November 1973.

[19] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[20] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1097–1105, December 2012, Stateline, NV.

[22] H. Kaiming, Z. Xiangyu, R. Shaoqing, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *Proceedings of the European Conference on Computer Vision*, pp. 346–361, September 2014, Zurich, Switzerland.

[23] R. Girshick, "Fast R-CNN," *Proceedings of the International Conference on Computer Vision*, pp. 1440–1448, December 2015, Santiago, Chile.

[24] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, August 2013.

[25] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[26] L. Zhang, L. Zhang, and B. Du, "Deep learning for remote sensing data: A technical tutorial on the state of the art," *IEEE Geoscience and Remote Sensing Magazine*, vol. 4, no. 2, pp. 22–40, June 2016.

[27] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, January 2015.

[28] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," *Proceedings of the European Conference on Computer Vision*, pp. 740–755, September 2014, Zürich, Switzerland.

[29] C.-C. J. Kuo, "Understanding convolutional neural networks with a mathematical model," *Visual Communication and Image Representation*, vol. 41, pp. 406–413, November 2016.

[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Proceedings of the International Conference on Learning Representations*, May 2015, San Diego, CA.

[31] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587, 2014, Columbus, OH.

[32] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Visual Communication and Image Representation*, vol. 104, pp. 154–171, September 2013.

[33] "Pinterest," URL: www.pinterest.com.

[34] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," *Proceedings of*

the European Conference on Computer Vision, pp. 21–37, October 2016, Amsterdam, Netherlands.

[35] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *Proceedings of th IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, June 2016, Las Vegas, NV.

[36] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," *arXiv:1612.08242*, 2016.

[37] S. C. Hoi, X. Wu, H. Liu, Y. Wu, H. Wang, H. Xue, and Q. Wu, "Large-scale deep logo detection and brand recognition with deep region-based convolutional networks," *arXiv:1511.02462*, 2015.

[38] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," *Proceedings of the European Conference on Computer Vision*, pp. 818–833, September 2014, Zürich, Switzerland.

[39] S. Bianco, M. Buzzelli, D. Mazzini, and R. Schettini, "Deep learning for logo recognition," *Neurocomputing*, vol. 245, pp. 23–30, 2017.

[40] "Flickr," URL: flickr.com/.

[41] Y. Kalantidis, L. Pueyo, M. Trevisiol, R. van Zwol, and Y. Avrithis, "Scalable triangulation-based logo recognition," *Proceedings of the ACM International Conference on Multimedia Retrieval*, pp. 1–7, April 2011, Trento, Italy.

[42] H. Sahbi, L. Ballan, G. Serra, and A. D. Bimbo, "Context-dependent logo matching and recognition," *IEEE Transactions on Image Processing*, vol. 22, no. 8, pp. 1018–1031, August 2013.

[43] S. Bianco, M. Buzzelli, D. Mazzini, and R. Schettini, "Logo recognition using cnn features," *Proceedings of the International Conference on Image Analysis and Processing*, pp. 438–448, September 2015, Genova, Italy.

[44] A. Tüzkö, C. Herrmann, D. Manger, and J. Beyerer, "Open Set Logo Detection and Retrieval," *Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, January 2018, Funchal, Portugal.

[45] H. Su, S. Gong, and X. Zhu, "Weblogo-2m: Scalable logo detection by deep learning from the web," *Proceedings of the IEEE International Conference on Computer Vision*, no. 8, pp. 20–25, October 2017, Venice, Italy.

[46] M. Paulin, J. Revaud, Z. Harchaoui, F. Perronnin, and C. Schmid, "Transformation pursuit for image classification," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3646–3653, June 2014, Columbus, OH.

[47] P. Letessier, A. Joly, and O. Buisson, "Scalable mining of small visual objects," *Proceedings of the 20th ACM international conference on Multimedia*, pp. 599–608, 2012, Nara, Japan.

[48] J. Xu, D. Vazquez, A. Lopez, J. Marin, and D. Ponsa, "Learning a part-based pedestrian detector in virtual world," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 2121–2131, October 2014.

[49] C. Beattie, J. Z. Leibo, D. Teplyashin, T. Ward, M. Wainwright, H. Kttler, A. Lefrancq, S. Green, V. Valds, A. Sadik, J. Schrittwieser, K. Anderson, S. York, M. Cant, A. Cain, A. Bolton, S. Gaffney, H. King, D. Hassabis, S. Legg, and S. Petersen, "Deepmind lab," *arXiv:1612.03801*, 2015.

[50] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "Draw: A recurrent neural network for image generation," *Proceedings of the International Conference on Machine Learning*, pp. 1462–1471, July 2015, Lille, France.

[51] S. C. A Radford, L Metz, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv:1511.06434*, 2016.

[52] "SynthText Github," URL: github.com/ankush-me/SynthText.

[53] F. Liu, C. Shen, and G. Lin, "Deep convolutional neural fields for depth estimation from a single image," *Proceedings of the IEEE Conference of Computer Vision and Pattern Recognition*, pp. 5162–5170, June 2015, Boston, MA.

[54] C. F. P. Arbelaez, M. Maire and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, pp. 898–916, May 2011.

[55] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the Association for Computing Machinery*, vol. 24, no. 6, pp. 381–395, June 1981.

Author Biography

Daniel Mas Montserrat graduated from Polytechnic University of Catalonia in 2015. He is currently attending graduate school at Purdue University under the supervision of Professor Edward J. Delp. He is working as a research assistant in a project funded by HP Labs under the supervision of Professor Edward J. Delp, Professor Jan P. Allebach and Qian Lin. His main areas of research are Deep learning and signal and image processing.

Dr. Qian Lin is a distinguished technologist working on computer vision and deep learning research in HP Labs. Dr. Lin joined the Hewlett-Packard Company in 1992. She received her BS from Xi'an Jiaotong University in China, her MSEE from Purdue University, and her Ph.D. in Electrical Engineering from Stanford University. Dr. Lin is inventor/co-inventor for 44 issued patents. She was awarded Fellowship by the Society of Imaging Science and Technology (IS&T) in 2012, and Outstanding Electrical Engineer by the School of Electrical and Computer Engineering of Purdue University in 2013.

Jan P. Allebach is Hewlett-Packard Distinguished Professor of Electrical and Computer Engineering at Purdue University. Allebach is a Fellow of the IEEE, the National Academy of Inventors, the Society for Imaging Science and Technology (IS&T), and SPIE. He was named Electronic Imaging Scientist of the Year by IS&T and SPIE, and was named Honorary Member of IS&T, the highest award that IS&T bestows. He has received the IEEE Daniel E. Noble Award, the IS&T/OSA Edwin Land Medal, and is a member of the National Academy of Engineering. He currently serves as an IEEE Signal Processing Society Distinguished Lecturer (2016-2017).

Edward J. Delp was born in Cincinnati, Ohio. He is currently The Charles William Harrison Distinguished Professor of Electrical and Computer Engineering and Professor of Biomedical Engineering at Purdue University. His research interests include image and video processing, image analysis, computer vision, image and video compression, multimedia security, medical imaging, multimedia systems, communication and information theory. Dr. Delp is a Life Fellow of the IEEE, a Fellow of the SPIE, a Fellow of IS&T, and a Fellow of the American Institute of Medical and Biological Engineering.