DigInPix: Visual Named-Entities Identification in Images and Videos

Pierre Letessier, Nicolas Hervé, Alexis Joly, Hakim Nabi, Mathieu Derval, Olivier Buisson

DigInPix: Visual Named-Entities Identification in Images and Videos

Pierre Letessier
Ina, France
pierre.letessier@ina.fr

Nicolas Hervé
Ina, France
nicolas.herve@ina.fr

Alexis Joly
INRIA Zenith, LIRMM, France
alexis.joly@inria.fr

Hakim Nabi, Mathieu Derval, Olivier Buisson
Ina, France
firstname.lastname@ina.fr

ABSTRACT

This paper presents an automatic system able to identify visual named-entities appearing in images and videos, among a list of 25,000 entities, aggregated from Wikipedia lists, and more specific websites. DigInPix is a generic application designed to identify different kinds of entities. In this first attempt, we only focus on logo identification (more generally on legal persons). The identification process mainly relies on an efficient CBIR system, searching in an indexed image database composed of 600,000 weak-labelled images crawled from Google Images. DigInPix proposes a responsive-design html5 interface[1] for testing purposes.

Keywords

Named-entities identification; CBIR; logos; images; videos

1. INTRODUCTION

Named-entities recognition and disambiguation in text documents is a well-known problem of the Natural Language Processing community, and there now exists some pretty good solutions, such as those submitted to the Microsoft Entity Recognition and Disambiguation Challenge [2]. In the same time, a lot of progresses have been made on image (instance) retrieval (logos, buildings, etc.) [1, 6, 10], and image classification, as demonstrated by recent results on the popular ImageNet dataset [9, 11, 12], or on fine-grained and instance classification tasks [3, 5]. Very few works did however specifically address the problem of recognizing very large numbers of named-entities (such as firms, government bodies, political parties, societies, associations, etc.) in images and videos. It can actually be very tricky to visually identify a named-entity based on global representations, especially when its visual representation is very small (tens of pixels). Furthermore, while a named-entity can be textually represented by a small synset, a visual synset has to deal with many variations (definition, encoding, scale, rotation, illumination, etc.).

Most existing related works were limited to the recognition of few tens of classes (37 logos in BelgaLogos dataset [6], 32 logos in FlickrLogos dataset [10]), each targeted entity being modeled by a single logotype. The only achievement dealing with more realistic numbers of entities and richer visual identities is the recent work of Leveau et al. [7]. The authors actually reported in that paper an experiment with more than 5,000 entities represented by a (noisy) training set of 370K images automatically crawled through a web search engine. The key point of their method it to rely on an efficient instance-based classification scheme (with fine-grained geometric constraints) rather than on the offline training of a supervised classifier on common aggregated image representations (such as Bag-of-words, VLAD or Fisher Vectors features [9]). They did show that such scheme outperforms state-of-the-art classification techniques on several challenging datasets while being much more scalable.

This paper describes an operational and interactive web application following a similar approach but at a larger scale (i.e. 25,000 entities and 600,000 images) and with additional features such as browsing functionalities and the possibility to process video contents. The core engine makes use of local descriptors hashing techniques and a scalable indexing and search structure based on hash tables. This allows matching all local descriptors one by one in the full index and favors a more precise matching of small objects in highly cluttered pictures. Using geometry in addition to this raw scheme further improves the search performances (thanks to a RANSAC-like spatial reranking algorithm) and the instance-based classifier is shown to provide very promising automatic annotation performances.

[1] http://diginpix.ina.fr

2. PROPOSED APPROACH

2.1 Creation of the dictionaries

We call "dictionary" a list of named-entities, grouped together because they belong to a high level concept (e.g. legal persons, physical persons, paintings, buildings, etc.). With each entity, we associate a set of images trying to represent its whole visual diversity. The easiest way to create a dictionary is to import an existing database, but such databases are quite infrequent and incomplete (especially when collaborative).

So the most common way to create this kind of dictionary is to crawl the web and download images. To build our legal persons dictionary, we first collected a list of 25,000 entities found on Wikipedia lists, Top-N company rankings, and specific websites talking about sport clubs, cars, political parties, etc.

For all entities, we then used a popular image search engine, querying the named-entities textual representations associated with disambiguation keywords like "logo", or the entity categories, in order to obtain their visual representation. It is important to notice that the results are often very noisy, except for the most common entities, and we can only consider the downloaded images as weak-labelled, which is the cause of most of the identification mistakes.

2.2 Video vs. image pipeline

Both video and image contents can be uploaded and processed by the application (only images for public users). When dealing with videos, keyframes are first extracted whenever the content changes significantly, i.e. new shot, moving camera, new objects appearing, etc., thanks to a keyframe detector using grid-based histograms of LBP features. The time between two keyframes typically varies from 100ms to 30s. All the detected keyframes are then processed one by one in the same way than single images. They are first issued as queries to the visual search engine described within subsection 2.2.1. In a second step, the returned matched images are passed to the instance-based classifier described in section 2.2.2). It returns a list of the most probable named-entities visible in each query image (the automatic annotations). In the case of videos, a last step is finally completed to merge the automatic annotations of all keyframes (see section 2.2.2). It returns a ranked list of all the entities detected in the video according to their number of occurrences.

2.2.1 Visual Search Engine

All images are described with SIFT features [8], which are then compressed to 128-bit binary hash codes. This is done by first computing a principal component analysis in the original feature space followed by a binary quantization. The binary quantization of each component of a hash code $\mathbf{h}(\mathbf{x})$ is computed as:

$$h_j = sgn(\mathbf{a_j}^T\mathbf{x} - \mu_\mathbf{j})$$

where $\mathbf{a_j}$ is the j-th eigenvector resulting from the principal component analysis and

$$\mu_\mathbf{j} = \frac{1}{N}\sum_\mathbf{x} \mathbf{a_j}^T\mathbf{x}$$

is the mean of all the projected values. The distance between any two features $\mathbf{x}$ and $\mathbf{z}$ can then be efficiently approximated by the Hamming distance between $\mathbf{h}(\mathbf{z})$ and $\mathbf{h}(\mathbf{x})$.

To avoid scanning the whole dataset, the hash codes $\mathbf{h}(\mathbf{x})$ derived from the local features of the training set are then indexed in a hash table whose keys are the t-length prefix of the hash codes $\mathbf{h}(\mathbf{x})$. At search time, the hash code $\mathbf{h}(\mathbf{q})$ of a query feature $\mathbf{q}$ is computed as well as its t-length prefix. We then use a probabilistic multi-probe search algorithm inspired by the one of [4] and allowing to efficiently return the approximate K-Nearest Neighbors (K-NN) of each query feature. We however use a simpler search model than the one of [4], based on a normal distribution with independent components parameterized by a single vector σ that is trained over the exact nearest neighbors of the training samples.
The raw visual matches returned by the approximate K-NN search are finally filtered by a spatial consistency checking. We therefore estimate an affine transformation (by a RANSAC algorithm) between the query image and each matched image in the training set (i.e. the ones having at least two matches in the set of the returned approximate K-NN's). At the end of this step, we have a set of M^Q images similar to the query image Q, each associated with a similarity score S_i^Q.

2.2.2 Instance-based classifier

We compute the reliability scores of the e^{th} entity for a query image Q by the following equation:

$$R_e^Q = \sum_{i=1}^{M_e^Q} S_i^Q$$

where M_e^Q is the size of the subset of images associated in the dictionary with the e^{th} name-entity, and retrieved by the previously described CBIR system.

This score has to be normalised between 0 and 100%, in order to be displayed and understood by the users. It is done with:

$$\hat{R}_e^Q = \tanh\left(\max\left(\left\{\frac{R_e^Q - \alpha}{\beta}, 0\right\}\right)\right)$$

where $\alpha = 2$ and $\beta = 80$, both empirically set, after testing with a pool of users.

In the Graphical User Interface (GUI), we only display the entities with a score $\hat{R}_e^Q$ greater than 3%. This threshold has been chosen very low to favor a high recall (preferred by expert users), rather than a high precision (more adapted to non expert users).

When dealing with a video, we display the detected entities in each keyframe following the same paradigm but we also display a global score for every entities detected in the v^{th} video. This global score is computed as the maximum score across all the keyframes:

$$\hat{G}_e^V = \max_{Q \in V} \hat{R}_e^Q$$

Considering that global score, we again filter the displayed entities by keeping only those in the top 25 reliability scores $\hat{G}_e^V$, and also those having a score higher than 50%, even if they are not in the top 25.

2.2.3 Results

In order to evaluate our system, we built a test dataset of 2,000 images from Flickr, containing 285 different named-entities. It was built by querying the Flickr text search

engine with the name of 100 popular entities appearing in French sport events (Tour de France, Roland Garros, 24h du Mans, etc.). As many other entities also appeared in these images, we had to manually annotate them and finally obtained a groundtruth on 285 entities. The graph in the figure 1 shows the precision/recall curve achieved by our named-entity detection scheme on these 2,000 images (using the web crawl of 25,000 entities and 600,000 images as training set). The curves were obtained by varying the threshold on the reliability score $\hat{R}_e^Q$. We can see that with this quite realistic test dataset we achieve a precision of 80% for a recall of 30%. We can also note that we cannot perform more than 40% of recall, probably due to the amount of bad-labelled images in the dictionary, and to the very small entities we have in the test images. These images are not very big either: their average-size is 214 Kpx.

In the case of a constant load (analysing the 2,000 test images as soon as possible), DigInPix can process an image in less than 2 seconds. In real conditions, it can be slightly longer, due to the server load. Processing time can even be as much as tens of seconds for large images, since the algorithm complexity is almost linear in the number of visual features (i.e. the image size).

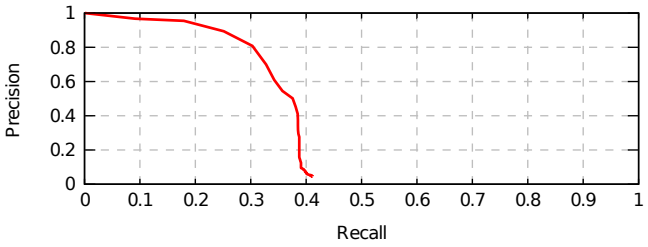

Figure 1: Evaluation

3. DEMONSTRATOR

The DigInPix's GUI is a responsive and flat design web-application. All actions are accessible on the same single page. There are two main actions to do with DigInPix: the first one is to analyse one's own photo, and the second is to browse the dictionaries as well as the videos of the sport collection taken from Ina's archives (automatically annotated by our named-entity detection scheme).

3.1 Analysing one's own image

Users can upload their own image (or video) from a local drive or from an image url. After the named-entity identification process, the GUI displays the analysed image, with the list of detected entities, and the reliability score $\hat{R}_e^Q$ for each of them. Figure 2 presents an example of an analysed image, where DigInPix identified six named-entities, sorted by reliability. The user then has the possibility to click on any of them to access to all their informations (see section 3.2), and can jump to the other documents in which it has been identified.

3.2 Browsing data

There are two categories of browsable datas in DigInPix: the dictionaries data and the analysed documents.

Users can browse the named-entities in each dictionary, sort them by label, or by their number of detections in the

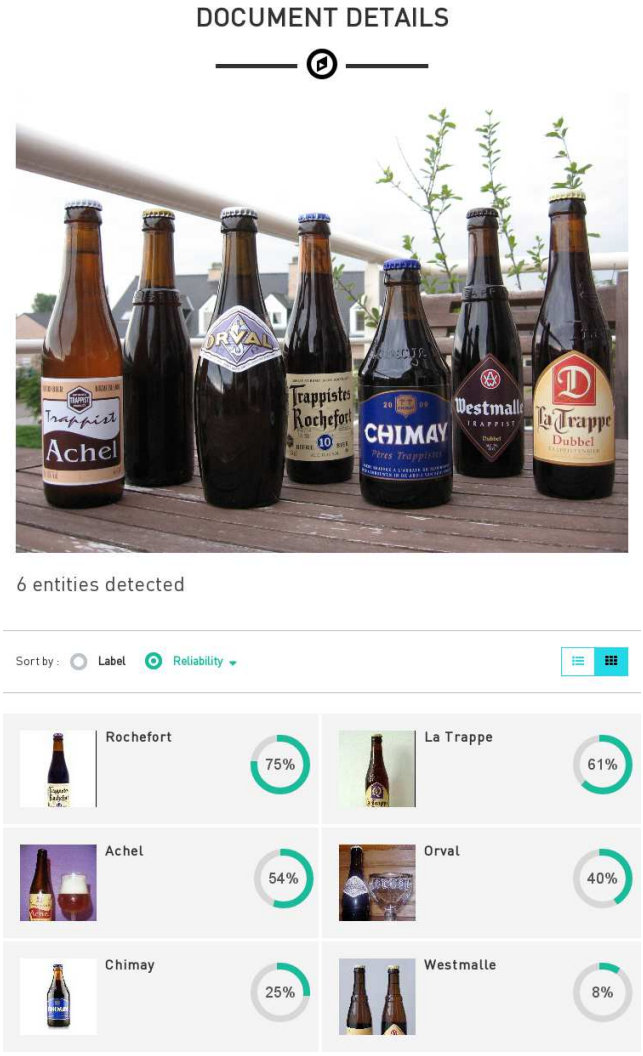

Figure 2: An analysed image with 6 detected entities

whole set of analysed documents, and filter them by their first character. For each entity, DigInPix display the list of representative images crawled from the web, and used by the algorithm to recognize this entity.

The analysed documents (images or videos) are distributed in coherent collections. Initially, only one demo collection of sports videos is proposed. The user can click any of them to see the list of detected named-entities. If the document is a video, DigInPix displays it in an open-source video player[2] enriched with timelines showing the time stamps at which each entity was detected. These time stamps are symbolised by clickable points on the line, as shown in the figure 3. Many buttons are available to handle the player, and the little gear in the bottom-right corner enables or disables the tooltips explaining their use.

4. CONCLUSIONS

DigInPix is a first attempt to provide public access to our large-scale named-entities identification algorithm, and we think that there is room for further improvements. The first

[2]http://ina-foss.github.io/amalia.js

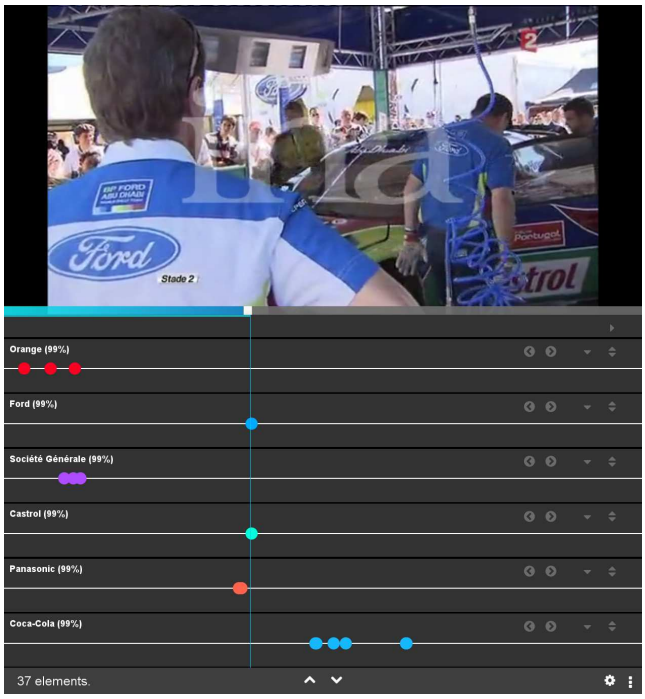

Figure 3: The video player with timelines

one could be to add new dictionaries, in particular buildings, places, paintings and faces, among many others. Another great improvement would be to interactively clean or revise the bad-labelled images which cause most of the identification errors. For this, we intend to develop new interfaces to provide users an easy way to participate, and give feedback. Finally, we could improve the user experience by drawing the precise location of the detected named-entities in the images and videos, as allowed by our video player, and also track the entities between the keyframes.

5. ACKNOWLEDGEMENT

We wish to thank the developers who participated in this project.

6. REFERENCES

[1] R. Arandjelovic and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012.

[2] D. Carmel, M.-W. Chang, E. Gabrilovich, B.-J. P. Hsu, and K. Wang. ERD 2014: Entity recognition and disambiguation challenge. *SIGIR Forum*, 2014.

[3] P.-H. Gosselin, N. Murray, H. Jégou, and F. Perronnin. Revisiting the fisher vector for fine-grained classification. *Pattern Recognition Letters*, 2014.

[4] A. Joly and O. Buisson. A posteriori multi-probe locality sensitive hashing. In *ACM MM*, 2008.

[5] J. Krapac, F. Perronnin, T. Furon, and H. Jégou. Instance classification with prototype selection. In *Proceedings of International Conference on Multimedia Retrieval*, page 431. ACM, 2014.

[6] P. Letessier, O. Buisson, and A. Joly. Consistent visual words mining with adaptive sampling. In *ICMR*, 2011.

[7] V. Leveau, A. Joly, O. Buisson, P. Letessier, and P. Valduriez. Recognizing thousands of legal entities through instance-based visual classification. In *Proceedings of the ACM International Conference on Multimedia*, pages 1029–1032. ACM, 2014.

[8] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999.

[9] F. Perronnin, J. Sánchez, and T. Mensik. Improving the fisher kernel for large-scale image classification. In *ECCV*, 2010.

[10] S. Romberg, L. G. Pueyo, R. Lienhart, and R. Van Zwol. Scalable logo recognition in real-world images. In *ICMR*, 2011.

[11] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint:1409.1556*, 2014.

[12] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.