Finding images with similar content elements based on LSH and ANN

FINAL PROGRESS REPORT

Markus Laubenthal, Lennard Alms
Supervisor: PD Dr. Michael Mock

Problem Definition

Build a python **framework** that takes a huge **database of images** as input and outputs a **structure** that can be **queried** for **approximate nearest neighbours** of an image or a slice of an image using **locality sensitive hashing**.

Problem Example

Image source: [1]

Given an image with trash containers

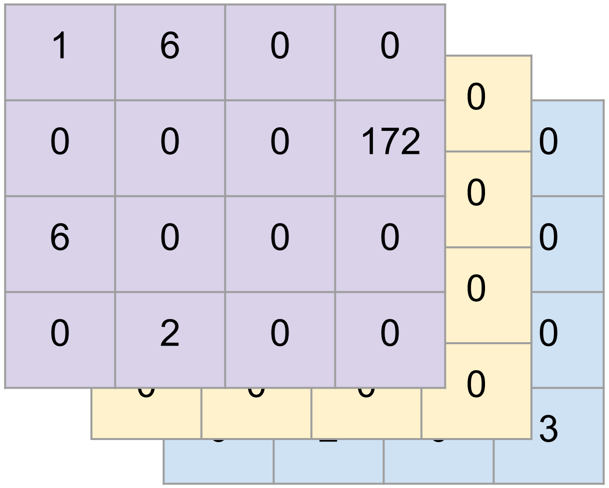
Reason:
Segmentation neural networks sometimes confuse trash containers with cars.

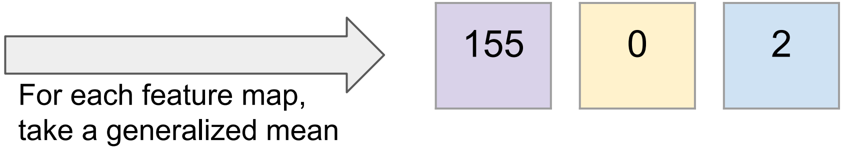
Find images in the database that also contain trash containers

Generalized Mean Pooling (GeM)

VGG16

Outputs 512 feature maps size H x W

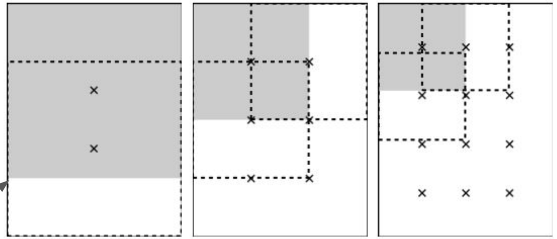


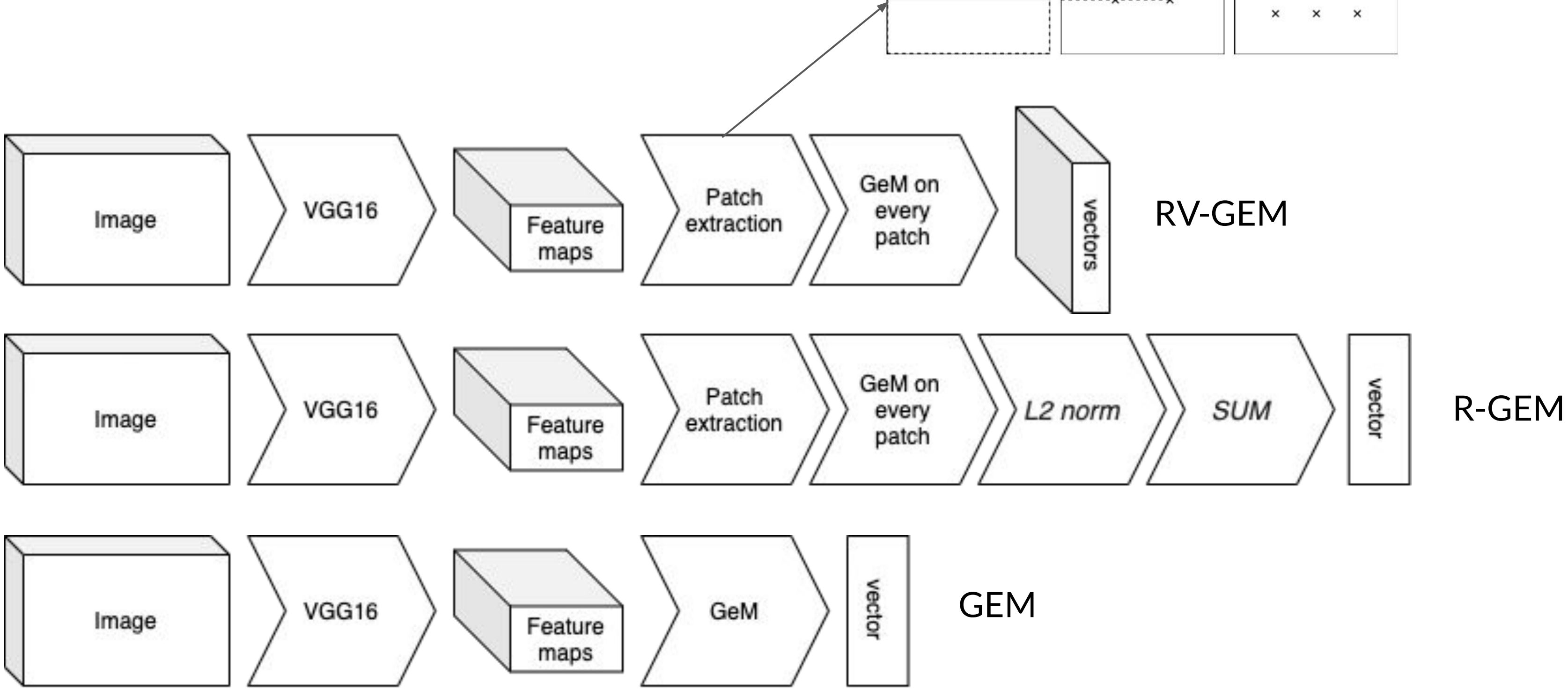


GeM

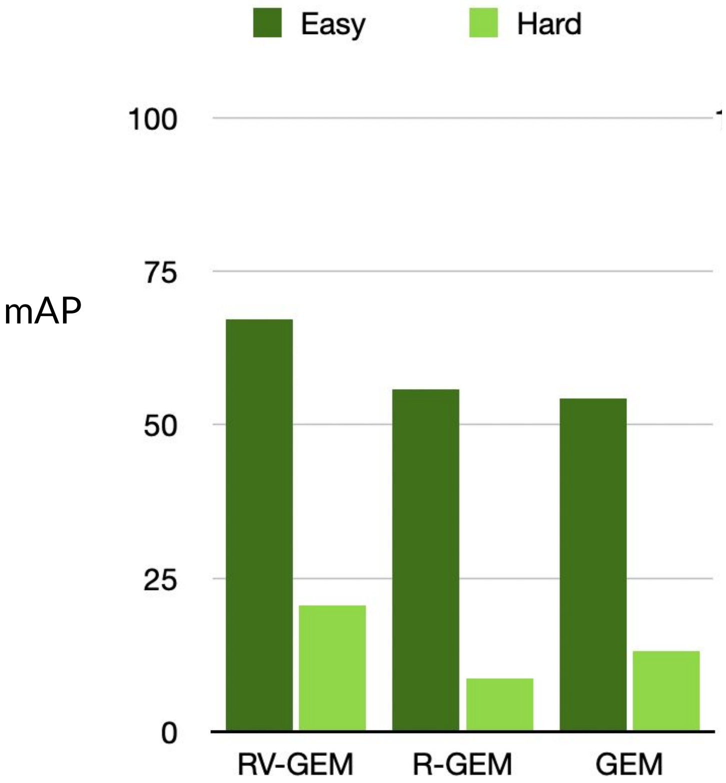
Outputs 512 feature maps of size 1 x 1
(Vector with 512 dimensions)

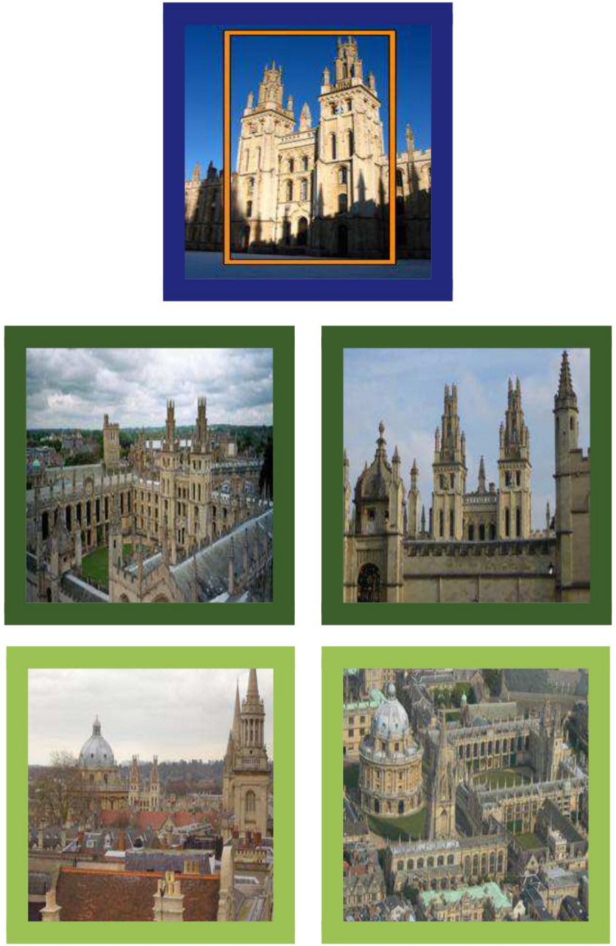
Feature Extraction Pipelines





Evaluation





[14]

Definitions

Near-Duplicate images:

Images that share the same content with only little modification
(e.g. all images that contain the same image of a horse)

Similar images:

Images that contain the same class of images
(e.g all images that contain any type of horse)

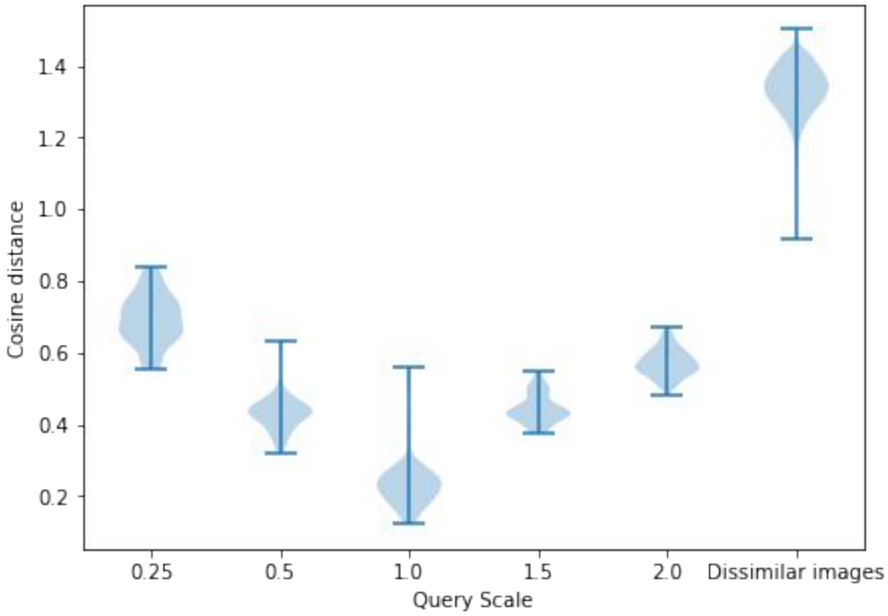
Query image:

The image for that we want to find similar content in a database

Target images:

All images that should be returned for a given query image

Query Scaling



Cosine distance of query image to target images

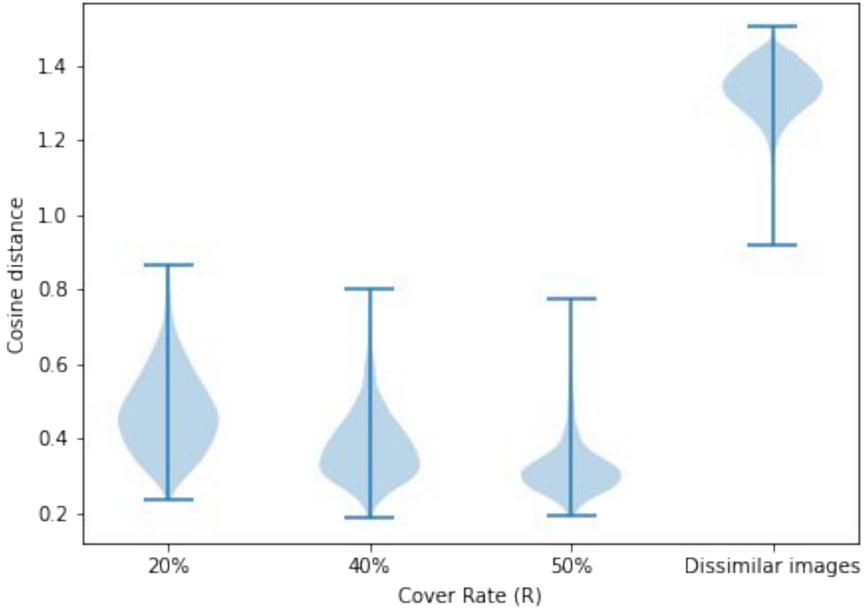
Effect of the query image and target image being differently scaled in a database with unscaled images (for near-duplicate images)

We observe higher cosine distances if query scale and target scale do not match up.

We achieve result improvement by querying the database with multiple differently scaled queries.

If query and target scale match, threshold of 0.4 distinguishes most near-duplicates from all dissimilar images

Patch Extraction of multiple sizes



Cosine distance of query image to target images

Cover Rate R determines how much percentage of the whole image was covered by the target image.

If target image has a big noisy background, the cosine distance increases

We improve performance by sampling multiple patch sizes from the GeM layer

If target image covers most of the area, threshold 0.4 contains most of our near duplicates

Examples:

Query image

Database

Locality Sensitive Hashing with random hyperplanes

$$H_{\cos} \text{ is } (\boxed{0.4}, \boxed{1.2}, \boxed{0.87}, \boxed{0.62}) - \text{sensitive}$$

Near-Duplicate Threshold

Upper threshold
(Dissimilar images)

Probability that near duplicates are in the same half space (bucket)

Probability that dissimilar images are in the same half space (bucket)

Locality Sensitive Hashing with random hyperplanes

AND-OR Amplification with random hyperplanes:

- Select k random hyperplanes
- Two images are stored in the same bucket if for all k hyperplanes, they lie in the same halfspace
- For every image, store it in L hash tables with k random hyperplanes each.

For L=50, k=20 (experimental result), this changes the sensitivity to:

$$H_{\cos,\, L=50,\, k=20} \text{ is } (0.4,\ 1.2,\ \boxed{0.97,\ 0.003}) - \text{sensitive}$$

Results

Results on a dataset that contains 10% augmented images with random images of horses

Model	GeM (3) (Exhaustive Search)			+ LSH, k=20, L=50				
Retrieval Type	R	P	A	R	P	A	% of DB	% below δ
Duplicates, δ=0.4	0.97	1.00	1.00	0.97	1.00	1.00	1.5%	26%
Duplicates, δ=0.6	0.99	0.86	0.99	0.99	0.93	0.99		32%
Similar Images, δ=0.8	0.38	1.00	0.92	0.12	1.00	0.9		45%
Similar Images, δ=1.0	0.91	1.00	0.92	0.15	1.00	0.92		65%
Similar Images, δ=1.2	0.96	0.98	0.92	0.15	0.97	0.94		70%

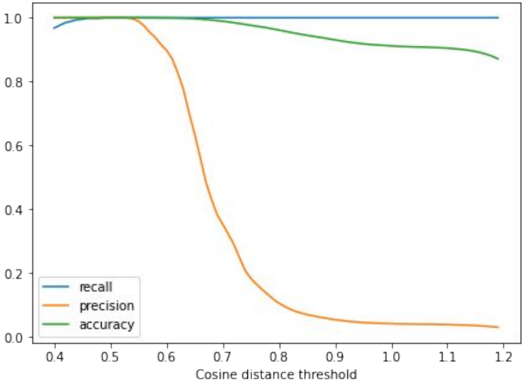
In a database with `200.000` images
Query time of exhaustive search: ~230 ms
Query time of LSH Method: ~19 ms

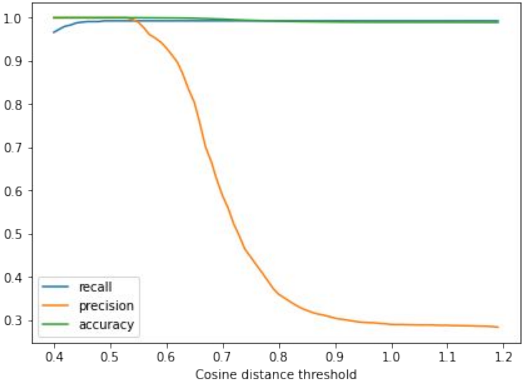
Machine: Google Colab Free plan
2 virtual cores

Exhaustive Search

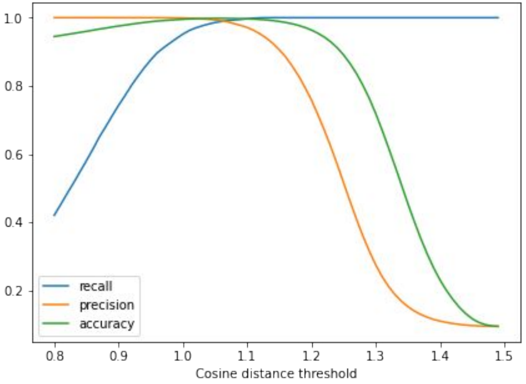
with LSH

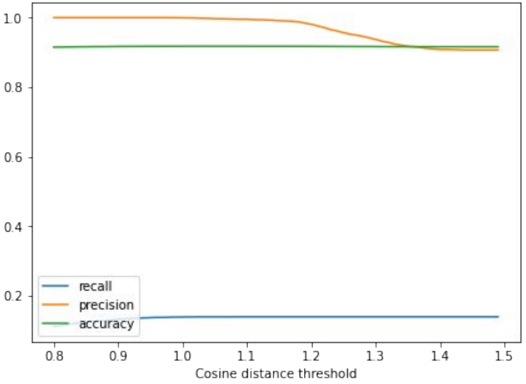
Near Duplicate Retrieval





Similar Image Retrieval



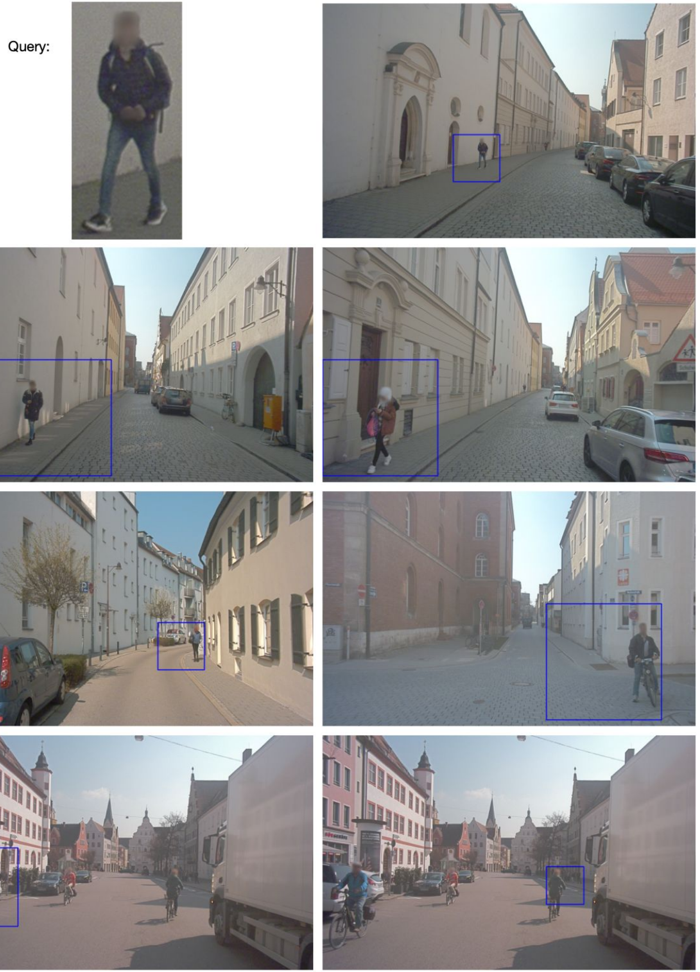


Demo

Demo (backup)

Query:

Demo (backup)



References

[1] A2d2 dataset. `https://www.a2d2.audi/a2d2/en/dataset.html`. Accessed: 2020-03-07.

[2] Annoy library. `https://github.com/spotify/annoy`. Accessed: 2020-03-07.

[3] Falconn documentation. `https://github.com/FALCONN-LIB/FALCONN/wiki/How-to-Use-FALCONN`. Accessed: 2020-03-07.

[4] Horse or human dataset. `https://www.kaggle.com/sanikamal/horses-or-humans-dataset`. Accessed: 2020-03-07.

[5] Pablo F Alcantarilla and T Solutions. Fast explicit diffusion for accelerated features in nonlinear scale spaces. *IEEE Trans. Patt. Anal. Mach. Intell*, 34(7):1281–1298, 2011.

[6] Alexandr Andoni, Piotr Indyk, Thijs Laarhoven, Ilya Razenshteyn, and Ludwig Schmidt. Practical and optimal lsh for angular distance, 2015.

[7] Jonathan Delhumeau, Philippe-Henri Gosselin, Hervé Jégou, and Patrick Pérez. Revisiting the vlad image representation. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 653–656, 2013.

[8] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus. End-to-end learning of deep visual representations for image retrieval. *International Journal of Computer Vision*, 124(2):237–254, 2017.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, page 604–613, New York, NY, USA, 1998. Association for Computing Machinery.

[11] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3304–3311. IEEE, 2010.

[12] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[13] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-scale image retrieval with attentive deep local features. In *Proceedings of the IEEE international conference on computer vision*, pages 3456–3465, 2017.

[14] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[15] F. Radenović, A. Iscen, G. Tolias, Y. Avrithis, and O. Chum. Revisiting oxford and paris: Large-scale image retrieval benchmarking. In *CVPR*, 2018.

[16] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1655–1668, 2018.

[17] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[18] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *Computer Vision, IEEE International Conference on*, volume 3, pages 1470–1470. IEEE Computer Society, 2003.

[19] Giorgos Tolias, Ronan Sicre, and Hervé Jégou. Particular object retrieval with integral max-pooling of cnn activations. *arXiv preprint arXiv:1511.05879*, 2015.

[20] Esteban Uriza, Francisco Roberto Gómez Fernández, and Martín Rais. Efficient large-scale image search with a vocabulary tree. 2018.