Orti, O. [et al.]. Real-time logo detection in brand-related social media images. A: International Work-Conference on Artificial and Natural Neural Network. "Advances in Computational Intelligence, 15th International Work-Conference on Artificial Neural Networks, IWANN 2019: Gran Canaria, Spain, June 12-14, 2019: proceedings, part II". Berlín: Springer, 2019, p. 125-136. The final authenticated version is available online at https://doi.org/10.1007/978-3-030-20518-8_11

Real-time Logo Detection in Brand-related Social Media Images

Oscar Orti[1], Ruben Tous[2], Mauro Gomez[1], Jonatan Poveda[1], Leonel Cruz[2], and Otto Wust[1]

[1] Adsmurai. Barcelona, Spain
[2] Universitat Politècnica de Catalunya (UPC). Barcelona, Spain
[3] Barcelona Supercomputing Center (BSC). Barcelona, Spain

Abstract. This paper presents a work consisting in using deep convolutional neural networks (CNNs) for real-time logo detection in brand-related social media images. The final goal is to facilitate searching and discovering user-generated content (UGC) with potential value for digital marketing tasks. The images are captured in real time and automatically annotated with two CNNs designed for object detection, SSD InceptionV2 and Faster Atrous InceptionV4 (that provides better performance on small objects). We report experiments with 2 real brands, Estrella Damm and Futbol Club Barcelona. We examine the impact of different configurations and derive conclusions aiming to pave the way towards systematic and optimized methodologies for automatic logo detection in UGC.

Keywords: Social Media · Instagram · User Generated Content · Deep Learning · Marketing · Object Detection

1 Introduction

Nowadays, there is a growing interest by profit and non-profit organizations in deriving benefit from the photos that users share on social networks such as Instagram or Twitter [17], a part of the so-called user-generated content (UGC). A significant part of these images has potential value for organizational intelligence and digital marketing tasks. On the one hand, users' photos can be analyzed to obtain knowledge about users behavior and opinions in general, or with respect to a certain products, services or brands. On the other hand, some users' photos can be of value themselves, as original and authentic content that can be used, upon users' permission, in the different organizations' communication channels.

Platforms for photo-centric UGC are proliferating rapidly nowadays (e.g. Adsmurai [15] , Olapic [11], Chute [1] and Curalate[2]), but analyzing images on social media streams is challenging. The potential bandwidth to analyze is huge and, while they help, user defined tags are scarce and noisy. Any automated processing component needs to be extremely efficient and scalable. A large part of current solutions relies on costly manual curation tasks over random samples. This way many contents are not even processed, and many valuable photos

go unnoticed. Adoption of image recognition techniques in commercial UGC systems is currently very limited.

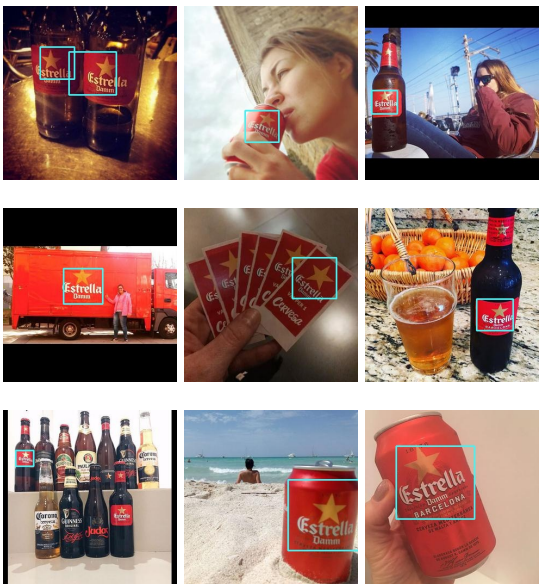


Fig. 1. Example images containing one or more occurrences of the Estrella Damm logo.

In this work, we address the logo detection problem, one of most challenging tasks that an automated UGC system needs to face. We have developed a system that automatically processes, in real-time, an incoming stream of social media images and detects and localizes all the occurrences of any of a set of supported logotypes. The system makes use of two state-of-the-art deep convolutional neural networks (CNNs) designed for object detection, SSD InceptionV2 and Faster Atrous InceptionV4 (that provides better performance on small objects). The resulting system is currently being integrated within a real commercial service, the Adsmurai's Visual Commerce Platform [16].

In this paper we describe the technical design of the system and the results of the performance evaluation experiments in which real images related with two commercial brands, Estrella Damm and Futbol Club Barcelona, have been used. We examine the impact of different configurations and derive conclusions aiming to pave the way towards systematic and optimized methodologies for automatic logo detection in UGC.

2 Related Work

The work presented in this paper is related to recent works attempting to facilitate the classification and search of images in social networks such as Instagram and Twitter. Some works, such as [10], [15] or [12], also apply scene-based and object-based image recognition techniques to enrich the metadata originally present in the images in order to facilitate their processing. All latest works rely on CNNs as an underlying technique. In our case, the applied image recognition techniques, while also relying in CNNs, are tuned for content curation for digital marketing tasks. This implies new problems, such as the need to deal with small datasets (e.g. brand-based image datasets). Previous works such as [5], [18] and [4] also classify social media data paying special attention to brands and products.

Regarding the detection of logos, one of the current state-of-art architectures is the Single Shot MultiBox Detector (SSD) [9]. SSD works similar than YOLO [13] but includes a RPN (Region Proposal Network, technique popularized by [3]) to improve the diversity of prior boxes by running it on multiple convolutional layers and different depth levels. This CNN is based on VGG feature extractor and performs faster than the Faster-RCNN [14]. Since SSD appeared some variations of that object detection architectures, but the main approach changed to object segmentation and instance segmentation with new architectures like MaskRCNN [7] or RetinaNet [8]. Nowadays it is possible to instantiate objects in almost real time video, but not in average computers, although it does not require as much computation capacity as ResNet because of there is an awesome work of memory and calculus optimization.

3 Methodology

3.1 Outline of a UGC curation platform

Logo detection is just one of the tasks of a UGC curation platform. A UGC curation platform is a software system that processes a continuous stream (usually with a huge and volatile bandwidth) of social media images and prepares them (e.g. by indexing them, annotating them with the proper metadata, etc.) to be exploited for organizational intelligence or digital marketing tasks. The functionality of these platforms is usually divided into two different stages, data acquisition and data consumption. Both stages interact through a common database containing metadata about the images, and they can occur concurrently (once images start feeding the database users or analytics tools can start using them). During data acquisition new images are captured in real-time, as they are published on the underlying sources (Instagram in our case). Descriptors of the images (including the URL pointing to the image content) are acquired using the APIs provided by these underlying sources. Once a new image is captured, it is processed by different components (e.g. scene recognition, face detection, logo detection) that automatically enrich the image's metadata with annotations that

describe their visual content. During the data consumption stage users (or analytics tools) can navigate, search and select images from the generated database.

3.2 Logo detection

The logo detection system described in this paper has two main functionalities, training and inference. Both functionalities are provided as APIs to facilitate their integration within the UGC curation platform. On the one hand, given dataset of training images depicting a logotype and labeled in PASCALVOC style, the system is able to train and store a new model for detecting that logotype. The system uses one model for each logo, to facilitate the inclusion of new logotypes dynamically. Fig. 2 outlines the workflow of the train API. The API is fed with training images for a given logotype. The checkpoints generated during training store the model's weights at a given training step, and are used by the inference API to generate a model's inference graph. The saved checkpoints are also used as pre-loaded weights to re-train the model. The model validation is conducted in parallel to the training (each 10 minutes).

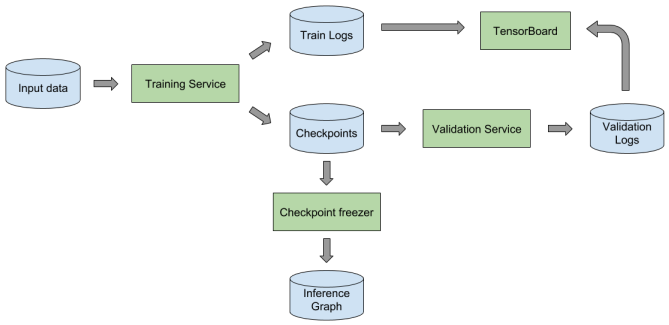


Fig. 2. Train API workflow

On the other hand, a scalable inference API is provided. The inference API downloads the checkpoints of the available models (the ones generated during training) from cloud storage (AWS S3). The API receives JSON requests with a list of new images and the identifier of the model to be used. The requested images are downloaded to the filesystem. If still not loaded, the API loads the inference graph. The inference graph is applied to detect all the occurrences of the requested logotype in each image. The bounding boxes of all the detections of each input image are finally returned in PASCALVOC style.

All the explained functionalities are integrated following the human-in-the-loop (HITL) pattern (see Fig. 3). When the inferred logo detections for a given

image are validated during the consumption stage of the UGC curation platform (e.g. if the image is selected by the user), the image is added to the model's training dataset, thus enabling to continuously improve the model's performance.

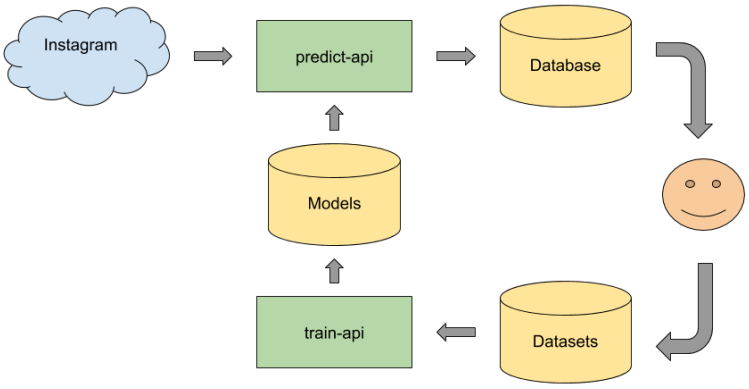


Fig. 3. Object detector HITL

3.3 Networks

Internally, the developed logo detection APIs rely on one or more object detection algorithms. With the help of the TensorFlow Object Detection API, the system has been designed to enable the implementation of any state-of-the-art logo detection CNN-based method. It can be configured with different feature extractors such as InceptionV2, InceptionV3, InceptionV4, ResNet50, ResNet101 or MobileNet. Each one could be highly customized by configuring the activation functions or batch normalization among others. It is also possible to define which kind of region proposal to use among SSD, FasterRCNN or RFCN and customize its hyper-parameters. To define the training it is possible to choose the batch size and the learning-rate optimizer, which could be RMSProp, momentum or scheduled. Different kind of data augmentations are available too, like random horizontal flip or random crops, among others.

The implementation described in this paper uses two CNNs, SSD InceptionV2 and Faster Atrous InceptionV4 (to improve the performance on small objects). On the one hand, Faster R-CNN uses a region proposal network to create boundary boxes and utilizes those boxes to classify objects. While it is considered the start-of-the-art in accuracy, the whole process runs at 7 frames per second. Far below what a real-time processing needs. SSD speeds up the process by eliminating the need of the region proposal network. To recover the

drop in accuracy, SSD applies a few improvements including multi-scale features and default boxes. These improvements allow SSD to match the Faster R-CNNs accuracy using lower resolution images, which further pushes the speed higher.

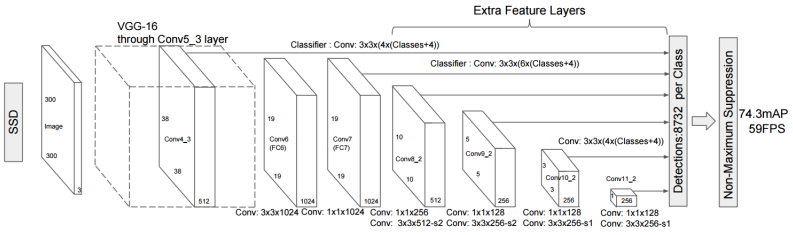


Fig. 4. SSD architecture from [9].

The original SSD uses VGG16 to extract features, but we used InceptionV2 as a feature extractor because it provides better image classification performance in our context. The SSD uses multi-scale feature maps to detect objects, so it uses higher resolution feature maps to detect small objects and lower resolution for bigger objects. SSD uses hard negative example mining because we make far more predictions than the number of objects presence. So there are much more negative matches than positive matches. This creates a class imbalance which hurts training. We are training the model to learn background space rather than detecting objects. However, SSD still requires negative sampling so it can learn what constitutes a bad prediction. So, instead of using all the negatives, we sort those negatives by their calculated confidence loss. SSD picks the negatives with the top loss and makes sure the ratio between the picked negatives and positives is at most 3:1. This leads to a faster and more stable training.

On the other hand, Faster Atrous InceptionV4 is designed to achieve the best performance without taking into account the training or inference time. This CNN uses InceptionV4 (also called InceptionResNetV2) as a feature extractor, which achieves the state-of-the-art in image classification accuracy. The atrous version of the Faster RCNN [6], uses dilated convolutions to achieve better results. Before the Faster RCNN, the slowest part in Fast RCNN was Selective Search or Edge boxes. Faster RCNN replaces selective search with a very small convolutional network called Region Proposal Network to generate regions of interests.

4 Experiments

We annotated two datasets related to two real brands, Estrella Damm and Futbol Club Barcelona. The manual annotation was performed using the LabelImg Tool. Each dataset is composed with 650 images. We have used transfer learning

to train the networks because of data scarcity. So the 650 images (70% training and 30% validation) were used to fine-tune the CNN architectures that had been pre-trained with the COCO dataset. The validation set is not presented to the neural network along all the training phase. The datasets were converted to TensorFlow's TFRecord format. During evaluation we measured the precision, recall and the mean average precision (mAP), which requires to define an intersection over union (IoU) condition (the ratio between the intersection and the union of the predicted boxes and the ground truth boxes). We used two IoU thresholds, 0.5 and 0.75, to determine if a detection is a true positive. The whole project is developed in Python 3.6. The CNNs have been implemented in TensorFlow 1.4. All data (datasets, models and inference graphs) are stored in Amazon AWS S3. APIs endpoints are implemented in Flask. Docker containers are used for the different system functionalities. One container is used for the predict API. The train API is divided into four containers, one for each service: train, validation, TensorBoard and checkpoint freezer. Docker Swarm is used for container orchestration.

4.1 Single Shot Detection InceptionV2 configuration

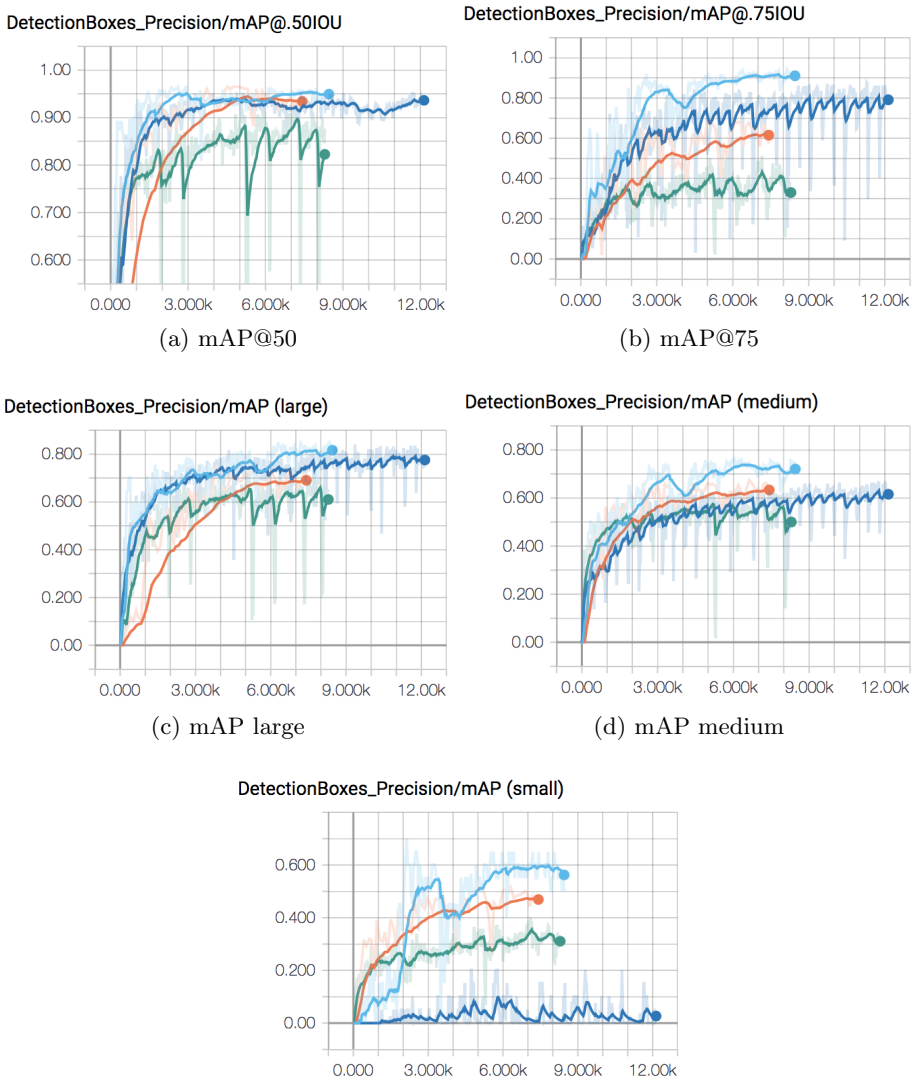
The SSD requires a fixed input shape, so all image are resized in the preprocessing step into (512, 512, 3). The SSD uses an anchor generator with the configured aspect ratios (1, 2, 3, 0.5, 0.33) to propose regions of interest. Then InceptionV2 configured with batch normalization is used as a feature extractor, with batch normalization. The classification loss is defined with the sigmoid, whereas the localization loss is the smooth L1. It is also configured a hard example minner to get between 0 to 3 negatives examples for each positive example. This allows the network to learn negatives and improve its performance. The training batch size is 24 (because of hardware limitations) and the learning rate is optimized using RMSprop. So, The SSD is trained using Mini-Batch Gradient Descent.

4.2 Faster Atrous RCNN InceptionV4 configuration

The Faster RCNN does not require a fixed input shape. So, in this case, the image size is bounded by a minimum (600) and a maximum (1024) dimension size. If the dimension of an image is lower or higher than a threshold, then the image is resized, maintaining its aspect ratio, to respect the bounds. The feature extractor is implemented using InceptionV4 (or Inception ResNet V2). The loss functions used are softmax. The training batch size is 1 because it's not possible to generate batch sizes greater than 1 with images of different sizes. So, this CNN is trained with stochastic gradient descent.

5 Results

Figures 5 and 7 show the mAP and recall results for both, the FC Barcelona and the Estrella Damm, datasets respectively.



(a) mAP@50

(b) mAP@75

(c) mAP large

(d) mAP medium

(e) mAP small

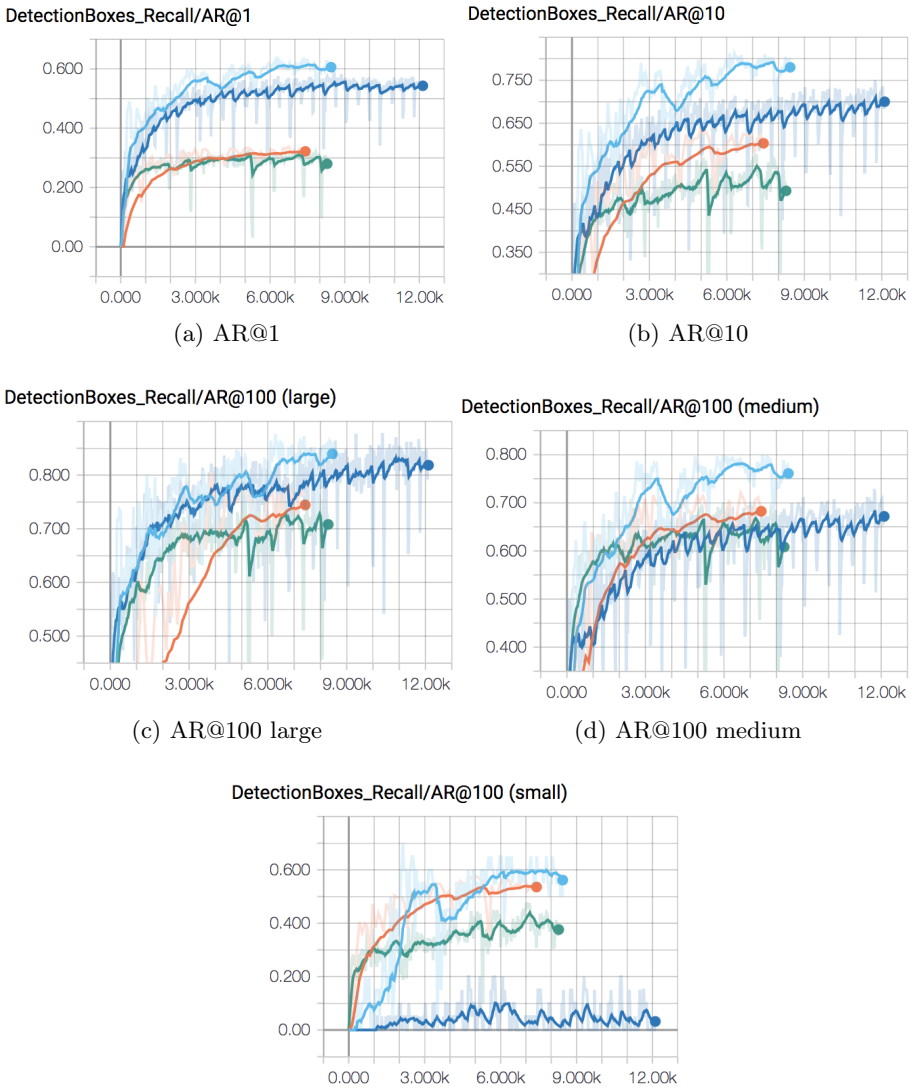
Fig. 5. Evaluation mAP results for both datasets (with different IoU thresholds, (a) 0.5 and (b) 0.75, and different logotype sizes, (c) large, (d) medium, (e) small). Dark blue: Estrella Damm SSD, Light blue: Estrella Damm Faster R-CNN, Green: FCB SSD, Orange: FCB Faster R-CNN.

The results show that the SSD provides good results, around 0.9 mAP with a 0.5 IoU threshold, for both datasets. The results for FC Barcelona are inferior to the ones for Estrella Damm mainly because the FC Barcelona logotype is frequently found over deformable objects (e.g. clothes). Considering that SSD is faster and also easier to train than Faster R-CNN, we conclude that it is well suited for real-time logo detection because of its good trade-off between inference time and accuracy.

However, the results obtained with SSD for small objects are not satisfactory. This is caused mainly because small logos may not appear in all feature maps (the more an object appears on a feature map, the more likely that the MultiBox algorithm can detect it). One way to address this limitation is to increase the size of the input image, but that reduces the speed at which SSD can run and does not completely alleviate the problem of detecting small objects. Faster R-CNN obtains better results with small logotypes for both datasets.

Fig. 6. Validation images along the training

Figure 6 and 8 show some example detections for both, the FC Barcelona and the Estrella Damm, datasets respectively.



(a) AR@1 (b) AR@10

(c) AR@100 large (d) AR@100 medium

(e) AR@100 small

Fig. 7. Evaluation recall results for both datasets (with different IoU thresholds, (a) 0.5 and (b) 0.75, and different logotype sizes, (c) large, (d) medium, (e) small). Dark blue: Estrella Damm SSD, Light blue: Estrella Damm Faster R-CNN, Green: FCB SSD, Orange: FCB Faster R-CNN.

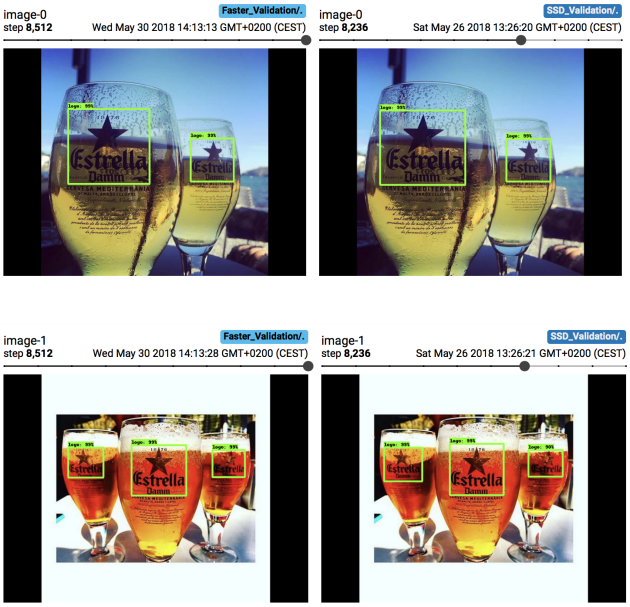


Fig. 8. Validation images along the training

6 Conclusions

This paper describes the design and implementation of a software architecture for real-time logo detection in brand-related social media images. The final goal is to facilitate searching and discovering user-generated content (UGC) with potential value for digital marketing tasks. The images are captured in real time and automatically annotated with two CNNs designed for object detection, SSD InceptionV2 and Faster Atrous InceptionV4. We report experiments with 2 real brands, Estrella Damm and Futbol Club Barcelona. Unlike Faster R-CNNs, which contain multiple moving parts and components, SSDs are unified, encapsulated in a single end-to-end network, making SSDs easier to train and well suited for real-time object detection because of their good trade-of between inference time and accuracy. However, our experiments show that SSDs provide poor performance for small objects, mainly because small objects may not appear on all feature maps. For this reason our proposed solution includes both, an SSD network and a Faster R-CNN network that can be selected depending on the goal.

Acknowledgements

This work is partially supported by the Spanish Ministry of Economy and Competitivity under contract TIN2015-65316-P and by the SGR programme (2014-SGR-1051 and 2017-SGR-962) of the Catalan Government.

References

1. Chute. enterprise ugc, http://www.getchute.com/ (Accessed June 6, 2017)
2. Curalate, https://www.curalate.com/ (Accessed June 6, 2017)
3. Erhan, D., Szegedy, C., Toshev, A., Anguelov, D.: Scalable object detection using deep neural networks. CoRR **abs/1312.2249** (2013), http://arxiv.org/abs/1312.2249
4. Gao, Y., Wang, F., Luan, H., Chua, T.: Brand data gathering from live social media streams. In: Proceedings of the International Conference on Multimedia Retrieval, ICMR 2014, Glasgow, United Kingdom - April 01 - 04, 2014. p. 169 (2014)
5. Gao, Y., Zhen, Y., Li, H., Chua, T.: Filtering of brand-related microblogs using social-smooth multiview embedding. IEEE Trans. Multimedia **18**(10), 2115–2126 (2016)
6. Guan, T., Zhu, H.: Atrous faster r-cnn for small scale object detection. In: 2017 2nd International Conference on Multimedia and Image Processing (ICMIP). pp. 16–21 (March 2017). https://doi.org/10.1109/ICMIP.2017.37
7. He, K., Gkioxari, G., Dollár, P., Girshick, R.B.: Mask R-CNN. CoRR **abs/1703.06870** (2017), http://arxiv.org/abs/1703.06870
8. Lin, T., Goyal, P., Girshick, R.B., He, K., Dollár, P.: Focal loss for dense object detection. CoRR **abs/1708.02002** (2017), http://arxiv.org/abs/1708.02002
9. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector (2016), http://arxiv.org/abs/1512.02325, to appear.
10. Nguyen, D.T., Alam, F., Ofli, F., Imran, M.: Automatic image filtering on social networks using deep learning and perceptual hashing during crises. CoRR **abs/1704.02602** (2017), http://arxiv.org/abs/1704.02602
11. Olapic. earned content platform, http://www.olapic.com/ (Accessed June 6, 2017)
12. Park, M., Li, H., Kim, J.: HARRISON: A benchmark on hashtag recommendation for real-world images in social networks. CoRR **abs/1605.05054** (2016)
13. Redmon, J., Divvala, S.K., Girshick, R.B., Farhadi, A.: You only look once: Unified, real-time object detection. CoRR **abs/1506.02640** (2015), http://arxiv.org/abs/1506.02640
14. Ren, S., He, K., Girshick, R.B., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. CoRR **abs/1506.01497** (2015), http://arxiv.org/abs/1506.01497
15. Tous, R., Gomez, M., Poveda, J., Cruz, L., Wüst, O., Makni, M., Ayguadé, E.: Automated curation of brand-related social media images with deep learning. Multimedia Tools Appl. **77**(20), 27123–27142 (2018). https://doi.org/10.1007/s11042-018-5910-z, https://doi.org/10.1007/s11042-018-5910-z
16. Tous, R., Wüst, O., Gomez, M., Poveda, J., Elena, M., Torres, J., Makni, M., Ayguadé, E.: User-generated content curation with deep convolutional neural networks. In: Proceedings of the 2016 IEEE International Conference on Big Data, BigData 2016, Washington DC, USA, December 5-8, 2016. pp. 2535–2540 (2016)
17. Tous, R., Torres, J., Ayguad, E.: Multimedia big data computing for in-depth event analysis. In: Proceedings of the 2015 IEEE International Conference on Multimedia Big Data (BigMM), April 20-22, 2015, Beijing, China. pp. 144–147. IEEE (2015)
18. Zhao, S., Yao, H., Zhao, S., Jiang, X., Jiang, X.: Multi-modal microblog classification via multi-task learning. Multimedia Tools Appl. **75**(15), 8921–8938 (2016)