Deep Learning for Logo Recognition

Simone Bianco, Marco Buzzelli, Davide Mazzini, Raimondo Schettini

DISCo - Università degli Studi di Milano-Bicocca, 20126 Milano, Italy

Abstract

In this paper we propose a method for logo recognition using deep learning. Our recognition pipeline is composed of a logo region proposal followed by a Convolutional Neural Network (CNN) specifically trained for logo classification, even if they are not precisely localized. Experiments are carried out on the FlickrLogos-32 database, and we evaluate the effect on recognition performance of synthetic versus real data augmentation, and image pre-processing. Moreover, we systematically investigate the benefits of different training choices such as class-balancing, sample-weighting and explicit modeling the background class (i.e. no-logo regions). Experimental results confirm the feasibility of the proposed method, that outperforms the methods in the state of the art.

Keywords: Logo recognition, Deep Learning, Convolutional Neural Network, Data augmentation, FlickrLogos-32

1. Introduction

Logo recognition in images and videos is the key problem in a wide range of applications, such as copyright infringement detection, contextual advertise placement, vehicle logo for intelligent traffic-control systems [1], automated computation of brand-related statistics on social media [2], augmented reality [3], etc.

Traditionally, logo recognition has been addressed with keypoint-based detectors and descriptors [4, 5, 6, 7]. For example Romberg and Lienhart [8] presented a scalable logo recognition technique based on feature bundling, where individual local features are aggregated with features from their spatial neighborhood into Bag of Words (BoW). Romberg et al. [9] exploited a method for encoding and indexing the relative spatial layout of local features detected in the logo images. Based on the analysis of the local features and the composition of basic spatial structures, such as edges and triangles, they derived a quantized representation of the regions in the logos. Revaud et al. [10] introduced a technique to down-weight the score of those noisy logo detections by learning a dedicated burstiness model for the input logo. Recently some works investigating the use of deep learning for logo recognition appeared [11, 12, 13]. Bianco et al. [11] and Eggert et al. [12] investigated the use of pretrained Convolutional Neural Networks (CNN) and synthetically generated data for logo recognition, trying different techniques to deal with the limited amount of training data. Also Iandola et al. [13] investigated a similar approach, proposing and evaluating several network architectures. Given the limited amount of training data available for the logo recognition task, all these methods work on networks pretrained on different tasks.

In this paper we propose a method for logo recognition exploiting deep learning. The recognition pipeline is composed by a recall-oriented logo region proposal [14], followed by a Convolutional Neural Network (CNN) specifically trained for logo classification, even if they are not precisely localized. Within this pipeline, we investigate the benefit on the recognition performance of the application of different machine learning techniques in training, such as image pre-processing, class-balancing, sample weighting, and synthetic data augmentation. Furthermore we prove the benefit of adding as positive examples candidate regions coming from the object proposal to the ground truth logos, and the benefit of enlarging the size of the actual dataset with real data augmentation and the use of a background class (i.e. no-logo regions) in training.

2. Proposed Method

The proposed classification pipeline is illustrated in Figure 1. Since logos may appear in any image location with any orientation and scale, and more logos can coexist in the same image, for each image we generate different object proposals, that are regions which are more likely to contain a logo. These proposal are then cropped to a common size to match the input dimensions of the neural network and are propagated through a CNN specifically trained for logo recognition.

In order to have performance as high as possible within this pipeline, we use an object proposal that is highly recall-oriented. For this reason, the CNN classifier should be designed and trained to take into account that the logo regions proposed may contain many false positives or only parts of actual logos. To address these problems we propose here a training framework and investigate the influence on the final recognition performance of different implementation choices.

In more detail, the training framework is reported in Figure 2. The training data preparation is composed by two main parts:

- **Precise ground-truth logo annotations**: Given a set of training images and associated ground-truth specifying logo position and class, we first crop logo regions and annotate them with the ground-truth class. These regions are rectangular crops that completely contain logos but, due to the prospective of the image or the logo particular shape, may also contain part of the background.

- **Object-proposal logo annotations**: Since we must automatically localize regions that may contain a logo, an object proposal algorithm is employed in the whole pipeline as shown in Figure 1. This algorithm is not applied only to the test images, but it is also run on the training images to extract regions that are more likely to contain a logo. Details about the particular algorithm used are given in the next subsection. Each object proposal in the training images is then labeled on the basis of its content: if it overlaps with a ground-truth logo region, it is annotated with the corresponding class and with the Intersection-over-Union (IoU) overlap ratio, otherwise it is labeld as background.

Within our training framework we investigate both the use of the precise ground-truth logo annotations alone or coupled with the object-proposal logo annotations. All positive instances, i.e. labeled logos and eventually object proposals that overlap with them by a significant amount (i.e. IoU$\geq$ 0.5), are used to train a Convolutional Neural Network whose architecture is given below. Different training choices are investigated within our framework in Figure 2:

- **Class balancing**: The logo classes are balanced by replicating the examples of classes with lower cardinality. Two different strategies are implemented: epoch-balancing, where classes are balanced in each training epoch, and batch-balancing, where classes are balanced in each training batch. The hypothesis is that this should prevent a classification bias of the CNN.

- **Data augmentation**: Training examples are augmented in number by generating random shifts of logo regions. The hypothesis is that this should make the CNN more robust to inaccurate logo localization at test time.

- **Contrast normalization**: Images are contrast-normalized by subtracting the mean and dividing by the standard deviation, which are extracted from the whole training set. The hypothesis is that this should make the CNN more robust to changes in the lighting and imaging conditions.

- **Sample weighting**: Positive instances are weighted on the basis of their overlap with ground-truth logo regions. The hypothesis is that this should make the CNN more confident on proposals highly overlapping with the ground truth logos.

- **Background class**: A background class is considered together with the logo classes. Background examples are not randomly selected, but are composed by the candidate regions generated by the object proposal algorithm on training images and that do not overlap with any logo. The hypothesis is that this should make the CNN more precise in discriminating logos and background class.

The actual contribution to the performance of each training choice considered will be discussed in Section 4.

After the CNN is trained, a threshold is learned on top of the CNN predictions. If the CNN prediction with the highest confidence is below this threshold, the candidate region is labeled as not being a logo, otherwise CNN prediction is left unchanged.

The testing framework is reported in Figure 3. Given a test image, we extract the object proposals with the same algorithm used for training. We then perform contrast-normalization over each proposal (if enabled at training time), and feed them to the CNN. The CNN predictions on the proposals are max-pooled and the class identified with highest confidence (eventually including the background class) is selected. If the CNN confidence for a logo class is above the threshold that has been learned in training, the corresponding logo class is assigned to the image, otherwise the image is labeled as not containing any logo.

2.1. Object proposal

For object proposal we exploit a Selective Search algorithm originally introduced by van de Sande et al. [15, 16]. The goal of Selective Search is to provide a set of regions likely to contain an instance of the object of interest, i.e. logos in our case. They can appear in any position and scale, and may have been acquired under different lighting conditions, and from slightly different point of views. The algorithm is designed to be highly recall-oriented; this implies that very few logos are not segmented, but also implies that a great number of false positive candidates are generated. The proposed regions will be disambiguated by the neural network that comes afterward.

2.2. Network Architecture

The architecture used for the experiments in the following sections is a tiny deep neural network. We opted for a tiny network because it is fast at test time and it can be trained on cheap GPUs in very short time. It also allows us to train the network without using any form of regularization like dropout [17], dropconnect [18], etc. decreasing even more the time needed for training and validating the network.

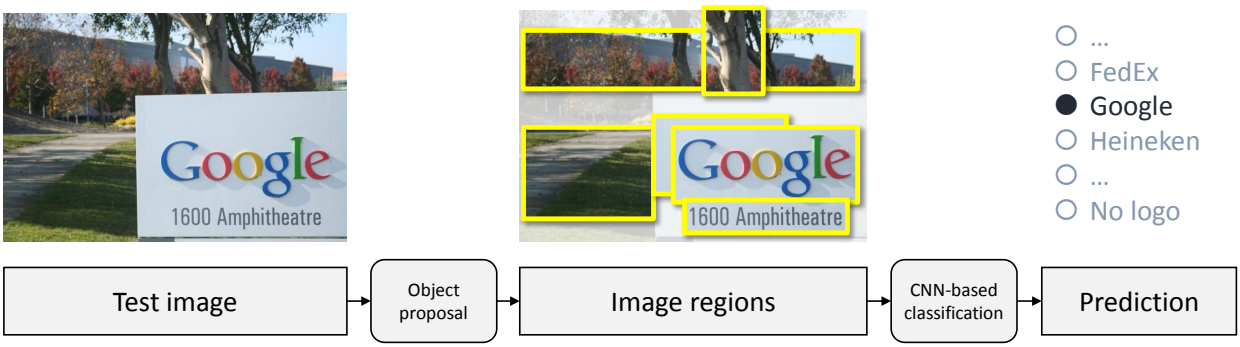

Figure 1: Simplified logo classification pipeline

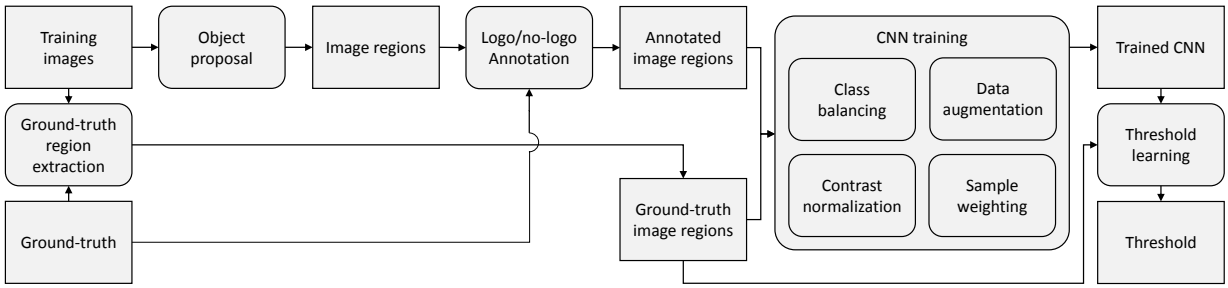

Figure 2: Logo recognition training framework.

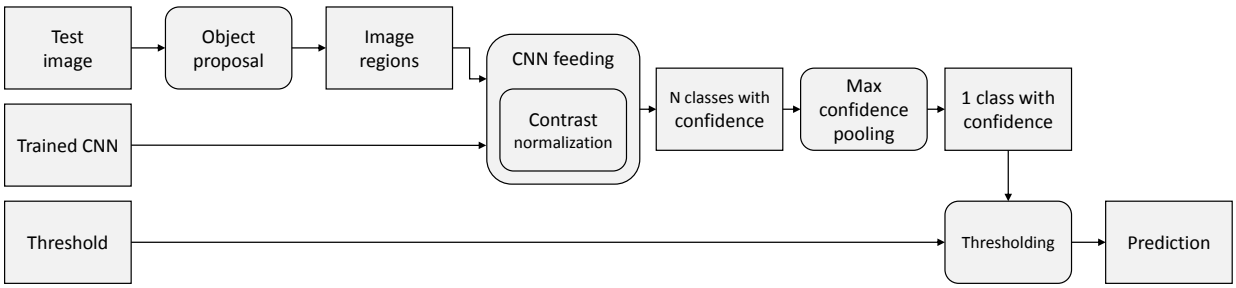

Figure 3: Logo recognition testing framework.

The same network structure was used by Krizhevsky in [19] on the CIFAR-10 dataset, where it was proven to be an high-performance network for the task of object recognition on tiny RGB images. It has three convolutional layers interleaved by ReLU nonlinearities and Pooling layers. All the pooling layers make the data dimensions halve after every Pooling block. The last part of the network (farthest from the input) consists in two Fully-connected layers with a final Softmax classifier. The whole net structure is presented in Table 1.
To give an idea of the network size, our network has 1.5×10^5 parameters whereas AlexNet (used in [12]) and GoogLeNet (a similar structure is used in [13]) have respectively 6×10^7 and 1.3×10^7 parameters. Therefore our network is less likely to overfit, even when the size of the training set is not large.

Table 1: Neural Network Architecture

	Layers
1	Conv 32 filters of 5x5
2	Pool (max) with stride 2
3	Relu
4	Conv 32 filters of 5x5
5	Relu
6	Pool (average) with stride 2
7	Conv 64 filters of 5x5
8	Relu
9	Pool (average) with stride 2
10	Fully Connected of size 64
11	Fully Connected of size 33
12	Softmax

3. Logos datasets

3.1. FlickrLogos-32 Dataset

FlickrLogos-32 dataset [9] is a publicly-available collection of photos showing 32 different logo brands. It is meant for the evaluation of logo retrieval and multi-class logo detection/recognition systems on real-world images. All logos have an approximately planar or cylindrical surface. For each class, the dataset offers 10 training images, 30 validation images, and 30 test images. An example image for each of the 32 classes of the FlickrLogos-32 dataset is reported in Figure 4.

3.2. Logos-32plus Dataset

Logos-32plus dataset is an expansion of the trainset of FlickrLogos-32. It has the same classes of objects as its counterpart but a larger cardinality (12312 instances). We collected this new dataset for three main reasons: first, since we want to test a deep learning approach, we needed a suitable dataset size. Second, we believe that Logos-32 dataset is not very representative of a data distribution for most real-world problems. Third, we hypothesize that synthetic data augmentation is not enough to model actual logo appearance variability. The Logos-32 dataset was collected with the aim to train keypoint-based approaches. Therefore the selection of images followed some implicit guidelines, such as: most of the images are on focus, no blurry or noisy images, and usually images with highly saturated colors. As a result, the variability of this dataset mainly resides on the amount of intraclass affine transformations which can be handled very well by keypoint-based detection methods. We collected this new dataset with the aim of taking into account a larger set of real imaging conditions and transformations that may occur in uncontrolled acquisitions.

We built the Logos-32plus dataset with images retrieved from both Flickr and Google image search. In particular, to increase the variability of data distribution we performed multiple queries for each logo. The dendrogram scheme in Figure 5 shows the tags used to compose the search queries used. To compose a single query we concatenate one leaf (a single logo) with a single tag of an ancestor node. The whole set of queries for each logo can be obtained by concatenating the logo name (leaf) with each tag contained in all the ancestors nodes. For example, all the queries used to search for the "Becks" logo are: "logo Becks", "merchandising Becks", "events Becks", "drink Becks", "bottle Becks", "can Becks", "beer Becks", "bier Becks" etc.

The dataset contains on average 400 examples per class, with each image including one or multiple instances of the same class. The detailed distribution of classes is shown in Figure 7 and a comparison between the FlickrLogos-32 and the Logos-32plus datasets is presented in Table 2. The dataset will be made available at http://www.ivl.disco.unimib.it/ (link available after acceptance).

3.3. Duplicates Removal

To ensure a high variability of the new dataset and to avoid any overlap with the existing one, we performed a semi-automatic check for duplicate images within the Logos-32plus dataset itself and with the FlickrLogos-32 dataset. The process has been carried out in two steps. First, we automatically found and discarded image duplicates using the SSIM measure [20]: we checked for similarity every pair of images within the Logos-32plus dataset itself and with the FlickrLogos-32 dataset using the SSIM measure. Images with SSIM measure over 0.9 have been discarded.

As a second step, we removed near duplicates in a semi-automatic manner. We say that two images are near duplicates if they depict the same scene with small differences in appearance with a particular focus on the portion of the image containing the logo. Examples of near duplicates are different overlapping crops of the same photo or images of the same scene from a different point of view.

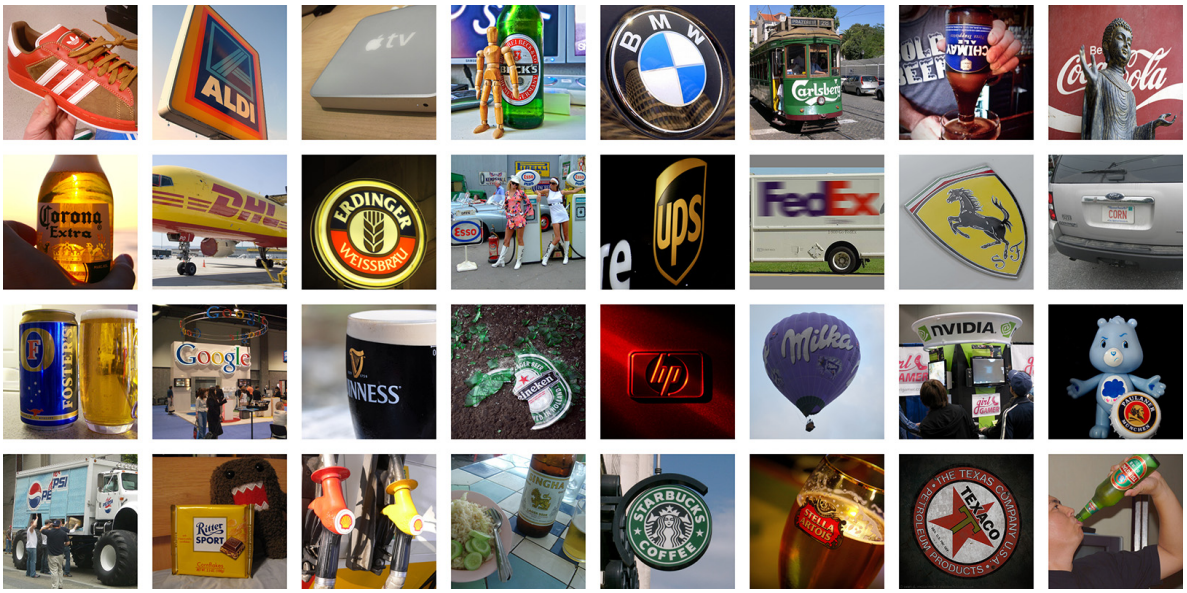

Figure 4: Example images for each of the 32 classes of the FlickrLogos-32 dataset.

An interesting example of near duplicates is shown in Figure 6. The two images depict the same gas station from a very similar point of view. The girls in the photo are in different poses but the appearance of the Esso logo in the two images is basically the same. In detail, to remove near duplicates we used the following procedure:

- we trained our CNN (structure in Table 1) from scratch on Logos-32plus dataset. To accomplish this task we fed the network with crops extracted from GT annotations and Object-proposals regions.
- We truncated the learned network leaving out the last two layers (softmax and last fully-connected). This network surgery operation let us use our network as a feature extractor exploiting the robust features learned by a deep neural network. We used this truncated network to extract features from every image crop that contains a tagged logo.
- We trained a k-NN classifier on top of the extracted features (using Logos-32plus as training set) and used it to retrieve from Logos-32plus and FlickrLogos-32 the nearest five results.
- Finally we manually checked for near duplicates among the five nearest results retrieved by the classifier. All the near duplicates have been discarded from the final dataset.

Figure 6: Example of near duplicates. The two images depict the same scene from a similar point of view. The appearance of the Esso logo in the two images is basically the same. We removed one of the two images from our Logos-32 plus dataset because the other one is included in the FlickrLogos-32 test set.

4. Experimental Setup and Results

Experiments are performed considering the different training choices described in Section 2. These include class balancing, data augmentation, image contrast normalization, sample weighting, addition of a background class, and addition of positive examples actually generated by the object proposal algorithm.

Each change to the training procedure is introduced one at a time, in order to assess its individual contribution, and the corrisponding value is underlined in Table 3 for better readability. All these configurations are trained using real data augmentation, i.e. with our extended Logos-32plus dataset in addition to FlickrLogos-32 training and validation sets. Results are reported in Table 3 in terms

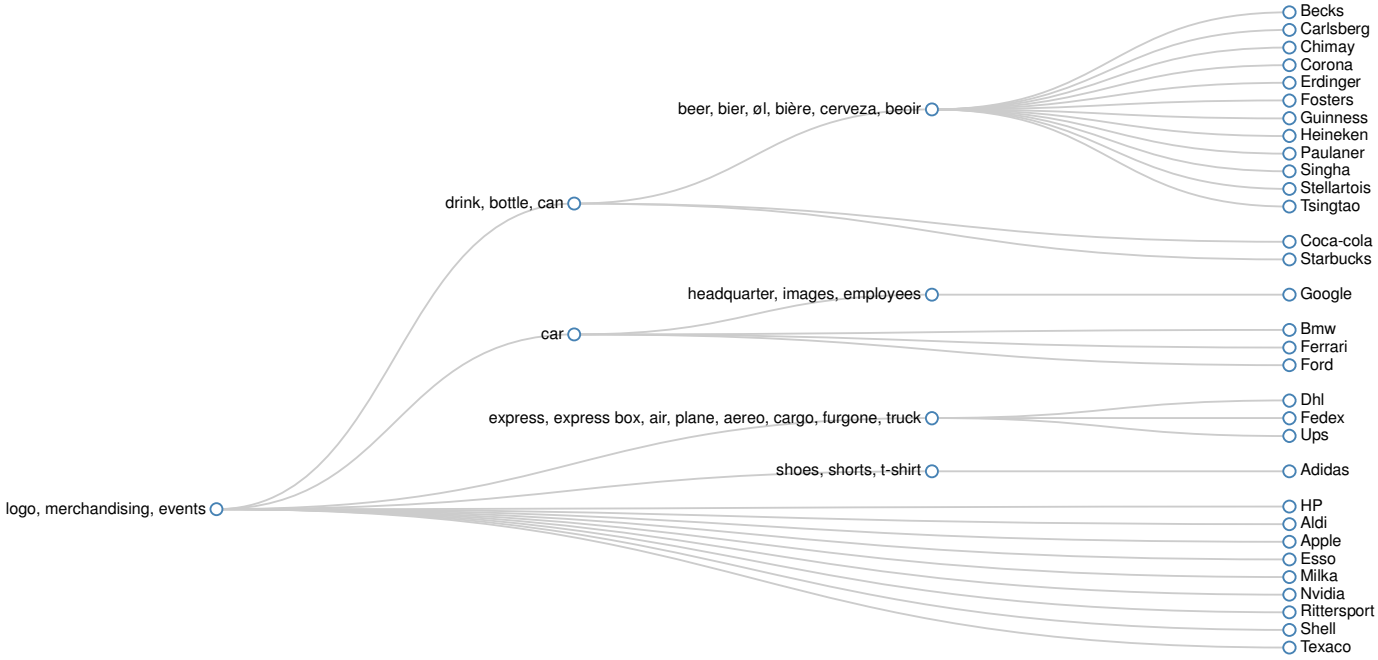

Figure 5: Dendrogram representing the queries composition used to download the Logos-32plus dataset. To retrieve images of becks logos we used for instance: "logo Becks", "merchandising Becks", "drink Becks", "bottle Becks", "beer Becks" etc.

Table 2: Comparison between FlickrLogos-32 and Logos-32plus datasets

	FlicrkLogos-32	**Logos-32plus**
Total images	8240	7830
Images containing logo instances	2240	7830
Train + Validation annotations	1803	12302
Average annotations for class (Train + Validation)	40	400
Total annotations	3405	12302

of both F1-measure and Accuracy on FlickrLogos-32 test set. With reference to Figure 3, the threshold on CNN predictions is automatically chosen to maximize the accuracy on FlickrLogos-32 training and validation sets. The best configuration is then compared to other state of the art methods in Table 4. As further investigation we quantify the contribution given from real data augmentation, by training the same solution on the original FlickrLogos-32 training set only. Finally, we assess the impact of the object proposal algorithm to the overall performance. To do this we add all the ground truth locations to the test set, instead of relying on the object proposal only.

From the results reported in Table 3 it is possible to see that with respect to a straightforward application of deep learning to the logo recognition task (i.e. Training Configuration I, *TC*-I), the different training choices considered are able to give a large increase in performance:

- The first jump in performance is obtained by including the background (i.e. no-logo examples) as a new class in training. Results are identified as *TC*-II and show an improvement in F1-measure and accuracy of 31.8% and 52.4% with respect to *TC*-I.

- A second jump is obtained by including object proposals coming from Selective Search as additional training examples. This configuration is named *TC*-III and improves the F1-measure and accuracy by 11.3% and 12.4% with respect to *TC*-II.

- A third jump in performance is obtained by augmenting the cardinality of object proposals coming from Selective Search by perturbing them with random translations (i.e. synthetic data augmentation). This configuration is named *TC*-IV and improves the F1-measure and accuracy by 11.7% and 20.9% with respect to *TC*-III.

- A further, smaller, improvement in performance is obtained by considering class balancing to account for different cardinalities, with "Epoch" balancing giving consistently better performance than the "Batch" counterpart (named *TC*-V and *TC*-VI respectively). In particular, *TC*-V improves the F1-measure and accuracy by 0.4% and 0.3% with respect to *TC*-IV.

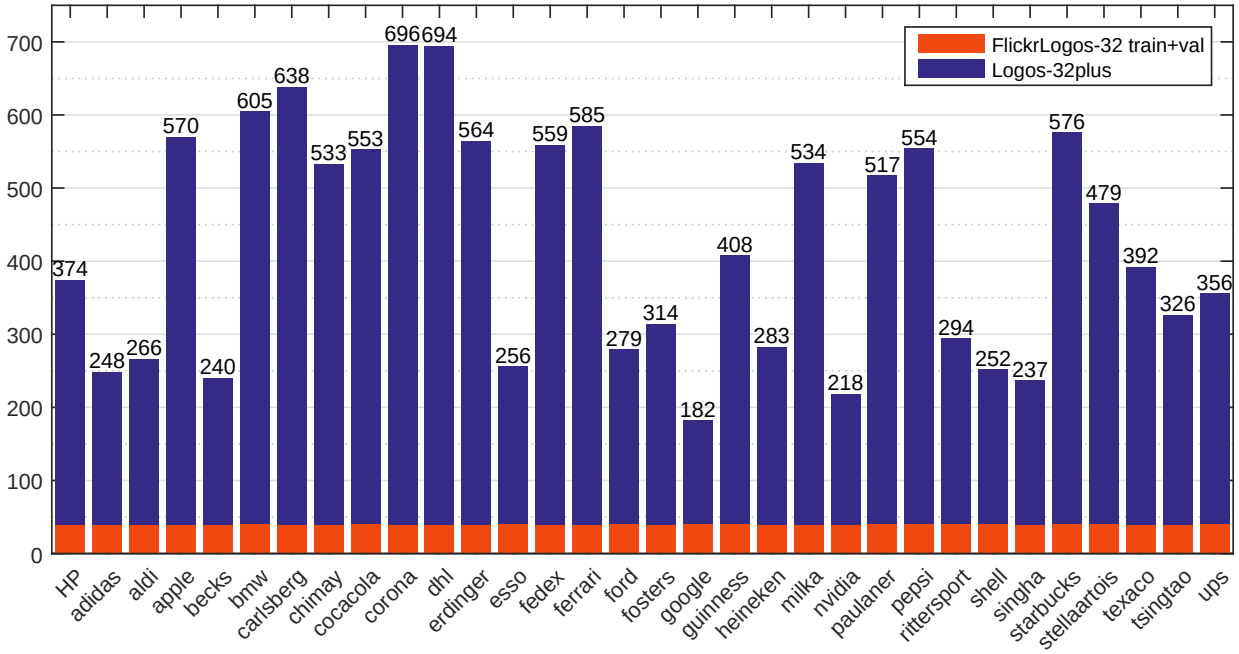

Figure 7: Graphical comparison of the distribution of the 32 logo classes between FlickLogos-32 and our augmented Logos-32plus dataset

Table 3: Experimental results showing the impact of the different training choices described in Section 2 on the final classification. Results are reported in terms of Precision, Recall, F1-measure and Accuracy.

Train. Config.	BG class	BBs	Data Augm.	Class bal.	Contr. norm.	Sample weight	Prec.	Rec.	F1	Acc.
I	No	GT	No	No	No	No	0.370	0.370	0.370	0.096
II	Yes	GT	No	No	No	No	0.713	0.665	0.688	0.620
III	Yes	GT+OP	No	No	No	No	0.816	0.787	0.801	0.744
IV	Yes	GT+OP	Yes	No	No	No	0.987	0.858	0.918	0.953
V	Yes	GT+OP	Yes	Epoch	No	No	0.986	0.865	0.922	0.956
VI	Yes	GT+OP	Yes	Batch	No	No	0.980	0.833	0.901	0.945
VII	**Yes**	**GT+OP**	**Yes**	**Epoch**	**Yes**	**No**	**0.989**	**0.906**	**0.946**	**0.958**
VIII	Yes	GT+OP	Yes	Epoch	Yes	Yes	0.984	0.875	0.926	0.951
IX	Yes	GT+OP	Yes	Batch	Yes	No	0.984	0.887	0.933	0.955
X	Yes	GT+OP	Yes	Batch	Yes	Yes	**0.989**	0.866	0.923	0.955

Legend to Table 3

Train. Config.	Identifier of the configuration used for training
BG class	Background class (no-logo examples) included in training
BBs	Bounding Boxes used as training examples
	GT Precise ground-truth logo annotations
	GT+OP Precise ground-truth and Object-proposal logo annotations
Data Augm.	Data Augmentation (translation)
Class bal.	Class balancing to account for different cardinalities
	Epoch Classes are balanced in each epoch
	Batch Classes are balanced in each batch as well
Contr. norm.	Pre-processing of training examples with contrast normalization
Sample weight	Weighting examples based on overlap between OP and GT

- Contrast normalization brings a further little but consistent improvement, with *TC*-VII improving the F1-measure and accuracy by 2.4% and 0.2% with respect to *TC*-V.
- Sample weighting instead (adopted in *TC*-VIII and *TC*-X), which consists in weighting training examples according to the degree of overlap between the object proposal and ground truth regions, results in lowering the final performance of the method.

The best configuration (i.e. *TC*-VII) trained on our extended training set is highlighted in bold in Table 3 and compared with the state of the art in Table 4. Performances of the other methods are taken from the respective papers and thus for some of them some performance measures are missing. From the results reported it is possible to see that the proposed solution is able to improve the F1-measure with respect to the best method in the state of the art by 3.8%, and the accuracy by 1.7%. It is worth to underline that the best results for the two metrics were obtained by different methods in the state of the art, i.e. by Romberg et. al [8] and BoW SIFT [8] respectively.

As a further comparison, we report the results obtained by our solution using only FlickrLogos-32 for training and keeping all the other training choices unchanged. This results in a drop in F1-measure by 14.7% and by 4.8% in accuracy, giving an idea of the benefit of real data augmentation with respect to a purely synthetic one [12]. As a final analysis, to understand if the major source of error in our method is the Selective Search module that is unable to have a high recall or if its the CNN itself that mispredicts the logo class, we perform an additional test by adding the actual logo ground truth region to the object proposals. This increases the F1-measure by 0.6% and the accuracy by 0.2% indicating that its the major source of error in our method is the CNN itself. Some examples of wrongly labeled candidate logo regions are reported in Figure 8. Candidates are generated by the object proposal and they have a IoU larger than 0.5 with the corresponding ground truth. The first and the third row depict the wrongly recognized regions labeled with their actual class, while the second and fourth one depict the nearest example in the training set using as features the activations of the last network layer before the softmax. Images are reported with the same resolution used to feed the CNN, i.e. 32×32 pixels.

4.1. Timings

Table 5: Timings of the whole recognition pipeline

Device	Proposal	Preproc.	Classif.	Overall
CPU	1.24 s	0.93 s	0.71 s	2.91 s
GPU	1.24 s	2.12 s	0.36 s	3.74 s

Table 5 shows the timings for the whole recognition procedure at test time. Experiments are performed on the same computer (Intel i7 3.40 GHz - 16 GB RAM) averaging the timings of 100 runs on different images.
Two different solutions are compared: the use of CPU or GPU (GeForce GTX 650) for the classification step.
The proposals extraction step runs always on CPU. The prepocessing time include the resize of every patch to match the CNN input size, the contrast normalization (negligible processing time) and eventually the time to copy the data from CPU to GPU memory. In Table 5 it is possible to notice that the overhead caused by the CPU-GPU memory transfer makes the overall time of the GPU solution higher than that of the CPU solution.

5. Conclusions

Logo recognition is fundamental in many application domains. The problem is that logos may appear in any position, scale and under any point of view in an image. Moreover, the images may be corrupted by many image artifacts and distortions.

The traditional approaches to logo recognition involve keypoint-based detectors and descriptors, or the use of CNNs pretrained on different tasks. Our solution employs a CNN specifically trained for the task of logo classification, even if they are not perfectly localized. We designed a complete recognition pipeline including a recall-oriented candidate logo region proposal that feeds our CNN.

Experiments are carried out on the FlickrLogos-32 database and on its enlarged version, Logos-32plus, collected by the authors. We systematically investigated the effect on recognition performance of synthetic versus real data augmentation, image pre-processing, and the benefits of different training choices such as class-balancing, sample-weighting and explicit modeling the background class (i.e. no-logo regions). Our best solution outperforms the methods in the state of the art and makes use of an explicit modeling of the background class, both precise and actual object-proposal logo annotations during training, synthetic data augmentation, epoch-based class balancing, and image contrast normalization as pre-processing, while sample weighting is disabled. The newly collected Logos-32plus and the trained CNN are made available for research purposes.

References

[1] A. P. Psyllos, C.-N. E. Anagnostopoulos, E. Kayafas, Vehicle logo recognition using a sift-based enhanced matching scheme, Intelligent Transportation Systems, IEEE Transactions on 11 (2) (2010) 322–328.

[2] Y. Gao, F. Wang, H. Luan, T.-S. Chua, Brand data gathering from live social media streams, in: Proceedings of International Conference on Multimedia Retrieval, ACM, 2014, p. 169.

[3] N. Hagbi, O. Bergig, J. El-Sana, M. Billinghurst, Shape recognition and pose estimation for mobile augmented reality, Visualization and Computer Graphics, IEEE Transactions on 17 (10) (2011) 1369–1379.

Figure 8: Wrongly labeled logos ordered by confidence. Highest confidence prediction is top-left. Images resolution is 32x32 pixels, i.e. the same used to feed the CNN. The first and third rows are the wrong labeled logos, the second and the fourth rows represent the nearest example in the training set (using the last network layer activations before the softmax as feature vector).

Table 4: Comparison of the best configuration in Table 3 with the methods in the state of the art.

Method	**Train data**	**Precision**	**Recall**	**F1**	**Accuracy**
BoW SIFT [8]	FL32	0.991	0.784	0.875	0.941
BoW SIFT + SP + SynQE [8]	FL32	0.994	0.826	0.902	N/A
Romberg et. al [9]	FL32	0.981	0.610	0.752	N/A
Revaud et. al [10]	FL32	$\geq$0.980	0.726	0.834÷0.841	N/A
Romberg et. al [8]	FL32	**0.999**	0.832	0.908	N/A
Bianco et. al [11]	FL32	0.909	0.845	0.876	0.884
Bianco et. al + Q.Exp. [11]	FL32	0.971	0.629	0.763	0.904
Eggert et. al [12]	FL32	0.996	0.786	0.879	0.846
DeepLogo [13]	FL32	N/A	N/A	N/A	0.896
Ours (*TC*-VII)	FL32	0.976	0.676	0.799	0.910
Ours (*TC*-VII)	FL32, L32+	0.989	**0.906**	**0.946**	**0.958**
Ours (*TC*-VII, adding GT to the obj. prop.)	FL32	0.968	0.755	0.848	0.917
Ours (*TC*-VII, adding GT to the obj. prop.)	FL32, L32+	0.989	**0.917**	**0.952**	**0.960**

[4] A. D. Bagdanov, L. Ballan, M. Bertini, A. Del Bimbo, Trademark matching and retrieval in sports video databases, in: Proceedings of the international workshop on Workshop on multimedia information retrieval, ACM, 2007, pp. 79–86.

[5] J. Kleban, X. Xie, W.-Y. Ma, Spatial pyramid mining for logo detection in natural scenes, in: Multimedia and Expo, 2008 IEEE International Conference on, IEEE, 2008, pp. 1077–1080.

[6] A. Joly, O. Buisson, Logo retrieval with a contrario visual query expansion, in: Proceedings of the 17th ACM international conference on Multimedia, ACM, 2009, pp. 581–584.

[7] J. Meng, J. Yuan, Y. Jiang, N. Narasimhan, V. Vasudevan, Y. Wu, Interactive visual object search through mutual information maximization, in: Proceedings of the 18th ACM international conference on Multimedia, ACM, 2010, pp. 1147–1150.

[8] S. Romberg, R. Lienhart, Bundle min-hashing for logo recognition, in: Proceedings of the 3rd ACM conference on International conference on multimedia retrieval, ACM, 2013, pp. 113–120.

[9] S. Romberg, L. G. Pueyo, R. Lienhart, R. Van Zwol, Scalable logo recognition in real-world images, in: Proceedings of the 1st ACM International Conference on Multimedia Retrieval, ACM, 2011, p. 25.

[10] J. Revaud, M. Douze, C. Schmid, Correlation-based burstiness for logo retrieval, in: Proceedings of the 20th ACM international conference on Multimedia, ACM, 2012, pp. 965–968.

[11] S. Bianco, M. Buzzelli, D. Mazzini, R. Schettini, Logo recognition using cnn features, in: Image Analysis and ProcessingICIAP 2015, Springer, 2015, pp. 438–448.

[12] C. Eggert, A. Winschel, R. Lienhart, On the benefit of synthetic data for company logo detection, in: Proceedings of the 23rd Annual ACM Conference on Multimedia Conference, ACM, 2015, pp. 1283–1286.

[13] F. N. Iandola, A. Shen, P. Gao, K. Keutzer, Deeplogo: Hitting logo recognition with the deep neural network hammer, arXiv preprint arXiv:1510.02131.

[14] R. Girshick, J. Donahue, T. Darrell, J. Malik, Region-based convolutional networks for accurate object detection and segmentation, Pattern Analysis and Machine Intelligence, IEEE Transactions on 38 (1) (2016) 142–158.

[15] K. E. A. van de Sande, J. R. R. Uijlings, T. Gevers, A. W. M. Smeulders, Segmentation as selective search for object recognition, in: IEEE International Conference on Computer Vision, 2011.
URL `https://ivi.fnwi.uva.nl/isis/publications/2011/vandeSandeICCV2011`

[16] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, A. W. M. Smeulders, Selective search for object recognition, International Journal of Computer Vision 104 (2) (2013) 154–171.
URL `https://ivi.fnwi.uva.nl/isis/publications/2013/UijlingsIJCV2013`

[17] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, The Journal of Machine Learning Research 15 (1) (2014) 1929–1958.

[18] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, R. Fergus, Regularization of neural networks using dropconnect, in: Proceedings of the 30th International Conference on Machine Learning (ICML-13), 2013, pp. 1058–1066.

[19] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, 2012, pp. 1097–1105.

[20] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, Image quality assessment: from error visibility to structural similarity, Image Processing, IEEE Transactions on 13 (4) (2004) 600–612.