A Project Report

On

IMAGE SEARCH ENGINES

BY

ROHAN RUSSEL NEDUNGADI

17XJ1A0544

Under the supervision of

DR. RAGHU KISHORE NEELISETTI

SUBMITTED IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS OF

PR 402 / PR XXX: PROJECT TYPE COURSE

Mahindra™ University
Global Thinkers. Engaged Leaders.
ÉCOLE CENTRALE
SCHOOL OF ENGINEERING

ECOLE CENTRALE COLLEGE OF ENGINEERING

MAHINDRA UNIVERSITY

HYDERABAD

(Dec 2020)

ACKNOWLEDGMENTS

I would like to thank my faculty incharge Dr Raghu Kishore for being an excellent guide for me , and being the person who I could share and discuss ideas with.

Mahindra University
Global Thinkers. Engaged Leaders.
ÉCOLE CENTRALE
SCHOOL OF ENGINEERING

Ecole Centrale College of Engineering

Mahindra University

Hyderabad

Certificate

This is to certify that the project report entitled "**IMAGE SEARCH ENGINES**" submitted by Mr/Ms. ROHAN RUSSEL NEDUNGADI (HT No. 17XJ1A0544) in partial fulfillment of the requirements of the course PR 402/PR XXX, Project Course, embodies the work done by him/her under my supervision and guidance.

Signature

(DR RAGHU KISHORE NEELISETTI)

Mahindra University, Hyderabad.

Date:

ABSTRACT

With the explosion of data that the web has made available for us, searching content on the internet cannot be limited to text queries. Web content is seeing a sharp rise in images and videos, with the use of Social media platforms. Video and image sharing apps have become very much popular amongst everyone carrying a smartphone these days. Apart from that movie sharing and streaming has also significantly become mainstream in current situations. Hence what if people wanted to search for items they see somewhere or in some movie and want to learn more about it. It would be amazing if we could query the web just like googling any doubt that comes to our mind is fingertips away. Hence this paper will discuss and shed light upon the various aspects and challenges faced to create such a search engine. Our goal will be to be able to create a working search application for images. We should be able to see the internal cogs required for such an application and also discuss the advancements taking place to improve each of these sub tasks. Finally we conclude with several promising directions for future research.

CONTENTS

1.Introduction

Everyone is well aware of the working of a general search engine such as Google. Search engines are essentially information retrieval applications, i.e. given an input they retrieve information that is likely to be the closest form of data available on the query. A popular known information retrieval algorithm is TF-iDF for document retrieval along with employment of clustering algorithms with it. While these are the basic building blocks of a search engine we have come a long way since then and have realised the need to create much more efficient and superior search algorithms.

A general search engine such as Google performs document/ information retrieval from the large data bank known as the internet. Relying on simple techniques like before does not apply here. Hence crucial study has undergone to improve this algorithm time and again. Over the past two decades Google has made several advances and updates to its novel search engine.

But it is not only the algorithms that have been improved over time, it is also the interaction and the way Google is able to understand a person's query. Given a query searching the most relevant and correct data is only half the aspect of an advanced search engine. The second and major part is how do we convert a humans query to machine interpretable query for utilising the efficient retrieval algorithms. What is the point of an efficient retrieval algorithm if it is not able to map the users actual query onto the algorithm?

And so many major advancements have been even in the field of human computer interaction. Major studies have undergone in the study of human languages and understanding the true meaning behind them. It is really funny how once we take a look back at our own languages (english for that matter) we tend to realise that it is absolutely difficult to train a machine to learn it. This is because humans are smart enough to be able to associate a different kind of thought with different kinds of words. This notion can be expressed as what is called as sentiment. Another byproduct of this notion is sarcasm. Hence major studies under the banner of NLP (Natural Language Processing) took place.

In the past decade most of the groundbreaking NLP originated in the Deep Learning field. With large amounts of data we were also able to see a rise in compute power, which was able to make Deep Learning a thing of today. With major progress in the NLP field still going on another question arises. Is language the only way humans interact?

Clearly the answer is no, humans are able to communicate huge amounts of data with just visual information. Moreover there are some things that can only be seen to be appreciated to its fullest extent. Putting them into words and description simply doesn't fit. Furthermore at times we might not even know how to put it in words. Hence don't we want our search engine to be able to do so too.

It would be truly amazing if we could have our search engine understand our query by an image. It could be a problem as simple as trying to get information about a flower whose name I do not know but have seen grow in my garden. Traditional search engines would require one to describe the image as much as possible and create a query only to be unsure whether it is the only flower with that description. Moreover people can be really bad with their descriptions, they may describe a Rose as a red flower. There are millions of varieties of flowers that can be red in color.

A simple way could be that I put an image into the search bar and it is able to retrieve information regarding it. This has become very much possible now and major work is going on in this work and area. The potential for this kind of a search engine is very high considering the major advances in the field of image processing over the past decade.

With introduction and extremely good results of Deep learning models image classification and understanding has become very easy today. Machines are able to comprehend and crunch information encapsulated in images. The processing has evolved from fourier to convolutions and have started giving very decent results.

Many search algorithms that worked in the NLP models can be easily visualised and extrapolated for images as well.

Traditionally given a text query the search engine could search tagged images for the incoming query. Our work would involve doing the exact opposite, given an image we must be able to tag it. The official name taken up for processing of images for information is coined as the reverse image search.

There are major advanced uses of such a reverse tagging. Just like in NLP document labelling is one byproduct of the algorithms devised for information retrieval. Similarly , initially where image data was manually labelled on the internet, can now be automatically labelled , this in turn improves all search results in general. A fundamental process involved in information retrieval on the web is tagging or labelling of data based on some sort of characteristics of the said information. Once already comprehended- either manually or using deep learning, we need not have to run expensive operations over and over again, and instead use the comprehended information saved.

Hence this sets the stage for a lot of current and future work - to be able to find the best fitting algorithms and most efficient techniques querying an image in real time and get responses pertaining to accurate information. Future works also involve understanding the sentiment and the part of the image that the user is interested in. A picture may contain a wide variety of information depending on the person trying to comprehend it. Over all this field and area proves to be challenging and exciting.

2.Problem Definition

The problem to solve in this project when put in simple words can be spelled out as- to be able to create an application that is able to take images as inputs, comprehend them and return web related information (cause internet is the biggest pool of information) or private database related information. In a broad sense we want to be able to search information via images.

To do so we need the application to be able to comprehend or understand images. Just as NLP handles machines to understand language, with deep learning frameworks we must be able to understand images. In this paper we shall see the progress and growth of ideas and implementations and how certain algorithms were able to improve the quality of our search engine.

The frameworks and modules in our current state do not allow us to be able to understand an image in a complete sense all at once unless trained carefully. Basically an image can communicate various kinds of information and our goal at a preliminary level is to first be able to extract some kind of information from it. The simplest way to go about doing so is to first get the machine to be able to differentiate between images. This task is known as classification. If given a set of curated images belonging to a finite set of classes(shirt, bag, shoes), are we able to get the application to simply differentiate an image(say of a shoe) from one another(of a shirt) by classifying them to their appropriate different classes.

Moving ahead, by being able to do this for simple images we can now classify images into a broad sense of labels. With these labels (single label per image) we can now perform regular text queries. This uptil now seems a little naive i.e. while it is a great feat it isn't the best that can be done or rather these results don't peek any attention. Being able to classify an image and then query a word from a given set of known words is not what was the motivation behind the need for an image search engine. We want to be able to query images which may or may not even be able to be broadly classified but simply find items similar to it. A simple solution is to be able to to tag an image with multiple labels. (eg mona lisa = painting, lady, antique)

A more refined work that we are looking for is similar to TFiDFs word/content matrix that is used in text based retrieval. We want the application to be able to assign a set of characteristics and the degree of that characteristics to images, so that based on it we can identify other many records(images) that represent similar information. This process is also known as vectorisation, of the information, as these set of features and the degree of belongings to these features will simply form a vector. Now using mathematical concepts of similarity (such as cosine similarity) we can find the similarity between different images. This helps us find data similar to anything without having to classify the query image into a definite class.(while also being able to classify images if needed).

The next question that arises is how are we gonna be able to define these sets of features and how many of them and what is the optimal number of features. All of these questions can be answered by delving deep into studies of autoencoders. Auto encoders in general are essentially unsupervised learning techniques which leverage neural networks for the task of representation learning. Specifically, we'll design a neural network architecture such that we impose a bottleneck in the network which forces a compressed knowledge representation of the original input. The algorithm works on the inherent nature of images, that the inputs(pixels) to make meaningful images have some sort of correlation and dependency between each other. This idea of dependency is also seen in GANs where the image is considered to be a probability distribution and the difference between images is nothing but the Kullback-Leibler divergence used on these distribution plots. We will also do a preliminary study to find the best amongst various techniques(such as cosine similarity, kl divergence or even clustering algorithms) to find similarities in images.

Auto encoders work by giving the input data and making them go through a set of compression layers(layers where the number of nodes are less than the previous layers) followed by a set of decompression layers(layers where the number of nodes are more than the previous nodes) until eventually the final layers has the same number of nodes as the input. The loss is calculated on these outputs as to how dissimilar they are to the original image.Special types of autoencoders also work as filters on corrupted images to remove noise.

The most popular autoencoder one must have come across is the Word2vec model in NLP. The common term used for these compressed encodings are embeddings. These embeddings can be visualised as characteristics of the image that the model was able to classify. The characteristics chosen by the model are not exactly the same as how humans would do it and some are not even easily understandable by humans. It is also important to know that these are not the only and best ways to create images embeddings and many studies in this paper will be done on advanced and adapted algorithms that originated from auto encoders that did significantly better.

Further we must also be able to answer many critical questions that arise. While image embeddings are characteristics to a particular object in the image, what about the case when the images have unremovable noise? Furthermore how do we classify an image with multiple objects in it? This paper will go through and discuss some novel algorithms and techniques that can be employed (yolo, sift, surf, fast).

Another aspect that can be viewed when seen from the perspective of an ecommerce market. When searching for images of say a tshirt, we can visualise this problem as a 2 step classification problem. First is to identify that it's a tshirt, and the second is the kind of the shirt - full sleeve, half sleeve, round neck, v neck etc. In the case of furniture we can think of this problem as the particular shape and design of the object. Hence we shall also look into designing the efficiency of embeddings and if we need for two sets of embeddings to represent a global and local embed.

Lastly but not the least we must delve into being able to calculate the accuracy of mapping from the thought behind the image that sparked a query in the user to an appropriate embedding. Does this mean an image can have multiple embeddings based on context? And if yes how do we enable the incorporation of this added context? Are studies that we will ponder.

3.Background and Related work

Image Classification:

As seen earlier, major work that bred valuable results in understanding images started only in the last decade where the resources and compute power to perform such herculean tasks had become feasible.

In 1998 LeCun et al. introduced CNN to classify handwritten digits. Their CNN model, called LeNet-5 [1] had 7 weighted (trainable) layers. This was the first time someone was able to create a model that could classify numerical digits from the mnist dataset with sufficiently good results.

This work had only minor betterments for a long time until the Alex net came out in 2012. In 2012 Krizhevky et al. designed a large deep CNN, called AlexNet [2], to classify ImageNet [3] data. The architecture of AlexNet is the same as LeNet-5 but much bigger. It is made up of 8 trainable layers

Simonyan and Zisserman used deeper configuration of AlexNet [2], and they proposed it as VGGNet [4]. They have used small filters of size 3 × 3 for all layers and made the network deeper keeping other parameters fixed. They have used a total of 6 different CNN configurations.

The architecture of GoogLeNet [5], proposed by Szegedy et al., is different from conventional CNN. They have increased the number of units in each layer using parallel filters called inception modules [32] of size 1 × 1, 3 × 3 and 5 × 5 in each convolution layer (conv layer).

He et al. experienced that a deeper CNN stacked up with more layers suffers from vanishing gradient problems. They have proposed a deep residual learning framework [6] as a solution to the degradation problem and called it the Resnet[6].

Huang et al. introduced Dense Convolutional Networks (DenseNet) [7], which includes dense blocks in conventional CNN. The input of a certain layer in a dense block is the concatenation of the output of all the previous layers.

Future works to read involve : CapsNets[8] and SENets[9].

All these papers kept improving on each other's architectures and made understanding of images much better. With the concept of transfer learning we can use large pretrained models for our own problem statements by simple fine tuning them. While the original model may have been trained for a different set of tasks(classification) the earlier levels of the weights are nothing but the models understanding of the image, which we can append with fresh weights to learn specifically for different tasks (embedding/tagging/labeling etc).

Image Embeddings:

The ideology behind the use of image embeddings originated from what were called as autoencoders. [10] shows a brief walk through the complete idea of autoencoders and how they evolved. We can see in [10] how the notion started with being able to create compressed representations of information, which slowly evolved to a higher understanding of this methodology. It also shed light on how it was found that we could use autoencoders with linear activation functions to perform PCA on the input data. Subsequently autoencoders with non linear activation function were considered a more superior form of PCA. The article later goes on to show how multilayer auto encoders were able to be used for multi-level embeds. We further see in [10] how autoencoders were also used to be able to remove noise in images by performing PCA like functions as discussed earlier.

The use of the popular autoencoding technique as shown in [10] was seen in the use of NLP for the model known as Word2Vec[11].

Image differentiation and object detection:

With the ideas seen in [10] and [11] we can now convert images into feature vectors known as image embeddings. Having a vector now makes it permissible to treat an image as a point in vector space and hence find similarities by spatial orientation. The first distance measurement that comes to find and fits our purpose is the cosine similarity. While this works well , after many years of looking into and testing many other methods have been proposed.

As seen in [12] , the use of KNNs and other such techniques which focused on grouping were adopted for various tasks when implementing a search engine. [12] also shows that Efficient Net was the best of models that worked far far better than all described in [1]-[9]. Using the auto encoding technique with this gave astonishing results.

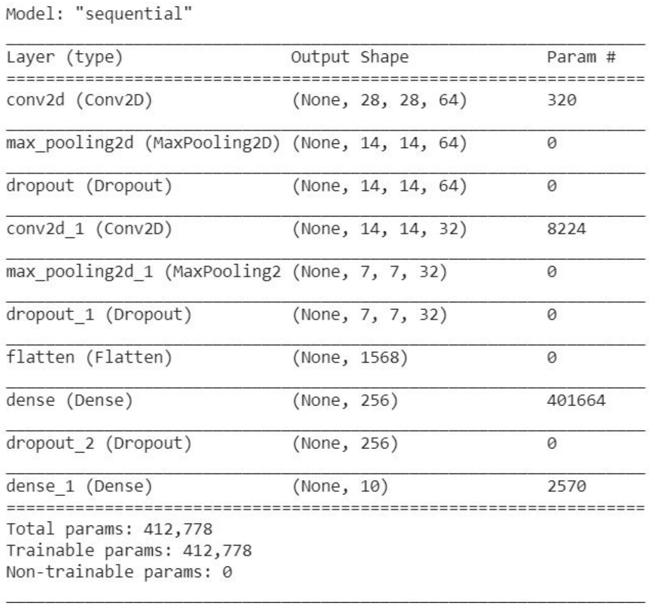
These vector spaces models faced certain issues too. It takes a long time and heavy amount of preprocessing elements to understand embeddings of objects in images than the images itself. What it means is, a slightly augmented image should give minimal shift in vector space. This issue was very well handled in Kulback Leibler divergence, where the divergence score was independent of augmentation and purely worked on the distribution of data.

Eventually much more specifically designed algorithms were introduced. [12] shows us how image descriptor algorithms such as SIFT,SURF, and FAST improved identification and labeling of images. SIFT as explained in [12] was an algorithm that was completely invariant of geometric augmentations, it was solely able to capture parts of images. It used the concepts of octaves and gaussian blurring to understand a single image from various aspects. With the use of DOGs(Difference of Gaussian Kernel) it is able to find similarities in various versions of the same image after blurring. The similarities that stayed intact were known as key points. Further operations are conducted on these said key points to identify direction and position of each of them in the various preprocessed versions of the same image. This way it was able to find key descriptions of the image to identify similarity or detection of the object.

4.Implementation

For this paper we have created a simple prototype to be able to visualise the concept and have an idea of what the end product will look like. For our prototype we have taken up a subset problem of being able to recognise and classify an image of apparel/clothing. The used case being that now he can be recommended clothes on the basis of his uploaded image.

The deep learning models have been created with tensorflow in Python. Simultaneous work is also going on to familiarise ourselves with another popular and highly customisable framework, Pytorch. Dataset employed for this prototype is the fashion-MNIST (28x28x1 sized), which consists 10 classes of clothing to which the images must be classified to. A simple 2 conv layer architecture (as seen in Figure 1)along with ELU activation and max pooling with dropout was employed to give accuracies of about 92% as seen in figure 2



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d (Conv2D)              (None, 28, 28, 64)        320       
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 14, 14, 64)        0         
_________________________________________________________________
dropout (Dropout)            (None, 14, 14, 64)        0         
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 14, 14, 32)        8224      
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 7, 7, 32)          0         
_________________________________________________________________
dropout_1 (Dropout)          (None, 7, 7, 32)          0         
_________________________________________________________________
flatten (Flatten)            (None, 1568)              0         
_________________________________________________________________
dense (Dense)                (None, 256)               401664    
_________________________________________________________________
dropout_2 (Dropout)          (None, 256)               0         
_________________________________________________________________
dense_1 (Dense)              (None, 10)                2570      
=================================================================
Total params: 412,778
Trainable params: 412,778
Non-trainable params: 0
_________________________________________________________________
```

Figure 1

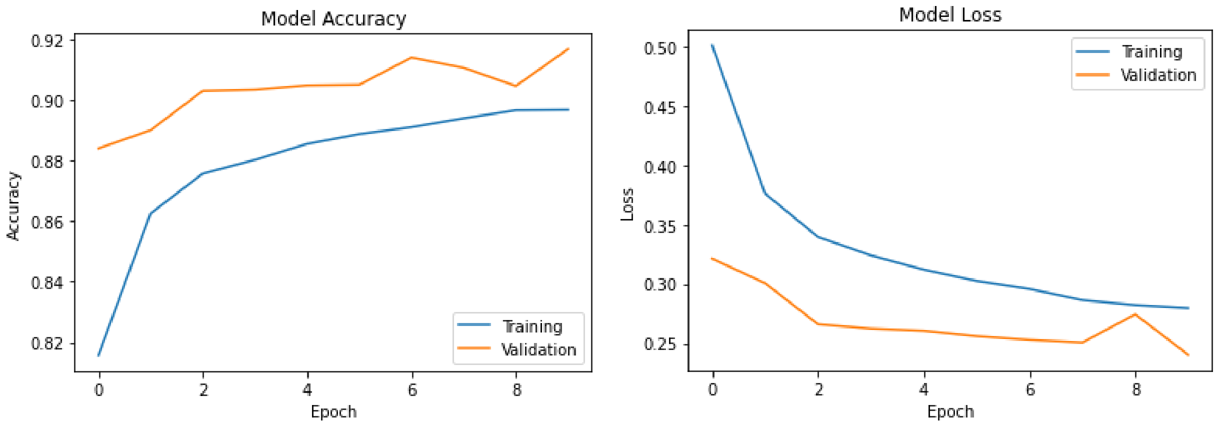


Figure 2

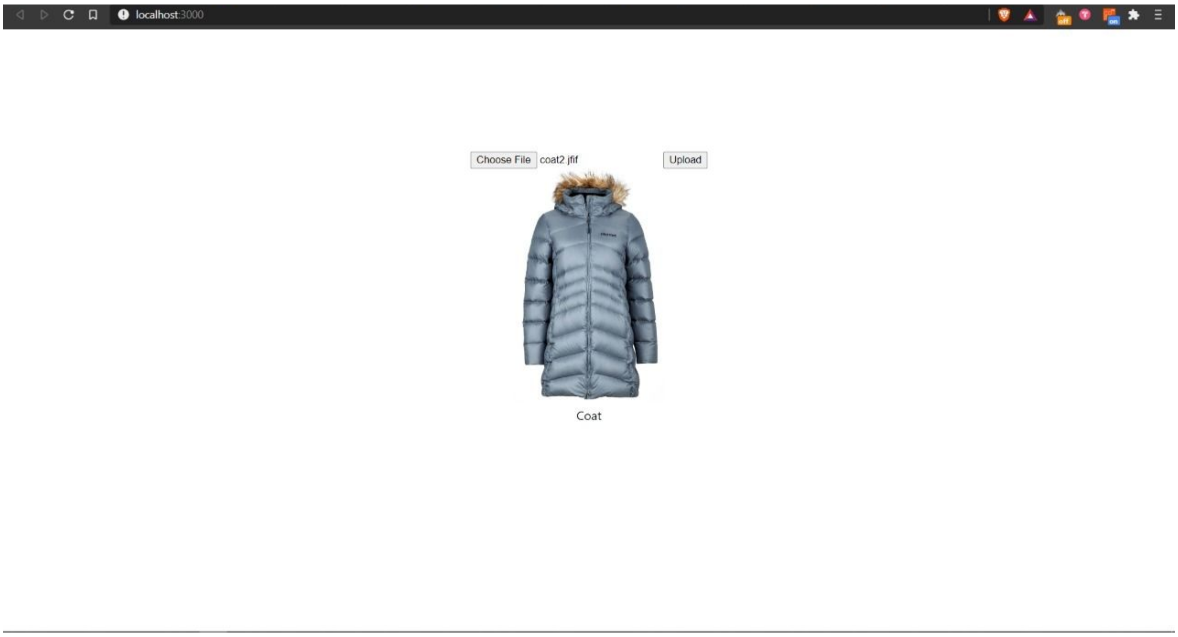


Figure 3

The front end and UI was created with the help of ReactJs. Other platforms such as android(Kotlin) are also being explored. For our current prototype we have implemented a simple web based app where the user can upload the image and get results as seen in figure 2.

The prototype has a flask backend to run the tensorflow model. With the help of an API we send the image across to the backend where necessary preprocessing is done before loading and running the model from a saved state. The classification is then sent back to the requesting front end. Exploration in TF-JS is also being carried forward. With this feature we will be able to save trained Tensorflow models as jsons and incorporate them directly into javascript. This means we can eliminate the need for a backend server and instead use the model in the browser itself.

5.Results

6.Conclusion

7.References

[1] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, Nov 1998

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in Neural Information Processing Systems 25, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. . Available:http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks. pdf

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A Large-Scale Hierarchical Image Database,” in CVPR09, 2009

[4] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” CoRR, vol. abs/1409.1556, 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015

[6] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016

[7] G. Huang, Z. Liu, and K. Q. Weinberger, “Densely connected convolutional networks,” CoRR, vol. abs/1608.06993, 2016. [Online]. Available: http://arxiv.org/abs/1608.06993

[8] S. Sabour, N. Frosst, and G. E. Hinton, “Dynamic routing between capsules,” CoRR, vol. abs/1710.09829, 2017. [Online]. Available: http://arxiv.org/abs/1710.09829

[9] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” CoRR, vol. abs/1709.01507, 2017. [Online]. Available: http://arxiv.org/abs/ 1709.01507

[10] Blog : https://www.jeremyjordan.me/autoencoders/

[12] Blog : https://rom1504.medium.com/image-embeddings-ed1b194d113e

[11] Distributed Representations of Words and Phrases and their Compositionality -Tomas Mikolov Google Inc. Mountain View mikolov@google.com Ilya Sutskever Google Inc. Mountain View ilyasu@google.com Kai Chen Google Inc. Mountain View kai@google.com Greg Corrado Google Inc. Mountain View gcorrado@google.com Jeffrey Dean Google Inc. Mountain View jeff@google.com - https://papers.nips.cc/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf

To Read in future :

[1] J. Sivic and A. Zisserman, “Video Google: A text retrieval approach to object matching in videos,” in IEEE Conference on Computer Vision and Pattern Recognition, 2003, pp. 1470–1477.

[2] D. Nister and H. Stewenius, “Scalable recognition with a vocabulary tree,” in IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, 2006, pp. 2161–2168.

[3] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, “Object retrieval with large vocabularies and fast spatial matching,” in IEEE Conference on Computer Vision and Pattern Recognition, 2007, pp. 1–8.

[3] H. Jegou, M. Douze, and C. Schmid, “Hamming embedding and weak geometric consistency for large scale image search,” in European Conference on Computer Vision, 2008, pp. 304–317.

[4] W. Zhou, H. Li, Y. Lu, and Q. Tian, “Large scale image search with geometric coding,” in ACM International Conference on Multimedia, 2011, pp. 1349–1352.

[5] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman, “Total recall: Automatic query expansion with a generative feature model for object retrieval,” in International Conference on Computer Vision, 2007, pp. 1–8.

[6] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, “Lost in quantization: Improving particular object retrieval in large scale image databases,” in IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.

[7] O. Chum, J. Philbin, and A. Zisserman, “Near duplicate image detection: min-hash and tf-idf weighting,” in British Machine Vision Conference, vol. 3, 2008, p. 4.

[8] Z. Wu, Q. Ke, M. Isard, and J. Sun, “Bundling features for large scale partial-duplicate web image search,” in IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 25–32.

[9] W. Zhou, Y. Lu, H. Li, Y. Song, and Q. Tian, “Spatial coding for large scale partial-duplicate web image search,” in ACM International Conference on Multimedia, 2010, pp. 511–520.

[10] O. Chum, A. Mikulik, M. Perdoch, and J. Matas, “Total recall II: Query expansion revisited,” in IEEE Conference on Computer Vision and Pattern Recognition, 2011, pp. 889–896.