A report on

Drawing based Image Retrieval

Submitted by

Nisarg Vora 2018A7PS0254P

In partial fulfillment of the course

Study Project

(CS F266)

Date: 12th May, 2021

Acknowledgement

I would like to express my appreciation to all those who helped me in making this project reach its completion.

The successful completion of this project could only be possible due to a lot of support and guidance from Prof. Navneet Goyal, Computer Science and Information Systems, BITS Pilani. I am extremely privileged to have obtained this all through the process. I would also like to thank Prof. Navneet Goyal for introducing me to the problem and giving me an opportunity to work on it.

A special thanks to Prof. Sudeept Mohan, Head of Department, Computer Science and Information Systems, BITS Pilani for providing the opportunity to work on this topic.

At last, I express my gratitude to my family and friends for a constant source of motivation that they consistently provided over the course of this project.

Abstract

With the growing usage of pen and touchscreen based devices, drawing based image retrieval becomes an important method that complements the traditional language based image search on the search engines. For this project we use the sketchy database which contains 75,471 drawings of 12,500 objects spanning over 125 different categories created by collecting drawings from a crowd consisting of various age groups. The recent developments of deep neural networks for the image classification tasks motivates us to look at its possible implementations.A model based Siamese networks with triplet loss is implemented for finding similarity between drawings and photos and retrieve relevant images based on it.

Table of Contents

Introduction

With the growing usage of pen and touchscreen based devices drawing based image retrieval becomes an important method that complements the traditional language based image search on the search engines.As a result, the domain has received a lot of attention in the recent years. It is also referred to as Query by visual example. It allows non artist and non professionals to retrieve relevant image results based on their drawings.

The domain has been studied for over 29 years[1]. The users provide a drawing as a query based on which the relevant results are retrieved from the gallery. The major challenge of Drawing based image retrieval lies in the fact that the drawing and images are very distinct in appearance and the drawing made by a human is not a perfect description of the image. However, the distinct drawings are recognizable to humans with their images. Various approaches have been designed that try to understand the gradient and edges that are present in both the images and the drawings. The task therefore requires high-level understanding of the drawing and images for proper retrieval of images.

Recent progress in deep learning and computer vision has strengthened the computerised understanding of both drawings and images, individually. For example Convolutional Neural Networks have enabled identification of images across 1000 different categories in the ImageNet challenge[2]. Face recognition has also been possible due to these techniques. Therefore we work on these techniques to develop a computer vision based model for drawing based image retrieval.

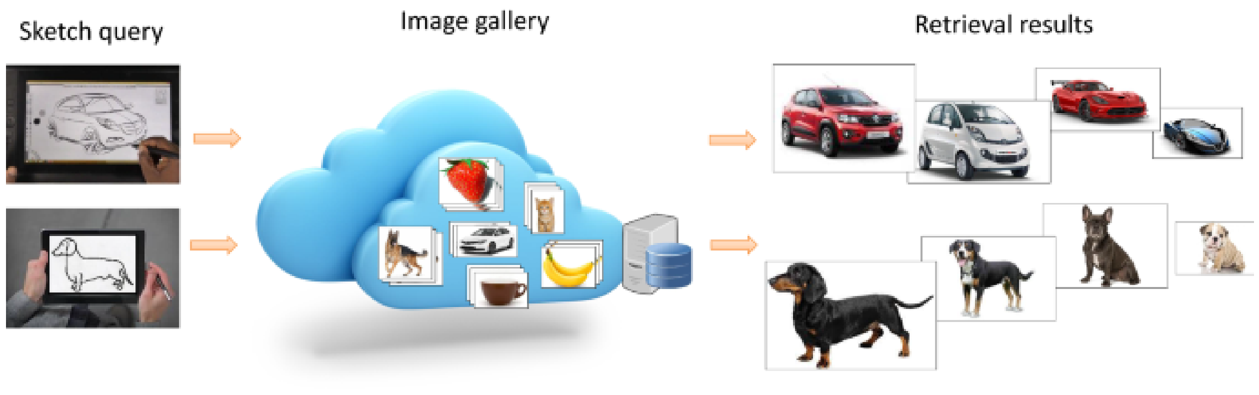


Figure 1. Drawing based Image Retrieval Overview

Literature review

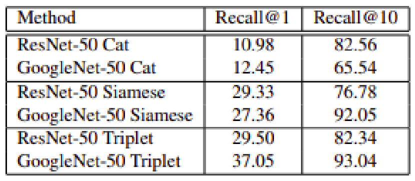
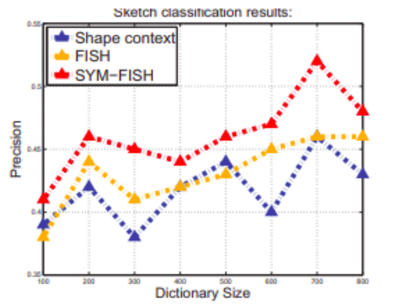
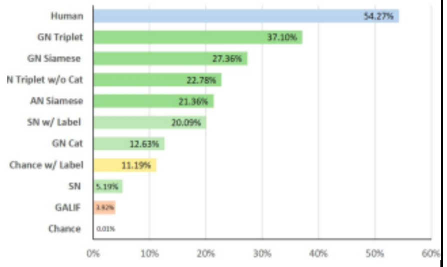
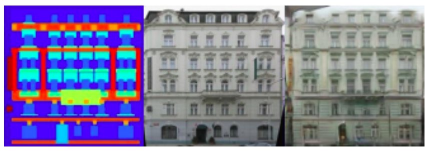
Various object detection techniques have been used for the task of drawing based image retrieval. C. Xiaochun et al. proposed a new shape context descriptor called the SYM FISH[3] descriptor instead of using the standard shape descriptor. The sym-fish descriptor is different from the shape context descriptor primarily due to the fact that it incorporates symmetry considerations as well which considers the pixels.

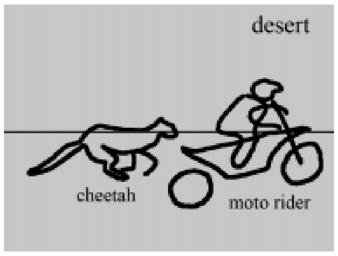
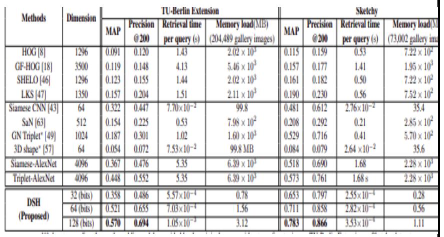
Li Liu et al. proposed a novel binary coding method, named Deep Sketch Hashing [4](DSH), where a semi-heterogeneous deep architecture is proposed and incorporated into an end-to-end binary coding framework. Specifically, three convolutional neural networks are utilized to encode free-hand sketches, natural images and, especially, the auxiliary sketch-tokens which are adopted as bridges to mitigate the sketch-image geometric distortion. The learned DSH codes can effectively capture the crossview similarities as well as the intrinsic semantic correlations between different categories. To the best of our knowledge, DSH is the first hashing work specifically designed for category-level SBIR with an end-to-end deep architecture.

Several researchers have also employed conditional Generative Adversarial Network to generate the images based on sketches instead of retrieving the images. cGAN is a good option for image synthesis from sketches however many times the results are very fuzzy. cGAN is not a good choice when it comes to image retrieval.

The paper on FaceNet[5], published in 2015 forms the primary base for this paper. In the paper, the authors propose a novel method for face recognition using one shot learning. The method was primarily designed to solve the task of face recognition with high accuracy, however it can also be implemented with drawing based image retrieval with proper changes. The key proposal in the paper is of Siamese Networks that by minimising the triplet loss give a mapping of the face in such a way that the euclidean distance between the same faces is minimized and that between different faces is maximized[9]. On the widely used Labeled Faces in the Wild (LFW) dataset, the system achieves a new record accuracy of 99.63%. On YouTube Faces DB it achieves 95.12%. The Siamese network based approach also leads to representational efficiency as large images can only be represented by the feature vector which is much smaller in size than the image itself. Such prior representations also speeds up the retrieval process at runtime.

The table on the next pages provides a brief overview with the links of the prior works.

	Paper Name:	Dataset:	Methodologies used:	Results:	Inference
1.	Sketch Based Image Retrieval [6]	Flickr images of 125 categories	1.ResNet / GoogleNet 2.Siamese Network 3.Triplet Network	 Method — Recall@1 — Recall@10 ResNet-50 Cat — 10.98 — 82.56 GoogleNet-50 Cat — 12.45 — 65.54 ResNet-50 Siamese — 29.33 — 76.78 GoogleNet-50 Siamese — 27.36 — 92.05 ResNet-50 Triplet — 29.50 — 82.34 GoogleNet-50 Triplet — 37.05 — 93.04	GoogleNet architectures inside a Siamese/Triplet network better than ResNet architecture on a Siamese network.
2.	SYM-FISH: A Symmetry-aware Flip Invariant Sketch Histogram Shape Descriptor [3]	ETH dataset.	1.Computing the FISH descriptor 2.Discovering the symmetry character 3.Constructing a symmetry table combined with FISH descriptor.	 re 6. The classification results w.r.t. different dictionary size.	Performs better than shape context descriptor but no comparison with other computer vision techniques available.
3.	The Sketchy Database: Learning to Retrieve Badly Drawn Bunnies [7]	Sketchy Database + 1000 Flickr photos	Varying Convolutional Neural Networks		Triplet Network with Googlenet model performed the best among a given set of Convolutional Neural Networks
4.	Image-to-Image Demo	Random Images	**cGAN** using pix2pix library tensorflow		GAN a good option for image synthesis from sketches however many a times the results are very fuzzy.

5.	[Sketch to photo](#) [8]	Web Search Images	1.Queries based on sketch + item tags. 2.Amalgamation of various techniques	 	Means of intermixing features of different images.
6.	[Deep Sketch Hashing](#) [4]	TU Berlin extension & Sketchy Dataset	1.Convert images to sketches. 2.Use CNN to convert input sketch and sketches of all the images. 3.Use different loss metrics		Huge computations involved. Variation of earlier CNN based versions.

Methods	Dimension	TU-Berlin Extension				Sketchy			
		MAP	Precision @200	Retrieval time per query (s)	Memory load(MB) (204,489 gallery images)	MAP	Precision @200	Retrieval time per query (s)	Memory load(M (73,002 gallery ima
HOG [8]	1296	0.091	0.120	1.43	2.02×10^3	0.115	0.159	0.53	7.22×10^2
GF-HOG [18]	3500	0.119	0.148	4.13	5.46×10^3	0.157	0.177	1.41	1.95×10^3
SHELO [46]	1296	0.123	0.155	1.44	2.02×10^3	0.161	0.182	0.50	7.22×10^2
LKS [47]	1350	0.157	0.204	1.51	2.11×10^3	0.190	0.230	0.56	7.52×10^2
Siamese CNN [43]	64	0.322	0.447	7.70×10^{-2}	99.8	0.481	0.612	2.76×10^{-2}	35.4
SaN [63]	512	0.154	0.225	0.53	7.98×10^2	0.208	0.292	0.21	2.85×10^2
GN Triplet* [49]	1024	0.187	0.301	1.02	1.60×10^3	0.529	0.716	0.41	5.70×10^2
3D shape* [57]	64	0.054	0.072	7.53×10^{-2}	99.8 MB	0.084	0.079	2.64×10^{-2}	35.6
Siamese-AlexNet	4096	0.367	0.476	5.35	6.39×10^3	0.518	0.690	1.68	2.28×10^3
Triplet-AlexNet	4096	0.448	0.552	5.35	6.39×10^3	0.573	0.761	1.68 s	2.28×10^3
DSH (Proposed)	32 (bits)	0.358	0.486	5.57×10^{-4}	0.78	0.653	0.797	2.55×10^{-4}	0.28
	64 (bits)	0.521	0.655	7.03×10^{-4}	1.56	0.711	0.858	2.82×10^{-4}	0.56
	128 (bits)	**0.570**	**0.694**	1.05×10^{-3}	3.12	**0.783**	**0.866**	3.53×10^{-4}	1.11

Table 1 Literature Review Overview

Dataset Description

For this project, we have used the Sketchy Database. It is one of the first and largest drawing-photo collection dataset and is made available by the Georgia Tech University.

The dataset consists of 75,471 drawings of 12,500 objects spanning over 125 different categories. The dataset was developed using a crowd collection technique. Two possible approaches were considered for the crowd collection of the drawings of different images. Prompt the creation of drawings from particular photos or have people associate existing drawings to photos, e.g. by ranking a list of potentially matching photos. The first strategy was employed because it is better able to create finegrained, instance-level associations.

The crowd consisted of people from varying age groups and backgrounds to properly incorporate randomness in the drawings. The participants were first shown a few images and were then asked to make the drawings representing those images based on their memory. This was done in order to mimic the real scenario in which the users make a particular drawing based on some old memory. Thus to achieve this the outline traced drawing of the images were not taken as done by other databases because the drawings based on memory are more prone to errors. The outline traced images are to some extent a more perfect representation of the image requirement.

Our focus for this project primarily lies on the animal drawings in the Sketchy Database as per the problem requirement. Ideas similar to those discussed in this project can be extended for other object images as well.

Sample :

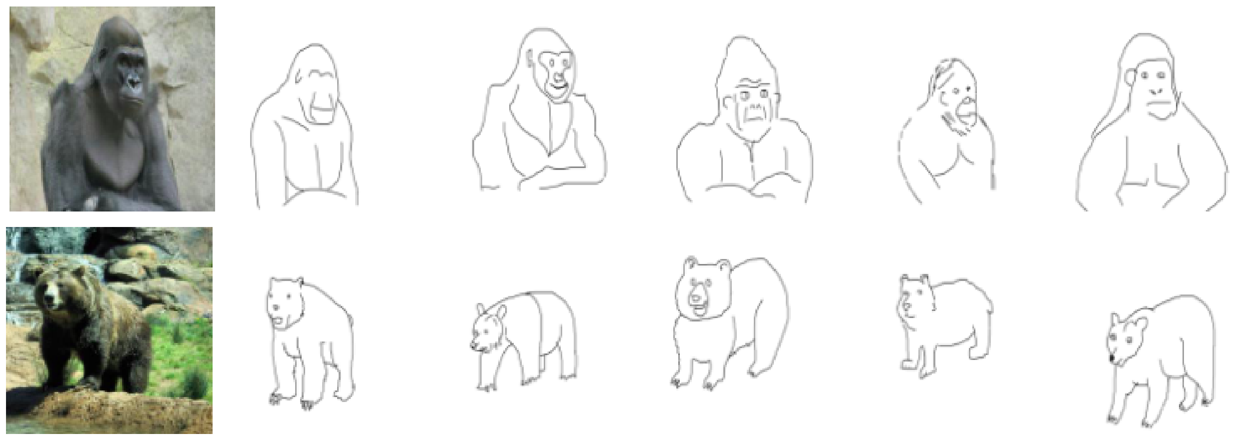


Figure 2. Sample data from sketch database

Metric Learning[10]

In the metric learning approach, the aim is to establish similarity or dissimilarity between objects, images or data points which is measured in terms of a distance metric. Usually the distance metric is such that the distance between closely related objects is minimised and that between different objects is maximized. In deep metric learning the underlying principle is the same apart from the fact that we use Deep Neural Networks to learn the discriminative factors automatically.

Nearest neighbour based image retrieval can be one of the potential applications of metric learning since in metric learning we essentially bring similar objects closer and dissimilar objects further. Therefore using this technique images that are very similar to each other based on the user query can easily be retrieved.

A typical use of metric learning can be in **One shot learning** applications. This is mainly due to the fact that classical Neural Networks when used for classification require a lot of training samples to learn the appropriate weight parameters. Usually such a large number of samples are not available for different tasks like face recognition where the network might have to learn to recognize a face in one or a few number of image samples. Metric learning can solve this problem by using a similarity function such that the distance between unrelated faces or objects becomes larger than a particular threshold, which would not require a lot of training samples for every new category introduced. Instead the training examples are required only for developing a proper similarity metric/function.

Deep Neural Networks

Application of traditional machine learning techniques requires handcrafted features, developing which demands a considerable amount of engineering skill and domain expertise. This, however, is not true for neural networks, which automatically learn these features from data using a general-purpose learning procedure.[11, 12] A standard neural network consists of many simple, connected processors called neurons, each producing a sequence of real-valued activations. Input neurons get activated through data applied at the input of the network. Other neurons get activated through weighted connections from previously active neurons. Each neuron can be seen as a single unit applying a non-linear activation function (such as sigmoid, tanh, ReLU) to a linear combination of the input activations to the neuron.[13]

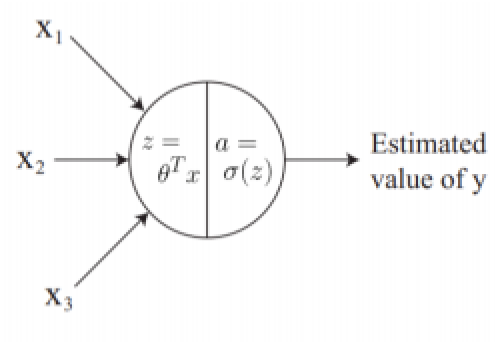


Figure 3. A single Neuron

These single neurons can be stacked so that one neuron passes its output as input into the next neuron. The resulting network of neurons can, hence, consist of several layers of neurons, each with their own learnable weights and biases. Used in conjunction with an appropriate loss function and optimization algorithm, such a network can be used to learn any complex function, if sufficient data is available for training. Forward propagation through the network yields its prediction for a given input. This prediction is compared with the actual class label, and the loss is computed. Backward Propagation is used to compute the gradients of the loss function with respect to the parameters, which are then used by the optimization algorithm to adjust the parameters and minimize the loss over a number of iterations.

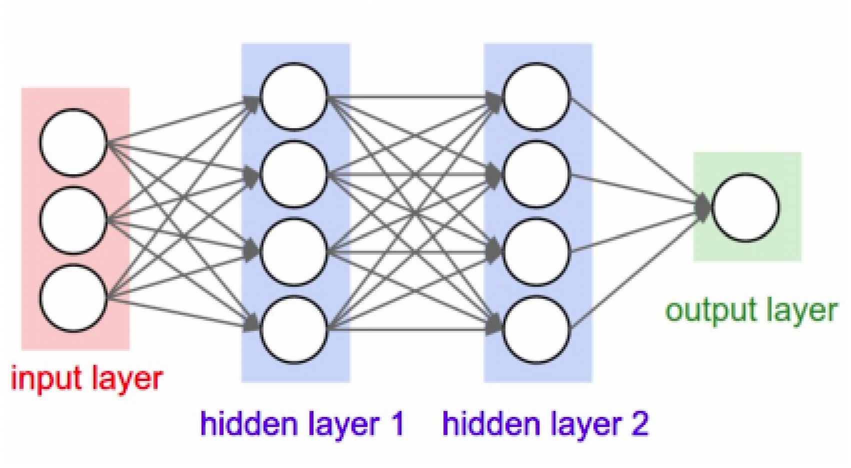


Figure 4. A two layer neural network with fully connected layers.

Convolutional Neural Networks

Regular neural networks do not scale well to full images. If the input to the neural network is a 200x200 RGB image, the number of weights for each neuron will be 200*200*3 = 120,000 weights. For large networks, the total number of learnable parameters become very large and lead the model to potentially overfit the training data, unless the training set is adequately large. A convolutional neural network (CNN) is a sequence of layers. Each layers transforms an input volume (images are represented as a three dimensional matrix) of activations to another with some differentiable function which may or may not have parameters.[12, 16] These layers are of three main types:

Convolution Layer

This is the core building block for convolutional networks. It is based on the convolution operation on images. Each convolutional layer of a CNN consists of N kernels or filters of a certain volume of neurons sized f ×f ×d, with f being the spatial dimension and d being the number of feature channels of the kernel, which is same as the number of channels in the image at its input (Di). Every one of these filters is slid through the entire image of size Hi×Wi×Di . Convolution refers to the summation of the element-wise dot product of the neurons in each filter with the corresponding values in the input, for each position in which the filter 8 is aligned with the image. Based on this notion, a convolution with a single filter at each layer results in a two dimensional output of a certain size. [12, 15, 16]

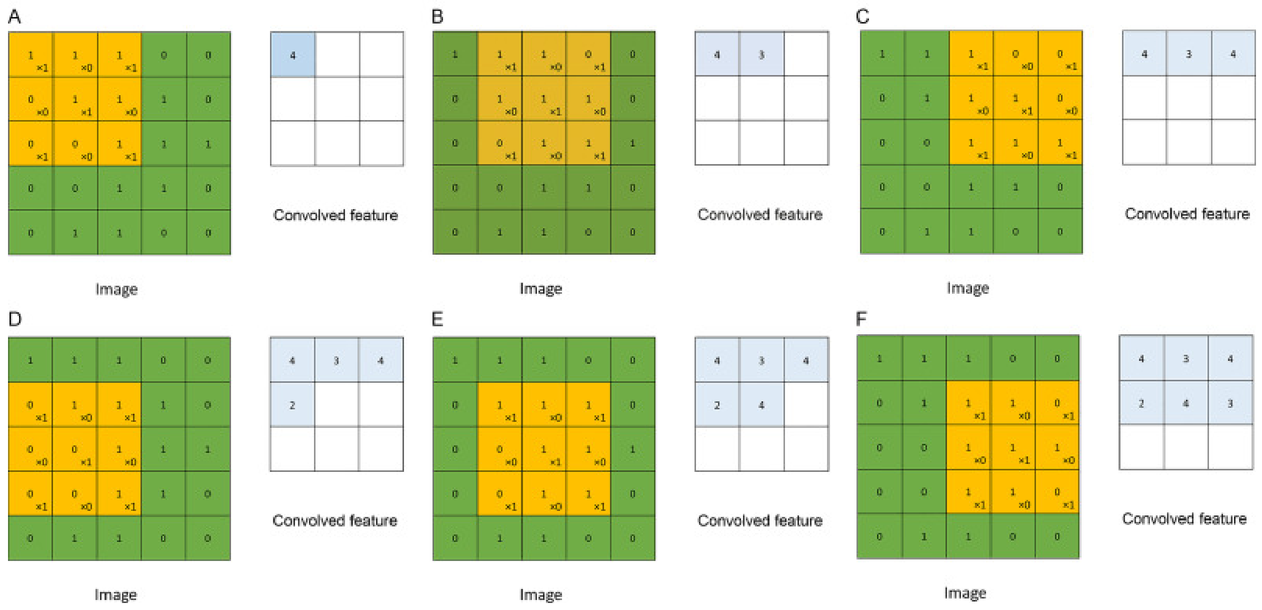


Figure x. The convolution operation performed using a 3x3 filter on a 5x5x1 image.

The intervals with which the filter moves in each spatial dimension is decided by the stride s. In order to prevent undue shrinkage of the volume along the spatial dimension, the image can be padded with pixels along the outer edges. The width of this padding, in pixels, is given by another hyper-parameter, p.[16, 15] The convolution operation is repeated for each of the N filters, and the resulting N activation maps are stacked together across the third dimension giving an output volume of dimensions.

$$H_o = \frac{H_i - f + 2p}{s} + 1$$

$$W_o = \frac{W_i - f + 2p}{s} + 1$$

$$D_o = N$$

In a convolutional layer, each neuron is connected to only a local region of the input volume, called the receptive field of that neuron. The extent of connectivity is limited to the filter size along the spatial dimensions, but is full along the depth axis.[12] It should also be noted that all activations belonging to a particular channel in the output volume correspond to a single filter applied on the input volume, and hence depend on the same. shared parameters. Local connectivity and parameter sharing not only help reduce the number of learnable parameters, but also make the CNN good at capturing translation invariance. This makes them an ideal choice for the image classification problem.

Pooling Layer

It is common to periodically insert a Pooling layer in-between successive convolutional layers in a CNN. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and hence to also control overfitting.[12] The most common form of pooling layer in CNN architectures employs filters of size 2×2 with a stride of 2, taking a max over 4 cells of the input image. It is worth noting that while this halves the width and height of the image, the depth remains unaffected as the max operation is applied independently to each channel of the image. This down-sampling process effectively discards 75% of the activations.

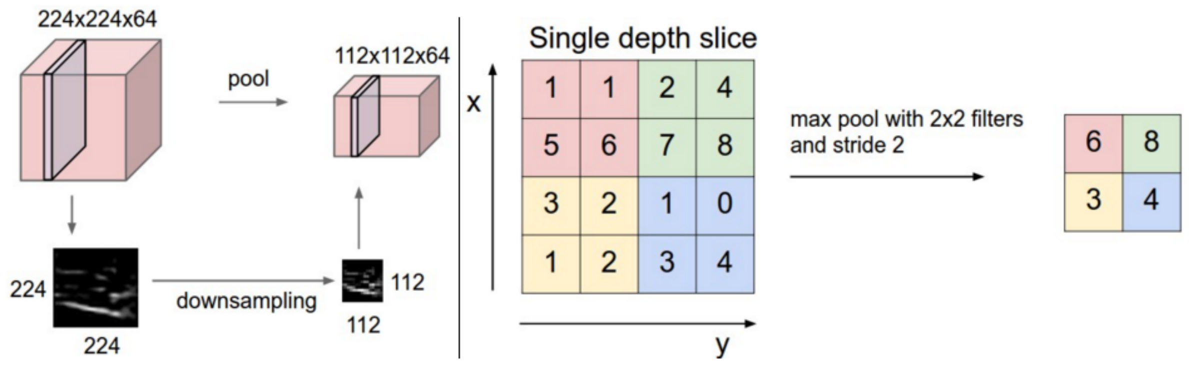


Figure 5. : (1) A typical max pooling layer. (2) The max pooling operation. [12]

Fully Connected Layer

Once higher level features are detected from the preceding convolution and pooling layers, a fully connected layer is usually attached at the end of the network. This layer is fully connected to all activations in the previous layer, as in regular neural networks, allowing all the features learned by the network to be taken into account by the output layer. All of these different types of layers can be stacked together in various ways to form a CNN.

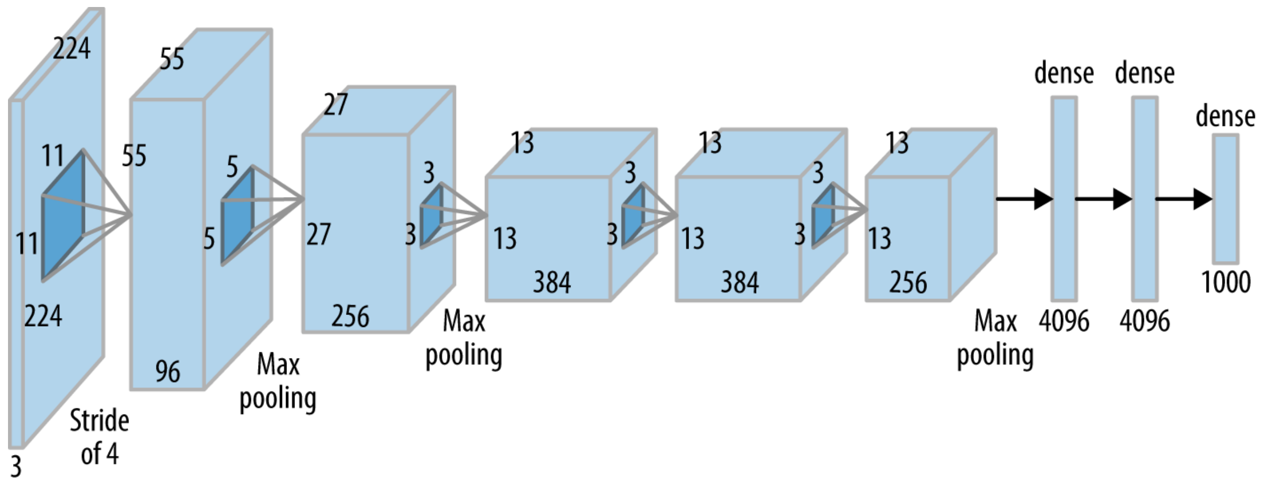


Figure 6. AlexNet - an influential NN architecture that popularized the use of CNNs and GPUs to accelerate deep learning.

Siamese Networks and One Shot Learning

The idea of the Siamese network was first described in a 2015 paper, Facenet[5]. The authors substituted the use of direct Deep Neural Network for classification with a Siamese Network based on which the similarity of inputs was determined. The approach was called one shot learning as it did not require a lot of training samples for each category. New categories could be added directly at run time without affecting the results unlike the traditional neural networks which require the categories to be predefined and a lot of training samples are required for each category.

Siamese neural networks, also known as twin neural networks consist of a neural network across which the two inputs are run. The two inputs run through the same neural network and a feature vector is received as output from the neural network. Based on the feature vector we compute a similarity metric and minimize the distance between similar items.

The common learning aim is to minimize a distance metric for similar objects and maximize for distinct ones. This gives a loss function like:

$$\delta(x^{(i)}, x^{(j)}) = \begin{cases} \min \ \| \mathrm{f}(x^{(i)}) - \mathrm{f}(x^{(j)}) \| \, , i = j \\ \max \ \| \mathrm{f}(x^{(i)}) - \mathrm{f}(x^{(j)}) \| \, , i \neq j \end{cases}$$

i, j are indexes into a set of vectors

$\mathrm{f}(\cdot)$ function implemented by the twin network

One of the most commonly used metrics is the Euclidean distance metric. The formula for which is as follows.

$$\delta(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \approx (\mathbf{x}^{(i)} - \mathbf{x}^{(j)})^T (\mathbf{x}^{(i)} - \mathbf{x}^{(j)})$$

The figure below shows a pictorial representation of the Siamese Neural Network Architecture.

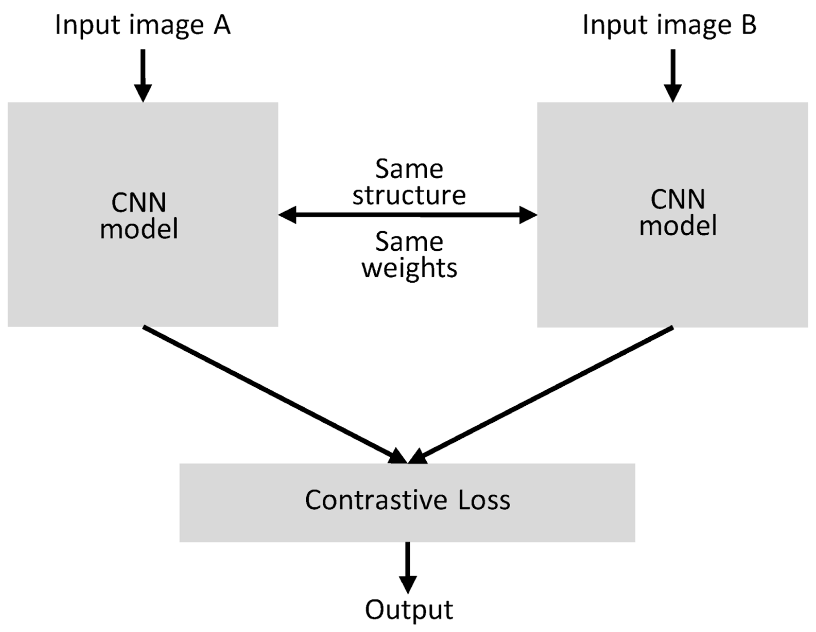


Figure 7. Siamese Neural Network Architecture

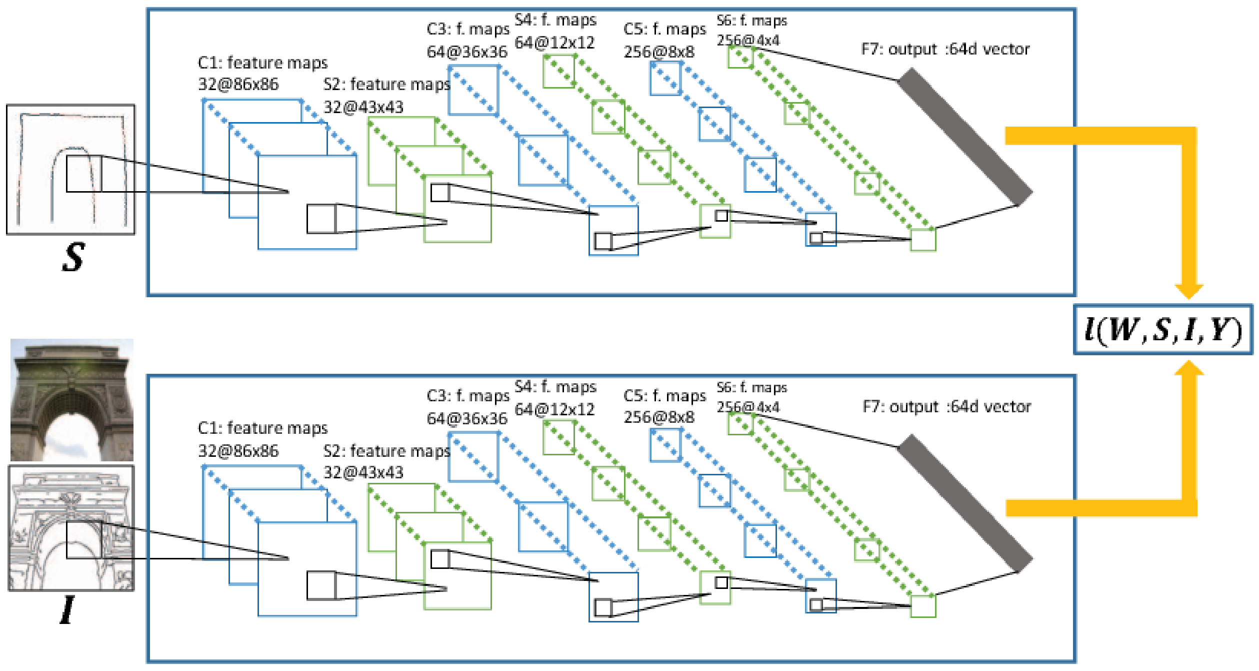


Figure 8. Sample Siamese Neural Network Architecture

Triplet Loss

In triplet loss[5], the loss function is such that it consists of a positive point and a negative point for an anchor data point. The loss function is calculated as a subtraction of the distance between anchor and negative point from the distance between the anchor and the positive point added with a margin alpha for soft margin treatment and so that the distance metric is not computed in such a way that all the distances are zero to satisfy the equation. Due to such a nature of the loss function, to minimize its value the distance between the anchor points and the positive points need to be minimised and that between the anchor and the negative points needs to be maximized.

It is often used for learning similarity for the purpose of learning embeddings, such as learning to rank, word embeddings, thought vectors, and metric learning.

For example take a neural network that recognizes the faces for unlocking a smartphone. The classifier for this task needs to ensure that the distance between the image captured earlier and the same person is minimized and the distance with the other faces is maximized beyond a certain threshold to unlock the phone. This is one of the applications of Triplet Loss.

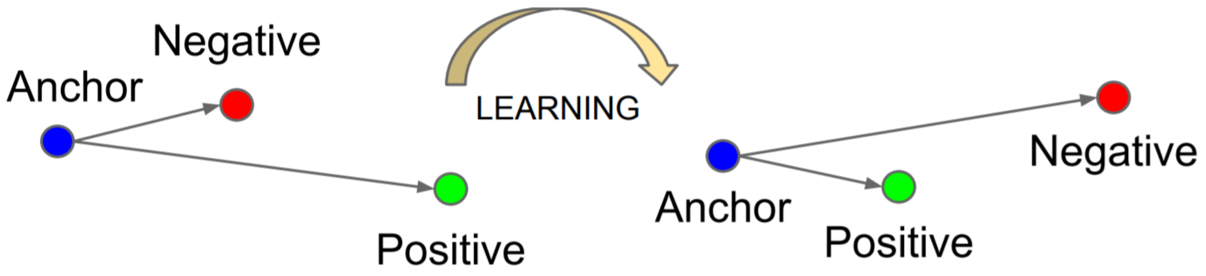


Figure 9. Triplet Learning: Anchor, Positive and Negative

The loss function can be described using a Euclidean distance function

$$\mathcal{L}(A,P,N) = \max\left(\| \mathrm{f}(A) - \mathrm{f}(P)\|^2 - \| \mathrm{f}(A) - \mathrm{f}(N)\|^2 + \alpha, 0\right)$$

where A is an anchor input,P is a positive input of the same class as A, N is a negative input of a different class from is a margin between positive and negative pairs, and is an embedding.

$$\mathcal{J} = \sum_{i=1}^{M} \mathcal{L}\left(A^{(i)}, P^{(i)}, N^{(i)}\right)$$

Implementation

A convolutional neural network based on the above architecture was implemented using TensorFlow in Python. However, some changes were made to enhance the performance of the model.

1. As a neural network is trained, the distribution of each layer's inputs changes continuously, as the parameters of previous layers change. This slows down learning and necessitates careful parameter initialization. This problem of internal covariate shift can be addressed by normalizing layer inputs to a learnable mean and variance, stabilizing the distribution of the activations. Batch Normalization allows us to use higher learning rates and makes the network more robust to the choice of hyper-parameters. Additionally, it also has a regularizing effect and helps prevent Overfitting.

2. The shape of the image accepted as input by the network was fixed to 256 × 256 x 1 by converting both drawings and photos to grayscale images as a part of preprocessing.

Preprocessing

The drawing images are grayscale images made with a pencil and do not have any color attribute. Since the colors can not be incorporated into pencil based drawing or most of the drawings on touch screen devices it is very unlikely to affect the output images.Therefore we convert the photos also to grayscale images and normalize both photos and drawings by dividing each by 255.

Architecture

The network consists of 3 2D convolutional layers of size 32 x 32, with ReLu activation function. Each layer is followed by a max pooling layer and batch normalization. The output of these layers is then flattened and fed as input into a fully connected layer with 128 neurons from which the final feature vector is received as output. The architecture was selected after a bunch of trial and error methods based on the RAM and GPU limitations of Google Colab which prevented us from selecting a better and more complex model.

Optimizer

Adam (Adaptive Moment Estimation) is an optimization algorithm designed specifically for deep neural networks that uses adaptive learning rates for each parameter.[25] It computes exponentially weighted averages of gradients and squared gradients.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

where m0 and v0 are zero-initialized and g t represents the gradient of the cost function with respect to parameter θ at step t. These are used for updating the parameters after every step of back-propagation

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v_t} + \epsilon} m_t$$

Adam reduces the oscillations in gradients associated with mini-batch gradient descent. The smoother loss curve allows for a larger learning rate and hence, quicker convergence.
In limited experiments, Adam was found to produce better results for our network as compared to other optimizers such as stochastic gradient descent.

Results

Given a drawing, the model retrieves the top k most relevant images to the drawing in the database. Retrieval @3 and @5 was used for measuring the performance of the model or all the relevant images that have a distance metric less than a threshold value.

Even though a very naive convolutional neural network model was used in the Siamese network for computing the feature vector, the results were promising.

At a threshold of similarity distance metric equal to 0.3 the model retrieved the relevant images for 43.22% of the drawings. In retrieval @3, for 20.4% drawing queries, the relevant image was retrieved as one of the three images whereas in retrieval @5, for 27.3% drawing queries the relevant image was retrieved as one of the five images. The percentage of queries for which the relevant result is retrieved increases with increase in the number of images retrieved for each query simply due to the randomness and therefore retrieval at larger numbers has been avoided.

Even though the model used was very naive the results were promising. It was an increase in random probabilistic accuracy by around 20 times. The naive CNN model had to be used due to RAM and GPU based limitations of Google Colab

which was used for the implementation of the project. Accuracy will significantly improve by using complex models like GoogleNet, ResNet etc.

Future Scope

Due to RAM and GPU based limitations on Google Colab only a very simple convolutional neural network in the Siamese Network on a partial dataset More complex neural networks like the GoogleNet, Alexnet etc could be implemented in the Siamese Network. Using the Network on the entire dataset would definitely yield better results. Also for the purpose of this project, we have limited ourselves to the images and drawings of animals only. The project can easily be extended to drawings of other images as well using the exact same techniques.

References

[1] KATO, T., KURITA, T., OTSU, N., AND HIRATA, K. 1992. A sketch retrieval method for full color image database-query by visual example. In Pattern Recognition, 1992. Vol.I. Conference

[2] RUSSAKOVSKY, O., DENG, J., SU, H., KRAUSE, J., SATHEESH, S., MA, S., HUANG, Z., KARPATHY, A., KHOSLA, A., BERNSTEIN, M., BERG, A. C., AND FEI-FEI, L. 2015. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) 115, 3, 211–252.

[3]C. Xiaochun, Z. Hua, L. Si, G. Xiaojie, and L. Liang. Symfish: A symmetry-aware flip invariant sketch histogram shape descriptor. IEEE International Conference on Computer Vision, 2013

[4] Li Liu et al.,” Deep Sketch Hashing: Fast Free-hand Sketch-Based Image Retrieval”,2017 ,Conference on Computer Vision and Pattern Recognition (CVPR)

[5]F. Schroff, D. Kalenichenko and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 815-823, doi: 10.1109/CVPR.2015.7298682.

[6] Shantanu Deshpande and Naman Goyal, “Sketch Based Image Retrieval”, 2018

[7]Patsorn Sangkloy et al., “The Sketchy Database: Learning to Retrieve Badly Drawn Bunnies”, 2017

[8]Tao Chen et al., “Sketch2Photo: Internet Image Montage” 2009, Siggraph Asia

[9]Jiwen Lu et al., “Neighborhood Repulsed Metric Learning for Kinship Verification”, 2014, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 36, NO. 2, FEBRUARY 2014

[10]B. Kulis,”A survey on metric Learning” Foundations and Trends inMachine LearningVol. 5, No. 4 (2012) 287–364 c 2013

[11] Manideep Kolla, Aniket Mandle, and Apoorva Kumar. Eye in the sky. GitHub Page. 2018. url: https://github.com/manideep2510/eye-in-the-sky

[12] Fei-Fei Li, Justin Johnson, and Serena Yeung. cs231n - Lecture Slides. 2017. url: http://cs231n.stanford.edu/slides/2017/

[13]Andrew Ng and Kian Katanforoosh. cs229 - Lecture Notes. 2018. url: http://cs229.stanford.edu/notes/

[14] Shivaprakash Muruganandham. Semantic segmentation of satellite images using deep learning. 2016.

[15] Andrew Ng, Kian Katanforoosh, and Younes Bensouda Mourri. Convolutional Neural Networks. 2017.