Do deep features retrieve X?: A tool for quick inspection of deep visual similarities

Julieta Martinez
julm@cs.ubc.ca

November 2015

1 Introduction

Computer vision is often associated with computational object recognition; that is, given an image of an object, predicting a posterior probability over a label set that corresponds to the objects depicted in the image. While it is true that this is the visual task that has experienced the largest improvement over the last couple of years, largely thanks to variations on the seminal work of Krizhevsky and colleagues [Krizhevsky et al., 2012] based on deep convolutional neural networks, other problems in computer vision have extensively benefited from the use of features extracted from deep networks.

One such common problem in visual recognition is visual retrieval, which is defined as the task of, given a query image q and a dataset of images $D = \{x_1, x_2, \ldots, x_n\}$, producing a total order that ranks the elements in D with respect to their similarity to q. It is expected that this order will produce a ranking that is semantically meaningful to people, such as retrieving objects of the same class, or with otherwise similar visual characteristics. The state of the art in this problem is heavily driven by benchmarks of object instance retrieval, and is defined by the use of features extracted from deep neural networks [Razavian et al., 2014, Babenko and Lempitsky, 2015].

While most retrieval benchmarks are focused on instance retrieval, we argue that a sense of visual similarity is often loosely defined, and in practice is not limited to exact object instance matching. The goal of this project is to create a tool that allows researchers to quickly verify if a technique for image retrieval works well for other tasks. Since the state of the art in visual retrieval is currently defined by deep features, we think that in practice our tool will allow researchers to quickly answer the question *do deep features work for retrieving X?*.

2 Datasets

We plan to use two datasets, each with different and unconventional tasks

- **Faces in things (FiT) dataset.** We plan to collect a dataset by downloading the images of the twitter account `@FacesPics`[1], which mostly contains images of objects where people tend to see a face. The task on this dataset is determining whether deep features can retrieve similar "facial expressions" in objects. For example, does a query of a grumpy toaster return other grumpy appliances? This is a medium size dataset of approximately 800 images.

- **Vision papers (VP) dataset.** We will download the papers from the ICCV 2013 and CVPR 2013, 2014 and 2015 conferences, which are publicly available[2], and create images by concatenating the pages of the papers. A task in this datasets is to quickly check if deep features capture some similarity in the papers; for example, does querying with a paper on optical flow return other papers on the same topic? or, does querying with papers authored by the top computer vision researcher Florence Fictitious return other papers by her group? This will be a large dataset of 2 000+ images.

3 Personal expertise

I am familiar with the task of visual retrieval, but I have only done video to mocap matching [Gupta et al., 2014, He et al., 2016], not single image matching. Over the last year, I have worked on a technique to improve the speed and accuracy of approximate nearest neighbour search in high dimensions based on multi-codebook quantization, which is often the computational bottleneck of visual retrieval systems (and I might re-implement to speed up matching) – said work is currently under review for CVPR 2016.

I work in computer vision and I am often tempted to investigate whether deep features can capture unconventional visual similarities, but I am also often deterred by the need to set up a whole retrieval infrastructure and the difficulty to assess the quality of the matchings.

4 Proposed solution

We propose to implement a system similar to VisDB. We use the what-why-how framework to describe our system in Table 1.

5 Scenario of use

We next enumerate a series of steps that the user would perform in a typical exploration scenario, where the user has the question *do deep features retrieve happy faces in things?* in mind. A sketch of our system in this use case is shown in Figure 1.

[1] https://twitter.com/FacesPics
[2] http://www.cv-foundation.org/openaccess/menu.py

1. The user points their browser to the url of our viz app. We have already included the faces in pics images and features for them.

2. Initially, the pictures are shown in a randomized space-filling layout. The user thus has an overview of the pictures in the database.

3. Before querying, the user sets an appropriate image size that gives enough context, and activates the fisheye lens to explore parts of the dataset in detail.

4. The user double-clicks the image that they want to use as query. This is an image of a "happy vacuum cleaner" (see Figure 1). The image is smoothly moved to the center, while the query runs in the background. This step should take less than a second.

5. Once the ranking is obtained, the images are rearranged from their current position their position around the query according to the ranking.

6. The user explores the new arrangement of images, looking in detail at some of the top matches. They repeat the process for a few more images of happy objects, and confirm that deep features are indeed good at retrieving happy faces in things.

Some variations include (i) indicating another set of images and features to use with a `csv` file of urls and features, (ii) making the initial arrangement, instead of random, reflect a dimensinoality-reduced embedding of the high-dimensional features.

6 Proposed implementation approach

I plan to do the application web-based, using D3 (javascript). This is with the goal of maximizing the number of people who will potentially look at the app.

7 Milestones and schedule

- Data gathering of FiT and VP datasets. (Nov. 17)
- Implement a quick feature matching system in javascript. I'll probably use product quantization [Jégou et al., 2011], an easy-to-implement method for fast approximate nearest neighbour search.
- Gaining familiarity with D3. This involves learning how to...
  - represent images with varying sizes (Nov. 19)
  - make the images not overlap each with other (Nov. 21)
  - make the images fill the space (Nov. 19)
  - use a fisheye lens (Nov. 24)

Figure 1: An example of what our interface would look like, not at scale (we will show many more images), and without the fisheye lens activated. The query is shown in yellow, and the most relevant images are shown around it. In this case, we assume that deep features actually are good at retrieving facial expressions, and a "happy vacuum cleaner" retrieves other happy items such as a "happy lemon loaf", a "happy chronometer" and a "happy mop" among the top matches.

  - animate the images to move them in the space according to a ranking (Nov. 26)
  - respond to a double-click by the user to trigger the reranking and rearrangement of the database elements (Nov 28)

- Integrate the knowledge above into a single system (Dec 5). Run tests with the FiV and VP datasets.

8 Previous work

Preliminarily, my work is loosely inspired by VisDB [Keim et al., 1994], but applied to images From VisDB, I am borrowing the idea of spiral arrangement, and the tiling of images. There is a large body of work on visualizations that facilitated the exploration of *personal* photo collections (see [Huynh et al., 2005] and references therein). The work dates from more than a decade ago, and

recognizes that visual retrieval systems are not very good at conveying semantic similarities. Thus, they resort to time-stamps and text tags to organize large collections of pictures. In contrast, my tool is used to argue that visual retrieval is now mature enough to be used as the primary tool to search and explore large image collections.

I found one journal article that describes a system similar to what I want to develop, and is actually focused on non-personal image collections [Wang et al., 2013]. The authors propose iMap, a tree-based visualization that aims to both fill the space and also respond quickly to additions and removal to the photo collection, and demonstrated their tool on a dataset of pictures of astronomical objects. iMap supports a spiral layout (see their Figure 2), with a space-filling layout and also incorporates text-based tag similarity. Given a query, the autrhos use a GPU to maintain interactivity; we will instead use a web interface, and hope to achieve real-time retrievla speeds using approximate distance computations with multi-codebook quantization.

Related to the focus+context part of our work, which we will achieve with a fisheye lens, previous work by Chen and colleagues [Chen et al., 2013] focused on benchmarking different ways to expand photos (where the user wants to focus) surrounded by small tiles of other imagess (that provide context). They investigated a series of algorithms that smootly expand the desired picture, while rearranging the pictures around them, and found that 2 methods, called sliding and expanding are preferred by the users. These methods preserve neighbourhoods in the original layout, but do not fill the space. If time allows, we would like to experiment with similar focus techniques in our tool.

References

[Babenko and Lempitsky, 2015] Babenko, A. and Lempitsky, V. (2015). Aggregating deep convolutional features for image retrieval. *arXiv preprint arXiv:1510.07493*.

[Chen et al., 2013] Chen, J., Xu, Y., Turk, G., and Stasko, J. (2013). Easyzoom: Zoom-in-context views for exploring large collections of images.

[Gupta et al., 2014] Gupta, A., Martinez, J. L., Little, J. J., and Woodham, R. J. (2014). 3d pose from motion for cross-view action recognition via non-linear circulant temporal encoding. In *CVPR*.

[He et al., 2016] He, J., Gupta, A., Martinez, J., Little, J. J., and Woodham, R. J. (2016). Efficient video-based retrieval of motion capture with flexible alignment. In *WACV*.

[Huynh et al., 2005] Huynh, D. F., Drucker, S. M., Baudisch, P., and Wong, C. (2005). Time quilt: scaling up zoomable photo browsers for large, unstructured photo collections. In *CHI'05 extended abstracts on Human factors in computing systems*, pages 1937–1940. ACM.

[Jégou et al., 2011] Jégou, H., Douze, M., and Schmid, C. (2011). Product quantization for nearest neighbor search. *TPAMI*, 33(1).

[Keim et al., 1994] Keim, D., Kriegel, H.-P., et al. (1994). Visdb: Database exploration using multidimensional visualization. *Computer Graphics and Applications, IEEE*, 14(5):40–49.

[Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *NIPS*.

[Razavian et al., 2014] Razavian, A. S., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). Cnn features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 512–519. IEEE.

[Torralba et al., 2008] Torralba, A., Fergus, R., and Freeman, W. T. (2008). 80 million tiny images: A large data set for nonparametric object and scene recognition. *TPAMI*, 30(11).

[Wang et al., 2013] Wang, C., Reese, J. P., Zhang, H., Tao, J., Gu, Y., Ma, J., and Nemiroff, R. J. (2013). Similarity-based visualization of large image collections. *Information Visualization*, page 1473871613498519.

Do deep features retrieve X?	
What: Data	A dataset D of images $D = \{x_1, x_2, \ldots, x_n\}$ and a set of image queries $Q = \{q_1, q_2, \ldots, q_m\}$. Potentially, $Q \subseteq D$.
What: Derived	n d-dimensional features representing the database, and m d-dimensional features for the queries. A similarity measure for the derived features $S : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$. Typically, for deep features $S(f_1, f_2) = \|f_1 - f_2\|_2^2$ or $S(f_1, f_2) = \langle f_1, f_2 \rangle$.
Why: Tasks	Browse / explore. If the user has an image prototype in mind, but can only find a similar image to what they want, they can use our system to browse the closest matches to their image in our system. The user might also query the database with different images and inspect the top matches in search for a pattern (e.g., whether deep features match facial expressions).
How: Encode	Each image will be displayed as a minified version of its central crop. It has been shown that a size of 32×32 is sufficient to recognize objects in pictures [Torralba et al., 2008], so we will use this as a lower limit for the encoding size. The user will be able to make the images bigger or smaller, and I still have to find an algorithm to tile the images in a space-filling manner.
How: Facet	The query will be shown at the center of the display, and the returned database entries will be displayed around it, wrapping in circles as in VisDB.
How: Focus+Context	There will be an optional fisheye lens for the user to move around and see the retrieved images in more detail. The idea is that the user will activate this feature when the individual image size is too small to distinguish the desired features in individual images, but wants to preserve the context of the whole ranking.
Scale	TBD, since this will depend directly on the size that the user chooses for each encoded image.)

Table 1: What-why-how analysis of the proposed infoviz solution.