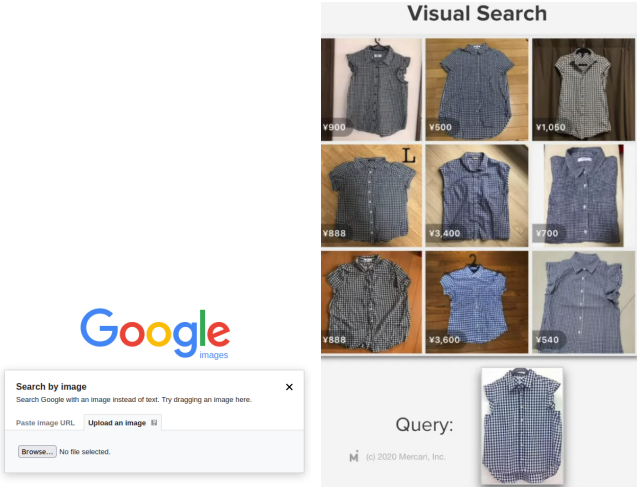
Image retrieval

Marcos Conde, Alex Martin, José Manuel López
UPC - UAB. M5-Visual Recognition, Group 6
{ marcos.conde , alex.martin }@uab.cat

Abstract

Image Retrieval and Recognition algorithms are behind many of the most popular search engines and e-commerce apps. This task is also known as Metric Learning, it aims at automatically constructing task-specific distance metrics from (weakly) supervised data, in a learnable manner. This method is used in the task of image retrieval with feature embeddings extracted from neural networks, as well as for image to text and text to image retrieval with embeddings extracted for images with VGG, FasterRCNN and for text using FastText and Bert. Once the metric learning models learn how to represent the embedding in the space to be close to similar samples and far away from different ones, a clustering method was applied to perform the retrieval, such as KNN and FAIS.



Google Image Retrieval E-commerce Image Retrieval

Figure 1. Modern Image Retrieval systems.

1. Introduction

The Image Retrieval problem consists on finding the most similar images (in a large database) to a given query image [2, 14]. The top-k closest images in the database, according to some sort of distance, are retrieved. Searching for the most similar images is based only on images' features (*e.g.* edges, shapes, illumination, colors, contours, contrast). One common procedure is to extract the features [4, 13] for each database image, then measure the similarity between the input image features and the database images to extract the most similar one. This is known as a bag-of-visual-words architecture, which has proven successful in achieving high precision at low recall.

Deep learning approaches have taking over the retrieval problem, by providing powerful ways of learning features (also known as descriptors) such that the distances between similar images is minimized, and the distances between opposite (or random) images is maximized [4]. These approaches use Convolutional Neural Networks (CNNs) as learnable feature extractors [4] and custom loss functions [7]. We show modern Image Retrieval pipelines in Figures 1 and 2.

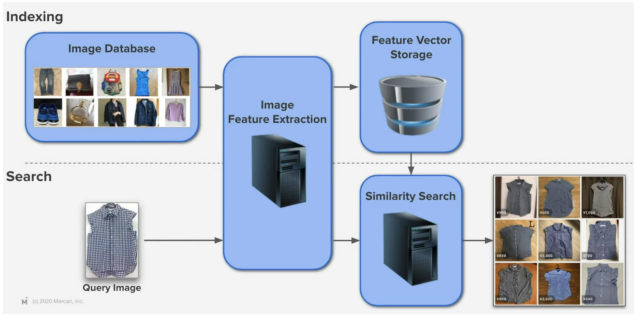


Figure 2. Image Retrieval Pipeline from Mercari e-commerce.

2. Related work

Traditional bag-of-visual-words (BoW) architectures use keypoints and handcrafted features as image descriptors: The Histogram of Oriented Gradients (HOG) [3] is a feature descriptor for objects and edges detection. Scale-Invariant Feature Transform (SIFT) [11] descriptors, SUF and Akaze methods for robust keypoint detection [1, 11, 16] among others.

Deep Metric Learning approaches use CNNs as learnable feature extractors to produce a global and compact fixed-length representation for each image. This, however, is mostly implicitly learned as part of a (weakly) supervised classification task. The Triplet Network model [7] aims to learn useful representations by distance comparisons. We show am example of this architecture in Figure 4. These approaches built on Siamese CNNs [9], this structure implies feeding multiple inputs (images) to the same feed-forward CNN network (some authors refer to this concept as "weight sharing"). In this context we can define: (i) anchor x the base image, (ii)positive x^+ image, an image similar or close to x, (iii) negative x^- image, an image not similar or related (far from) x. The CNN produces $N-$dimensional descriptors $\mathcal{D}$ (typically N=2048) for these 3 inputs. We aim at minimizing the l_2 distance between $\mathcal{D}_x$ and $\mathcal{D}_{x^+}$ while maximizing the distance between $\mathcal{D}_x$ and $\mathcal{D}_{x^-}$.

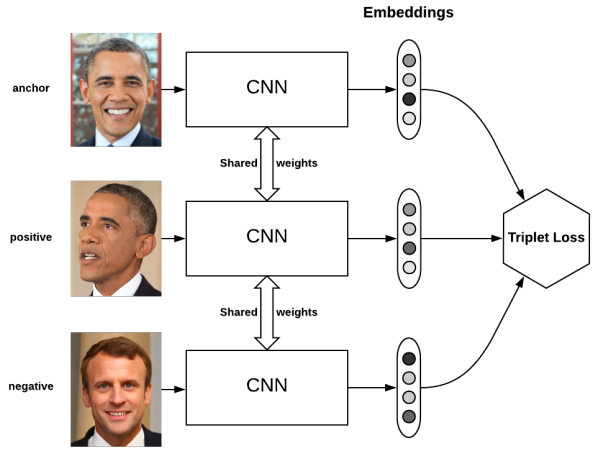


Figure 3. Triplet Loss Architecture.

Various representation techniques for text have been introduced over the course of time. In recent years, none of these representations have been as popular as the word embeddings, such as Word2Vec [12] and GLoVE [15], that took contextual usage of words into consideration. This has led to very robust word and text representations.

Text embedding has been a more challenging problem over word embeddings due to the variance of phrases, sentences, and text. Le and Mikolov [10] developed a method to generate the embeddings that outperforms the traditional bag-of-words approach [5]. More recently, deeper neural architectures have been developed to generate these embeddings and to perform text classification tasks [8] and some of these architectures involve sequential information of text, such as LSTMs, BERT, and XLNET. Furthermore, recently developed attention models can also provide insights about word importance, however they require large amounts of training data.

Method	Pre-trained	Siamese	Triplet
KNN	0.77	0.17	0.58
FAIS	0.78	0.18	0.6

Table 1. MAP Retrieval Results using different descriptors and search algorithms.

3. Experiments

3.1. Image retrieval

We explore the following approaches for image retrieval:

1. **Pre-trained CNN.** We use ResNet50 [6] pre-trained on the MIT split for 5 epochs with cross entropy loss and SGD as the optimizer. We then removed the last layer of the fully connected layer to obtain an embedding of the images and performed KNN and FAIS to these $2048-$dimensional vectors in order to obtain the retrieval.

2. **Siamese Network** [9] for embedding extraction, as it is optimized to create different between images that are in different classes. We use a very simple architecture as backbone with the following architecture: 2 Convolutional layers with $5x5$ kernels of size 32 and 64 respectively, each one followed by a ReLu activation; next 3 linear layers of sizes 256,256 and 10, each one followed by ReLu activations. This simple network was trained for metric learning for 20 epochs. From this network, we then obtained a $10-$dimensional vector with 10 values. We use the contrastive loss with margin 1, Adam optimizer and a learning rate scheduler that reduces the learning rate by a factor of 10 every 8 epochs with an initial learning rate of 0.001.

3. **Triplet Loss** [7] We use the same architecture as above, but in this case we use the Triplet network to perform the training. The details about the inputs are explained in Sec. 2. The optimizer and the learning rate scheduler used in this task were the same as in the previous one. This model was also trained for 20 epochs and provides a $10-$dimensional vector.

We show in Table 1 the retrieval results on the MIT Test Split using descriptors from each of the proposed methods. As we can see, a pre-trained network trained for classification provides a powerful global descriptor of the image. However, a simple $10-$dimensional descriptor from a Triplet Network achieves good results. The dimension of the descriptor is key to perform better, typically this dimension is 1024 or 2048 [2, 4].

3.2. Image to text and text to image retrieval

In the same way that we used the Triplet network for the image retrieval, we used it to perform the metric learning

Figure 4. Example of Image Retrieval using a pre-trained model to obtain the image descriptors. Top left is the query image.

for the embedded features for images and text in both tasks. The embeddings used for text were done using FastText and BERT while the image embeddings were from the last layer of the classification head of the FasterRCNN and VGG. In this case, the embedding model used was different for the text than for the image, using 2 layers of Linear with Relu in the first case and 3 in the second one. The dataset to perform the experiments was Flickr30k images which haves, 31014 images with 5 text captions associated to each image. The training, validation and test split was the one given by Karpathy et al. [8].

As the Triplet network returns an embedding, we tested how it affected the number of features used in the embedding to the accuracy of KNN to retrieve correctly images given text or text given images. The text embeddings had a vector for each word in the sentence, and to use the whole sentence we had to aggregate the different words. The two methods used to aggregate the words of a sentence were adding up the vectors or performing the mean of them.

The trainings were done for 25 epochs, using Adam as optimizer and triplet loss. The learning rate was of 0.01 and a batch size of 1024.

3.3. Image to text and text to image retrieval

3.3.1 Image to text

In this case, an image was used as an anchor for the training and the positive and negative embeddings were text. So, the evaluation of the model was done by giving an image to the model and retrieving the caption to that image. The best results in this case were given by using the text aggregation method of mean, but as you can see the accuracy of the model is in all the cases around the 10 per cent, as it can be seen in 2. It can be seen a little variance in the accuracy of the model depending on the embedding size, the output size of the triplet network.

Embedding size	Train Loss	Validation Loss	Accuracy
256	0.146	0.163	0.103
512	0.145	0.172	0.099
1024	0.138	0.169	0.087
2048	0.137	0.158	0.102
4096	0.152	0.206	0.093

Table 2. Loss and accuracy for image to text retrieval with VGG and Fast Text embedding and mean as text aggregation method

Embedding size	Train Loss	Validation Loss	Accuracy
256	0.275	0.309	0.344
512	0.319	0.367	0.358
1024	0.489	0.567	0.344
2048	0.558	0.771	0.368
4096	0.832	1.704	0.334

Table 3. Loss and accuracy for image to text retrieval with VGG and Fast Text embedding and sum as text aggregation method

3.3.2 Text to image

This time, the evaluation was meant to retrieve images from a caption. In this case, the model performed overall better, and we could test two different embeddings for the images, the ones from VGG and the ones from the classification head of Faster R-CNN.

In 3 we can see that the accuracy was much higher than for the previous task using the same embeddings FastText and VGG. The accuracy moved around the 35 per cent and the difference between the embedding size with most and least accuracy is of more than a 2 per cent. In this task, aggregating the text embedding by using the mean of the words in the sentence led to better results. In 4 we can see the results when instead of using the image embeddings given by VGG, we used the image embeddings given by the classification head of the FasterRCNN. The accuracy of the method used increased by almost a 10 per cent, and the losses of the training of the triplet net can be seen to be lower than with the VGG embeddings. In this case, the output size from the triplet network didn't reflect a lot of variance in the performance.

Looking at the qualitative results, we observed that in many cases the errors were understandable as the captions used to perform the retrieval had only on correct ground truth image but in many cases there would be some other

Embedding size	Train Loss	Validation Loss	Accuracy
256	0.162	0.210	0.422
512	0.160	0.184	0.427
1024	0.167	0.170	0.420
2048	0.186	0.252	0.419
4096	0.222	0.300	0.412

Table 4. Loss and accuracy for image to text retrieval with Faster-RCNN and Fast Text embedding and sum as text aggregation method

images in the dataset that could also be considered as 'correct' as the captions would coincide with the images. In **??** we can see one of those cases.

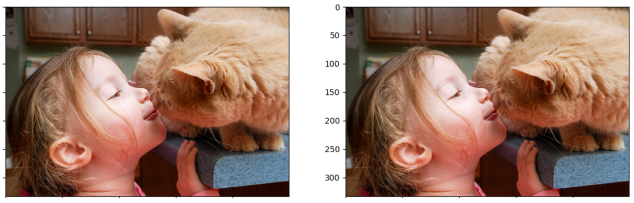


Figure 5. Example of correct text to image retrieval left ground truth, right retrieval by the model

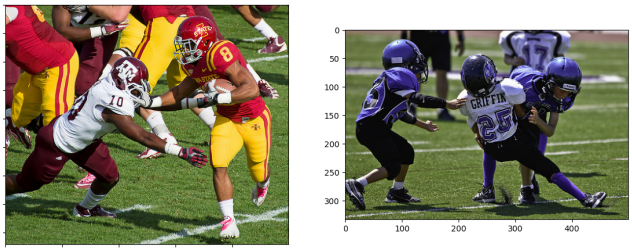


Figure 6. Example of incorrect text to image retrieval at left, ground truth, at right, retrieval by the model

4. Conclusions

As we've seen, metric learning is a powerful method to perform retrieval, although we couldn't the expected good results as we were using a setup that underperformed to see how much it affected the model in the image retrieval. In the case of the cross modal retrieval we can see that the model has its limitations what given the conditions of the datasets the model proved to work. Regarding how much it affects the embedding size of the Triplet network in this task, we can say that it has a minor impact in the performance, although it can be taken into account when fine-tuning this approach. The text aggregation methods used, adding or performing the mean, showed that any of the methods can be suitable depending on the task.

References

[1] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, pages 404–417, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. 1

[2] Ondřej Chum, Andrej Mikulík, Michal Perdoch, and Jiří Matas. Total recall ii: Query expansion revisited. In *CVPR 2011*, pages 889–896, 2011. 1, 2

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005. 1

[4] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus. Deep image retrieval: Learning global representations for image search, 2016. 1, 2

[5] Zellig S Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954. 2

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 2

[7] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*, pages 84–92. Springer, 2015. 1, 2

[8] Andrej Karpathy and Fei Li. Deep visual-semantic alignments for generating image descriptions. pages 3128–3137, 06 2015. 3

[9] Gregory R. Koch. Siamese neural networks for one-shot image recognition. 2015. 2

[10] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR, 2014. 2

[11] D.G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, 1999. 1

[12] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. 2

[13] Anastasiya Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins: Local descriptor learning loss, 2018. 1

[14] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-scale image retrieval with attentive deep local features, 2018. 1

[15] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014. 2

[16] Shaharyar Ahmed Khan Tareen and Zahra Saleem. A comparative analysis of sift, surf, kaze, akaze, orb, and brisk. In *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pages 1–10, 2018. 1