Visual versus Textual Embedding for Video Retrieval

Danny Francis, Paul Pidou, Bernard Merialdo, and Benoît Huet

EURECOM, Campus SophiaTech
450 Route des Chappes, 06410 Biot, France
{danny.francis,paul.pidou,bernard.merialdo,benoit.huet}@eurecom.fr
http://www.eurecom.fr/en

Abstract. This paper compares several approaches of natural language access to video databases. We present two main strategies. The first one is visual, and consists in comparing keyframes with images retrieved from Google Images. The second one is textual and consists in generating a text-based description of the keyframes, and comparing these descriptions with the query. We study the effect of several parameters and find out that substantial improvement is possible by choosing the right strategy for a given topic. Finally we investigate a method for choosing the right approach for a given topic.

1 Introduction

Managing large databases of multimedia content such as videos is more and more a topical issue. The problem is that in such databases, video content cannot be manually annotated. Therefore, searching videos in big video databases is especially a matter of using self-contained information. In addition, the lack of structure in such databases implies that queries must be built as freely as possible; preferably in natural language. Pattern Recognition techniques seem to be nowadays one of the most adapted to such problems: they offer great performance in both natural language and computer vision.

The National Institute of Standards and Technology (NIST) organizes every year TRECVID [**?**], an international evaluation campaign on video information retrieval. In 2016, a new task called "Ad-Hoc Video Search" (AVS) was proposed. The goal was to process Natural Language queries to retrieve relevant shots from a large database containing about 600 hours of video, representing 300,000 shots. Participating teams were provided with 30 test topics, and could submit up to four runs. Each run had to be a list of 1,000 shots, ranked from the most relevant shot to the least one. The NIST performed a manual evaluation of the runs and gave the Mean Inferred Average Precision (an approximation of the Mean Average Precision) for each of them.

We took part in the AVS task and even though we were allowed to submit only four runs [**?**] we implemented many possible systems. The data that were used to evaluate runs were published recently, therefore we could evaluate all

our systems. In this paper we present and analyze the performances of all the approaches we implemented for the AVS task. These approaches were built upon two orthogonal strategies:

- Strategy 1: use the natural language queries to take images from a web search engine, and compare keyframes with these images to select the best ones.
- Strategy 2: generate a text-based description of the keyframes and compare them to the queries.

We implemented these strategies using tools that are freely available from the Internet.

- We got images from the Google ImageSearch engine [**?**] to implement Strategy 1: we entered a text query and the search engine returned a list of images related to it. The implementation of this search engine is not open-source, but we think that it is likely to be based on the textual content that surrounds images.
- To get a text description from an image, we used several tools:
  - the VGG Deep Networks [**?**], which have been trained on part of the ImageNet database and can analyze an image to provide scores for 1,000 predefined concepts,
  - the ImageNet Shuffle [**?**], which provides classifiers trained on a larger share of the ImageNet database, and analyze images to produce scores for up to 13,000 concepts,
  - the NeuralTalk [**?**] package, which generates sentences describing the visual content of images.
- To compare visual contents, we compute a visual feature vector for an image by applying the VGG Deep Network to each image and extracting the outputs of the one-before-last and two-before-last layers, to build visual vectors. The similarity between visual vectors is computed as the usual scalar product, sometimes with normalization.
- To compare textual content, we use the GloVe vector representations of words [**?**], to build a textual vector from either the topic description, the concept name or the descriptive sentence. The similarity between textual vectors is again computed as the usual scalar product.

We implemented several types of runs inasmuch as many combinations of these modules are possible, as well as different values of the parameters involved can be chosen. These runs boil down to three types: runs based on Strategy 1, runs based on Strategy 2 and runs mixing both strategies.

We noticed that performances where not the same depending on topics and on some other parameters. We will elaborate on what worked and what did not.

2 State of the Art

Some previous works have shown that words could be successfully represented by vectors [**?**,**?**]. It has also been shown that sentences could also be efficiently

represented as vectors by averaging the vectors of its words [**?**]. In particular, averaging word vectors has been shown to be more efficient in some tasks than more complex systems [**?**].

Natural Language Processing and Computer Vision problems can also be addressed with Deep Learning techniques. In Natural Language Processing, Recurrent Neural Networks (RNN) [**?**] are now widely used to model languages: sentences are divided into word vectors that are processed one after the other by such networks. This kind of Neural Networks is particularly used for sentence generation [**?**] and for sentence embedding [**?**].

Convolutional Neural Network are well-adapted to Computer Vision tasks. After being trained they can detect visual concepts with very high precision [**?**,**?**]. Recent works have shown that these networks could be combined to match visual content with text content. For instance neural networks such as NeuralTalk [**?**], DenseCap [**?**], Show and Tell [**?**] or more recent systems [**?**] can produce sentences to describe visual content. These systems produce vector representations for images thanks to Convolutional Neural Networks. These representations are then input in a Recurrent Neural Network that generates a sentence describing the image.

Some combinations of techniques make it possible to perform zero-shot video search with simple text descriptions [**?**,**?**]. In particular, it has been shown that good results could be obtained in zero-shot video search by using images taken from Google Images [**?**].

We will propose different zero-shot video retrieval systems in the following and analyse their performances according to their hyperparameters and according to the queries that are given as inputs.

3 Description of the Runs

3.1 Generic Architecture

Fig. **??** illustrates the generic architecture that we have put in place. The green modules represent text-based information, the blue modules contain visual information, the yellow modules represent similarity computations. We tried various combinations to define the four runs that we submitted to the final evaluation.

All our runs are part of the "Fully Automatic" category, since no manual processing was done at any stage, and with training type "D", as we are using tools which were trained on data external to TRECVID.

3.2 Runs Using the First Approach

For each of the topics, we performed a search using the Google Images search engine, and retained the first 100 pictures of the ranked list. To each image, we applied the VGG Deep network, and kept either the last, the penultimate or the antepenultimate layer as feature vector of dimension 4K. Thus we obtained vectors for each of the 100 pictures. We tried to normalize them using

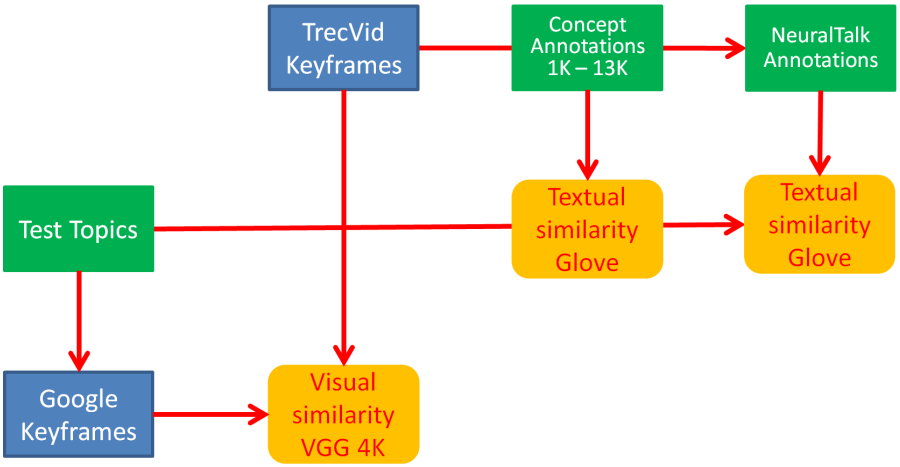

Fig. 1. Definition of our runs

L2-normalization and not to normalize them. We applied the same visual processing to each of the TRECVID keyframes in the test collection, and ranked them according to a Nearest Neighbor distance from Google images.

3.3 Runs Using the Second Approach

We implemented two types of systems based on the second approach. The first one uses 13,000 ImageShuffle concepts. The second one is based on NeuralTalk. In both cases we make a comparison between vectors. We tried to normalize them using L2-normalization and not to normalize them.

With ImageShuffle Concepts We used the ImageShuffle system to obtain scores for 13,000 concepts, which we used as feature vectors for each TRECVID keyframe. We used these scores as weights to compute a semantic vector of dimension 50 (resp. 100, 200 or 300) by a linear combination of the GloVe vectors corresponding to the concepts. For each topic, we constructed a semantic vector of dimension 50 (resp. 100, 200 or 300) by averaging the GloVe vectors of the words appearing in the topic. Then we used the cosine similarity to find the images whose semantic vectors were most similar to the topics.

With NeuralTalk We used the NeuralTalk system to generate text descriptions for each of the TRECVID keyframes. Then, we built a semantic vector of dimension 50 (resp. 100, 200 or 300) by averaging the GloVe vectors of dimension 50 (resp. 100, 200 or 300) of the words appearing in these descriptions. We did the same for the test topics. Finally, we used again the cosine similarity to find the images whose semantic vectors were most similar to the topics.

3.4 Runs Combining Both Approaches

During the development phase, we experimented with a number of combinations of the modules that we have described, using different dimensions, different projections, different layers, different similarity measures. We tried several combinations of our previous approaches and computed a score for each image by averaging its inverse ranks in all results lists.

4 Evaluations

We evaluated all our models. Their results are summed up in Table **??**.

Table 1. Results of our different strategies

Type	Best MAP	Average MAP
Strategy 1	0.0173	0.0098
Strategy 2 (13,000 ImageShuffle)	0.0285	0.0219
Strategy 2 (NeuralTalk)	0.0021	0.0016
Mix of both strategies	0.0167	0.0113

The best strategy seems to be the second one, with ImageShuffle concepts. But we also found out that the efficiency of our strategies depended on the topic. The graph in Fig.**??** is a PCA of the different runs: we built a vector for each run, whose coordinates are the average precisions corresponding to the different topics.

As one can notice, the two strategies seem to have orthogonal behaviors, and the mix of both strategies seems to be a "middle ground". Therefore, we argue that trying to find the best strategy for a given topic instead of mixing strategies would be worthwhile. We will elaborate on that later on.

5 Analysis of the Results

5.1 Effect of L_2-Normalization

As said in the introduction all our runs were based on vectors, and we wondered if it was worth normalizing them. Therefore we made tests with raw vectors and with L2-normalized vectors. In Fig. **??**, each point corresponds to a model. Its abscissa corresponds to its Mean Average Precision (MAP) without L2-normalization and its ordinate corresponds to its MAP with L2-normalization. The red line is the line of equation y = x.

As one can notice, L2-normalization often improves our results, and never deteriorates them.

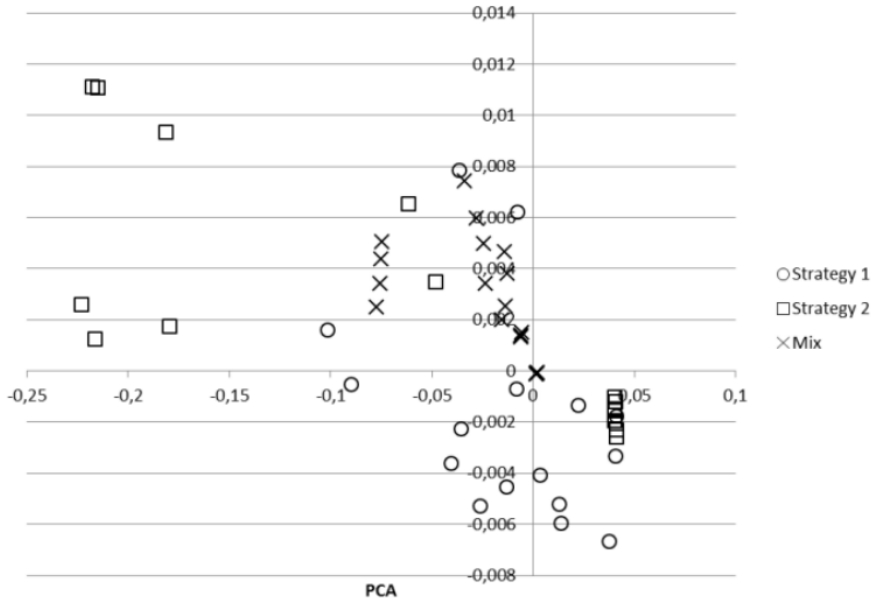

Fig. 2. Behavior of our strategies (X-Axis: min = -0.223, max = 0.041, σ = 0.065; Y-Axis: min = -0.007, max = 0.011, σ = 0.003)

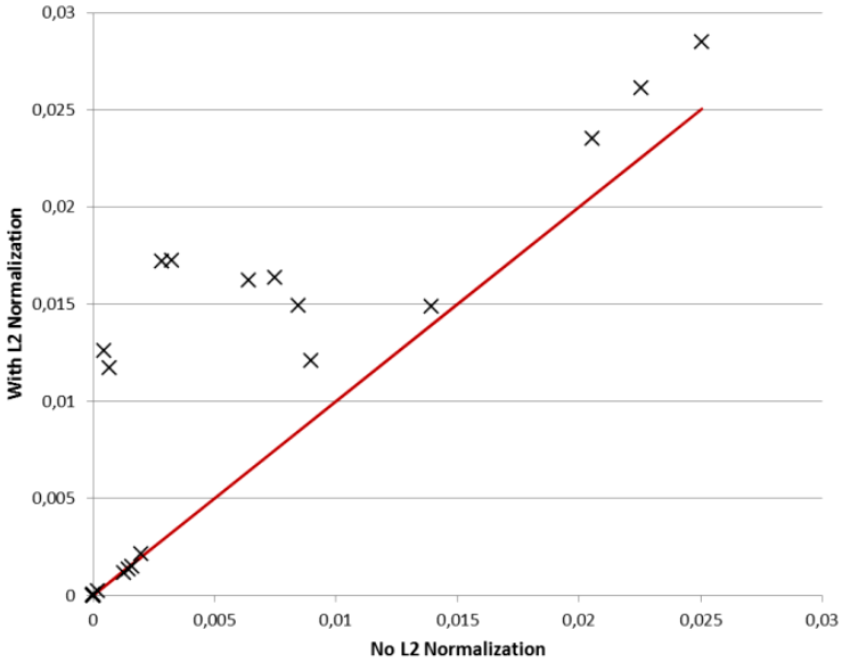

Fig. 3. Influence of L_2-normalization

5.2 Dimension of GloVe Vectors

Our models based on Strategy 2 need a word embedding. We used GloVe vectors, and tried different dimensions (50, 100, 200 and 300). In Fig. **??** we give the MAP for the best five models based on Strategy 2, according to the dimension of the GloVe vectors we chose.

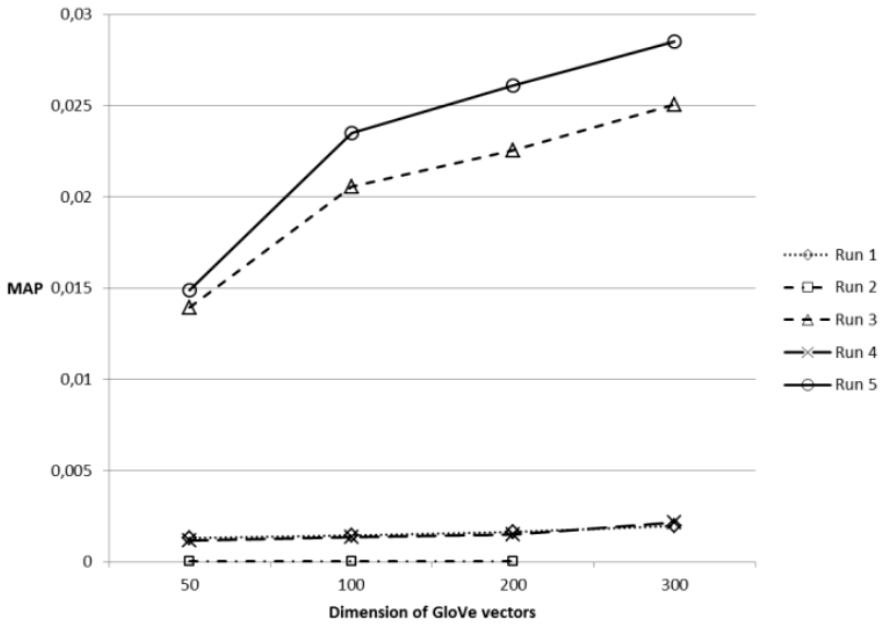

Fig. 4. Influence of GloVe vectors dimensions

As expected the higher the dimension is, the higher the MAP.

5.3 How to Choose the Right Approach?

As we said in part III., Strategy 1 and Strategy 2 are orthogonal in that they do not have good results on the same topics. It would be an interesting challenge to find a way to decide whether we should use the first or the second strategy. More precisely our goal is to choose the best combination of two models from the ones we implemented and use a binary classifier to use the most adapted to a given topic. As expected, we found out that the best combination for the 30 topics of TRECVID was composed of one model based on Strategy 1 and one model based on Strategy 2:

– the first one (Model 1) uses the penultimate layer of the VGG Deep Network, with L2 normalization, to compute the vectors for Google images;
– the second one (Model 2) uses GloVe vectors of dimension 300 to represent ImageShuffle concepts (with L2 normalization).

The results we obtained are presented in Table **??**.

Table 2. MAP of best model and of best combination of two models

Model	MAP
Best model among all	0.0285
Best combination of two models	0.0371

As one can notice the MAP increases by 30% if we choose the best combination of two models, and then apply the most relevant one for a given topic: it would be a significant improvement.

We tried to check if such a system was feasible. For each group of 29 topics, we found the best combination of two models. Then for each topic, we averaged the GloVe vectors of their words, thus getting an overall vector. Next we trained a linear SVM to classify topics according to which model was the most adapted. We eventually applied the SVM on the remaining topic.

Unfortunately we did not get good results, as the MAP of the resulting system was only 0.0084. We think that there are two main reasons explaining these bad results:

– as we only had 30 topics it was difficult to build a general model that would generalize to new topics;
– averaging GloVe vectors to obtain topic vectors may not be adapted, as one can see on Fig. **??**, representing a PCA of the topic vectors (circles correspond to topics that should be processed by Model 1 and crosses correspond to topic that should be processed by Model 2).

6 Conclusion

In this paper we evaluated different models based on two main strategies, aiming at doing ad-hoc video search. The result of any of these models is a vector space that we use to compare queries and keyframes. We studied the importance of two factors: the suitability of L2-normalization and the dimension of word embeddings if they are needed. We also showed that our two strategies were orthogonal: they do not give good results on the same topics. Therefore we proposed a way to decide between these two strategies.

We will need more data, and an efficient embedding for topics to make a suitable implementation of it: our future work will focus on these issues.

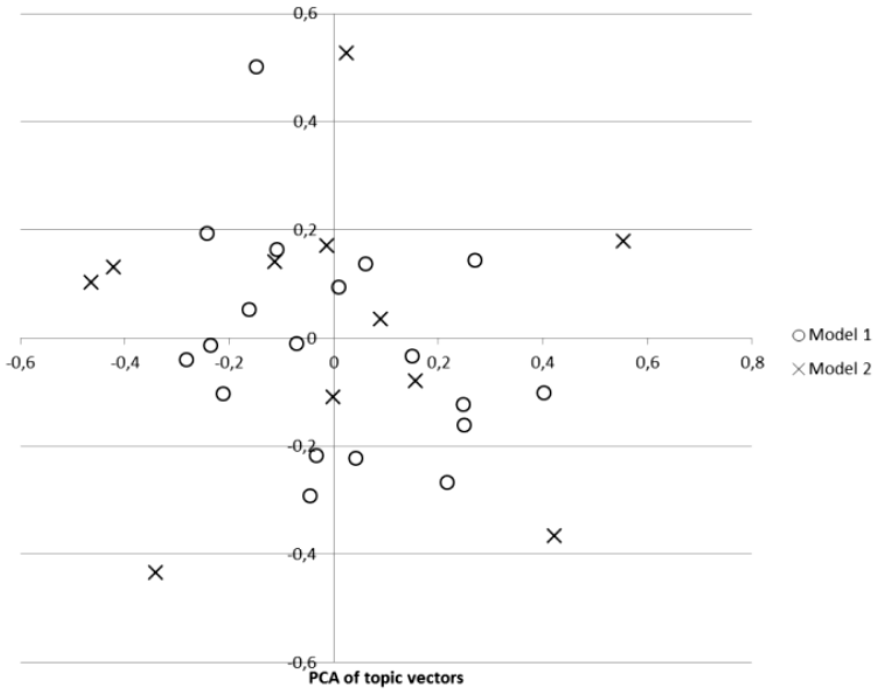

Fig. 5. Topics cannot be easily separated into two groups (X-Axis: min = -0.465, max = 0.555, $\sigma = 0.244$; Y-Axis: min = -0.434, max = 0.526, $\sigma = 0.217$)

References

1. Awad, G., Fiscus, J., Michel, M., Joy, D., Kraaij, W., Smeaton, A. F., Quénot G., Eskevich M., Aly R., & Ordelman, R. (2016, November). Trecvid 2016: Evaluating video search, video event detection, localization, and hyperlinking. In Proceedings of TRECVID (Vol. 2016).
2. Ayache, S., & Quénot, G. (2008, March). Video corpus annotation using active learning. In European Conference on Information Retrieval (pp. 187-198). Springer Berlin Heidelberg.
3. Blacoe, W., & Lapata, M. (2012, July). A comparison of vector-based representations for semantic composition. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (pp. 546-556). Association for Computational Linguistics.
4. Dalton, J., Allan, J., & Mirajkar, P. (2013, October). Zero-shot video retrieval using content and concepts. In Proceedings of the 22nd ACM international conference on Information & Knowledge Management (pp. 1857-1860). ACM.
5. Francis, D., Pidou, P., Merialdo, B., Huet, B. (2017, April). Natural Language Access to Video Databases. BIGMM 2017, 3rd International Conference on Multimedia Big Data. `http://www.eurecom.fr/publication/5199`
6. Habibian, A., Mensink, T., & Snoek, C. G. (2014, April). Composite concept discovery for zero-shot video event detection. In Proceedings of International Conference on Multimedia Retrieval (p. 17). ACM.
7. Han, X., Singh, B., Morariu, V., & Davis, L. S. (2017). VRFP: On-the-fly video retrieval using web images and fast fisher vector products. IEEE Transactions on Multimedia.
8. Johnson, J., Karpathy, A., & Fei-Fei, L. (2016). Densecap: Fully convolutional localization networks for dense captioning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 4565-4574).
9. Karpathy, A., & Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3128-3137).
10. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).
11. Mettes, P., Koelma, D. C., & Snoek, C. G. (2016, June). The imagenet shuffle: Reorganized pre-training for video event detection. In Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval (pp. 175-182). ACM.
12. Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., & Khudanpur, S. (2010, September). Recurrent neural network based language model. In Interspeech (Vol. 2, p. 3).
13. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).
14. Niaz, U., Merialdo, B., & Tanase, C. (2014, February). EURECOM at TrecVid 2014: The semantic indexing task. TRECVID.
15. Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global Vectors for Word Representation. In EMNLP (Vol. 14, pp. 1532-1543).
16. Safadi, B., Sahuguet, M., & Huet, B. (2014, April). When textual and visual information join forces for multimedia retrieval. In Proceedings of International Conference on Multimedia Retrieval (p. 265). ACM.

17. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
18. Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Advances in neural information processing systems (pp. 3104-3112).
19. Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3156-3164).
20. Wang, C., Yang, H., Bartz, C., & Meinel, C. (2016, October). Image captioning with deep bidirectional LSTMs. In Proceedings of the 2016 ACM on Multimedia Conference (pp. 988-997). ACM.
21. Wieting, J., Bansal, M., Gimpel, K., & Livescu, K. (2015). Towards universal paraphrastic sentence embeddings. arXiv preprint arXiv:1511.08198.
22. Google Images Search Engine, `https://www.google.fr/imghp?`