Proceedings of the Fifth International Conference on Communication and Electronics Systems (ICCES 2020)
IEEE Conference Record # 48766; IEEE Xplore ISBN: 978-1-7281-5371-1

Auto-Encoders for Content-based Image Retrieval with its Implementation Using Handwritten Dataset

Vaibhav Rupapara
Teladoc Health

Manideep Narra
Teladoc Health

Naresh Kumar Gonda
Teladoc Health

Kaushika Thipparthy
Teladoc Health

Swapnil Gandhi, *Mileset Softwares, Pune, India*

Abstract*—Image retrieval technology is a very fast-growing digital technology for researchers in the field of computer science from a very long period. It is a system for retrieving digital images from a large database. The well-known organizations that are using this system are Google and Pinterest. In this conference paper, a content-based image retrieval system that uses an innovative type of neural network known as autoencoder is discussed and developed a basic system to understand it. The methodology that has been used is an unsupervised method which is a machine learning algorithm in which the system retrieves images without searching about its name, labels, and tags. This system retrieves images just by its visual information. This approach of image retrieval is known as Content-Based Image Retrieval (CBIR).

I. INTRODUCTION

The recent development in the field of technology has left us all amazed, as there is a vast increment in the utilization of digital cameras, smart devices, Internet as the human lives are more dependant on the digital world in which everything is digitized so amazingly. Multimedia data is growing immensely and to find and retrieve any relevant content in terms of an image from a large amount of data is quite challenging.[2] So, in this digital world, it is very essential to attain a very efficient system that can easily retrieve images from the large database according to the need of users. So, researchers have found a way to overcome this issue by developing a system that is known as CBIR using an Auto-encoder. This system searches out images from the database by comparing its color, nature, pixel values, in short, this system looks much deeper into an image to find out its perfect match. This system encodes a particular image through an encoder and then it decodes the image through the decoder. The compression of the image occurs between this encoding and decoding process. Auto-encoder works on an unsupervised machine learning technique which is specifically designed to overcome image retrieval problems. There are different types of auto-encoder from which one is chosen according to our requirements. In this research paper, a convolutional auto-encoder is used to build an image retrieval system that uses a neural network (a network that helps to differentiate between the images according to their shapes). It is referred as a process of machine learning because proper training is given to an auto-encoder of several stages which are Feature Extraction, Feature Computation, and retrieval of an image. The data set that is used for the sake of this paper is a hand-written and the framework that has been used is Keras-Deep learning which is an open-source library of python. Let's discuss every step in this system in detail.

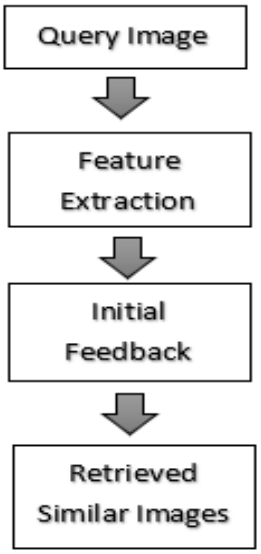

Figure 1 Steps involved in query and retrieval

II. CONTENT- BASED IMAGE RETRIEVAL SYSTEM

Content-based image retrieval (CBIR) is a system that can overcome the issue of efficient image retrieval as it is based on the analysis of images that are concerned with the query image. It is an application for computer vision techniques to deal with the image retrieval problem which can be defined as facing difficulty in finding digital images in large databases. In this definition, by Content, it means that it is associated solely with images instead of any other form of data like keywords, description, etc. This term can be associated with the color, type, dimension, texture, or anything regarding the image itself. A query image should be provided as an input which is the most fundamental articles, blogs, etc.[3] After the successful implementation of CBIR, it has been used in many other fields like in medical Disease Analysis through an image, Crime Detection, Textile Industry, Remote Sensing, etc.

The images can be used directly for checking their similarity, but there may be some problems which are:

- Images have a huge dimension.
- There may be a lot of redundancy in the pixels of images.

- Semantic information is not carried out by the pixel.

So, to overcome the above-mentioned issue, a model is used for retrieval that classifies the objects and then uses its features. The image that shows the closest features is given as output. It is also said that CBIR is a fancy name used for image search engines like a text search engine with a difference that it searches and then returns the most relevant images to you instead of any sort of articles, blogs, etc.

A. Content-based Image Retrieval Process

The process of CBIR comprises of searching large databases of digital image for retrieval of images query by the user which is sometimes a difficult task to be done due to several reasons which include similarity between the retrieved one and the queried one but this can be solved if the images are analyzed deeper on pixel-level because each pixel have different value and there are thousands and thousands of pixels in an image according to its resolution. So, it becomes much easier to differentiate images based on their pixel information. Another issue associated with image retrieval is time complexity while searching a very large database and this becomes an issue when it is dealt with big data.

III. AUTO-ENCODER

Auto-encoder is a machine learning technique that is an unsupervised technique. It is used to solve the image retrieval problems based on computer vision application. It comprises of the images that are visually similar or much alike that are stored in a certain database.[1] The neural network is capable enough of unsupervised feature learning.

Although neural networks are used for supervised learning problems but here for an autoencoder network, the most important challenge is to reconstruct the core of the original input from the compressed, corrupted, and noisy data that has been input.[4]

A. **Types of Auto-encoders**

There are many types of auto-encoder like Variational Autoencoder, Contractive Autoencoder, Denoising Autoencoder, Deep Autoencoder, Sparse Autoencoder, Undercomplete Autoencoder, Convolutional Autoencoder. All of them have different functionalities and purposes.[5]

B. **Components Required for Auto-encoders**

Autoencoder is a neural network that is unsupervised which means that it has no class labels or any sort of any labeled data which requires:

- An input (set of data).
- After accepting the set of data as an input, it compresses the data and represents it through a technique called Latin space.
- After compression, it rebuilds the input through this latent space representation to generate output.

So, it is also said that an autoencoder must be containing two essential components or maybe subnetworks which are:

- *ENCODER:* An Encoder is the one that accepts data as input to which it compresses into latent space.
- *DECODER:* Decoder accepts the latent-space representation and rebuilds the original input with latent space representation. [2]

C. **Latent space**

Latent space is a process to represent data in the neural network. It is very common in machine learning when working with images. It matches two images and maps similar data points close to each other. It is also said that a latent space is a space where the features of the images

Figure 2 Latent Space Representation

lie.[6]

It is also simplified that the compressed form or state of any data is latent space representation. For understanding its purpose lets assume a big data set that contains a large number of hand-written digits as shown in the above image (Figure 2). The images of the same digit like all the 2's have much more similarity than if the images of 2s with 5's is compared, which is actually due to the difference in the digit. For this purpose, the algorithm is trained. So, Latent space has much more importance in deep learning as it finds, identifies, and then learn the features of a particular data based on its patterns. Now a question arises that why it is called latent space, It is because any compressed form of data may not induce any sort of space, but this data point is a vector containing 3 dimensions (x, y, and z). By this, it is assumed that this data point can be a graph on a 3-dimensional plane concerning x, y, and z-axis.

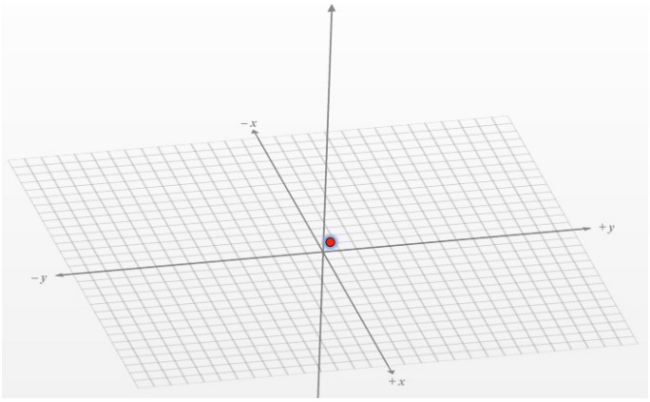

Figure 3 Graphical Representation

D. **Equation of Encoding/Decoding**

Mathematically, this encoding and decoding process can be written as the following equation;

$$O=D(E(x))$$

Equation 1:Encoding/ Decoding Equation

Where: 'x' is the input data, 'E' is the Encoder, 'D' is the Decoder & 'O' is the Output.

E. Main Concept of Auto-encoder

The concept behind Autoencoder is to construct a network that comprises of hidden, slender, and thin layer between Encoder & Decoder that presents a compressed presentation of the data that has been given as an input. The narrow middle layer that exists between the encoder and decoder can be named as "Bottleneck", through which the original data can be rebuild. When the input data is passed through an Encoder it creates a shrink or compressed form of that data that passes again through the decoder to reconstruct it into original form. The major difference between an Autoencoder and an engineered compression algorithm is that its compression and decompression functionalities are learned through the data itself without being purposely designed and then implemented programmatically.

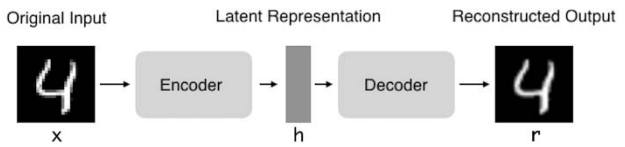

Figure 4 Encoding/ Decoding

It is observed in (Figure 4) that the input is a digit as an autoencoder after which an encoder network constructs its latent representation which is smaller as compared to the input in terms of dimension. After which the decoder network rebuilds the original input digit from the latent representation. The entire network is directed to reduce the difference between the input and output, after being compressed in the middle. Autoencoders once prepared are very explicit and will experience difficulty summing up to data collections other than those they were prepared on.

F. Why use Auto-encoder?

While Autoencoders can be utilized for compression, their exhibition is lossy, due to its characteristic of dimensional decrease. This implies that they're not exactly as effective as designed algorithms like JPEG, MP3, and so on. A progressively reasonable use for an Autoencoder is Denoising. A Denoising Autoencoder (DAE) is one that gets undermined or corrupted data as information and is prepared or trained to generate an uncorrupted data as output.

IV. NEURAL NETWORK

A neural network is based on deep learning a subfield of machine learning where algorithms are inspired by the structure of a human brain's neural network which takes in data and trains themselves about that data to recognize the pattern in the data and then predict the output with new sets of similar data according to the query. Let's understand how it's done. [7]

Create a neural network that differentiates between rectangle, square, and circle. Neural networks are made up of layers of neurons and these neurons are the core processing units of the networks. The first layer is the input layer which receives the input and the final layer is the output layer which shows the result, in between, there are hidden layers that perform most of the calculation to give us a result. Suppose if there is an image of a rectangle which is 28 by 14 pixels which means it has a total of 392 pixels. Each pixel value is then inserted into the separate neuron, the value of the pixel is called weight. The middle layers perform a calculation on those weights and finally give a value to match with the original value of the rectangle and this whole process is called forward propagation. If the system is guessing it right, then this network has a table where it keeps all the data of its prediction. If the guess is wrong, then the system moves back to the initial step and adjust the weight according to the final weight then again predict a result, this process is called backward propagation. By following this process, the networks train itself and mostly give an accurate result. The training time sometimes may take weeks, months, or even years.

V. AUTO-ENCODERS FOR CONTENT-BASED IMAGE RETRIEVAL

The idea proposed in this conference paper is using auto-encoders for content-based image retrieval system which is a very efficient image retrieval/search system. When training an auto-encoder will not use any labels or tags as mentioned above. The auto-encoder is then used to compute the latent-space representation for each image in the dataset. After this, during the search time t has observed the distance between the space vectors, the smaller the distance the more likely two images are.[8] Let's discuss how this technique works by building a basic Auto-encoder for CBIR. This process can be divided into further three parts.

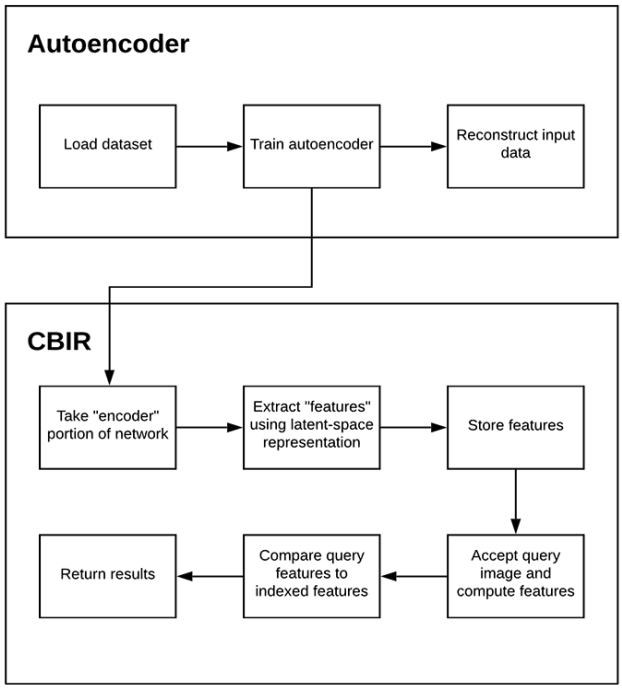

Figure 5 Working of CBIR with Auto-Encoder

A. Extracting features

Analyze how an auto-encoder can be used for image retrieval and then train an auto-encoder with a training set, the dataset consists of a large amount of data(images). It is required to teach an Auto-encoder in such a way to encode dataset images into latent space representation. When auto-encoder is trained enough then, Extract features of all the images that are present in the dataset by calculating the space representation. This representation works as a feature vector that quantifies the content of an image.

B. Computing features

Once all the images are encoded, it compares each of them Compare latent space to search out all the relevant images that are present in the dataset. Now, compare vectors by calculating the distance between them. The smaller the distance, the more similar they are, and if the distance is larger, they will be less similar.

C. Image Retrieval

This sorts out the result based on the distance i.e. from smallest to largest and displays the image to the user. Lastly, it will review the results for applying an auto-encoder for content-based image retrieval.

D. Training

Once the system retrieves an image according to query then it matches that either the retrieved images are correct or not. If the image retrieval is successful, then the system marks it as trained otherwise system changes the weight of the pixels according to final weight then redo the process, and by doing this, this system will train itself eventually. The training period may last for days, weeks, or even months depending upon the size of the database.

E. Query Result

Once all the images are encoded, it is compared. Compare latent space to search out all the relevant images that are present in the dataset. Now it compares the vectors by calculating the distance between them. The smaller the distance, the more similar they are, and if the distance is larger, they will be less similar. The results are sort out based on the distance i.e. from smallest to largest and display the image to the user. Lastly, it reviews the results of applying an auto-encoder for content-based image retrieval.

VI. AUTO-ENCODERS FOR CONTENT-BASED IMAGE RETRIEVAL

For building an image retrieval system with a convolutional auto-encoder that uses neural network layers for encoding and decoding of images. The dataset used is hand-written. The framework which has been used in this process is Kera's Deep Learning.

A. Kera's Deep Learning

Kera's is a library of python for Deep learning. It is open-source, easy to use, a powerful library for developing models in deep learning. It is especially recommended for beginners due to its modular and minimalistic approach.[9]

It is an API designed specifically for humans, not for machines, it is very useful for humans as it offers very simple and continuous APIs. It follows the easiest and best practices to reduce the load. Even in some cases it also reduces the actions taken by the user of common sense, it also provides errors clearly whenever it is required. It also has a proper kind of documentation and guidelines by the developer if needed. Due to these reasons, Kera's is the most used framework for deep learning. The most fascinating property of Kera's is that it allows and makes it easier to run new experiments and facilitates us to try out more extraordinary ideas in a better way.

There are many Python Deep Learning libraries out of which Kera's is the most favorite, the reason behind this is that It works as a wrapper of Theano or TensorFlow which are two computational libraries, which means that it can easily switch between these two depending on our requirements. In some of the common neural network structures, it has out of the box implementations.[10]

Figure 6: Handwritten dataset example

The data set that are used to train this model is a handwritten data set consist of digits from 0-9 being randomly repeated as shown in figure 6.

B. Convolutional Auto-encoder

It is a neural network and a special model of unsupervised learning which is designed to take input of an image and reproduce this image in the output. The image is passed through an encoder and decoder in between which it compresses.[11] The ConvNet compresses this image in the encoding phase and then the decoder ConvNet decodes it and rebuilds the compressed image. By this it is clear that the encoder is compressing the image data and decoder is building this image back to its original condition which shows that autoencoders can be used for the compression of data, for searching images, making the images noise-free and producing a clearer and clearer image from a previous one.

C. How does it work?

when the user makes a query, the system starts searching the database for similar images. After the query is called the system compare the query code with the code present in the database of digital images and finally searches for the closet image. For this comparison, the nearest-neighbors technique is used in latent space representation.

D. Nearest Neighbors

Nearest Neighbors is an algorithm through the process of retrieving the most similar code is applied. The idea behind this algorithm is to search among the samples that are already defined and are very near to the new point. Although there are many distance measures, Euclidean distance is the most common.[4] Mathematically, if 'q' is for the required image, 's' is for the sample, and their dimension is 'n', then

distance can be calculated by following mathematical formula:

$$d(q,s) = \sqrt{(q_1 - s_1)^2 + (q_2 - s_2)^2 + \ldots + (q_n - s_n)^2}$$

Equation 2: Nearest Neighbors calculation formula

E. Training data code

The code below is used to train the data and make the system learn how it represents the data in latent space.

```
input_img = Input(shape=(28,28,1))
x = Conv2D(16,(3,3), activation='relu',
padding='same')(input_img)
x = MaxPooling2D((2,2), padding='same')(x)
x = Conv2D(8,(3,3), activation='relu', padding='same')(x)
x = MaxPooling2D((2,2), padding='same')(x)
x = Conv2D(8,(3,3), activation='relu', padding='same')(x)
encoded = MaxPooling2D((2,2), padding='same',
name='encoder')(x)
x = Conv2D(8, (3, 3), activation='relu',
padding='same')(encoded)
x = UpSampling2D((2, 2))(x)
x = Conv2D(8, (3, 3), activation='relu', padding='same')(x)
x = UpSampling2D((2, 2))(x)
x = Conv2D(16, (3, 3), activation='relu')(x)
x = UpSampling2D((2, 2))(x)
decoded = Conv2D(1, (3, 3), activation='sigmoid',
padding='same')(x)
autoencoder = Model(input_img, decoded)
autoencoder.compile(optimizer='adam', loss='mse')
encoder = Model(inputs=autoencoder.input,
outputs=autoencoder.get_layer('encoder').output)
```

F. Testing of system

After training the data it is now time to test how accurate it is. The encoding process is done in the training session now it is time for decoding by the following code.

```
Encoder = Model(inputs=autoencoder.input,
Outputs=autoencoder.get_layer('encoder').output)
```

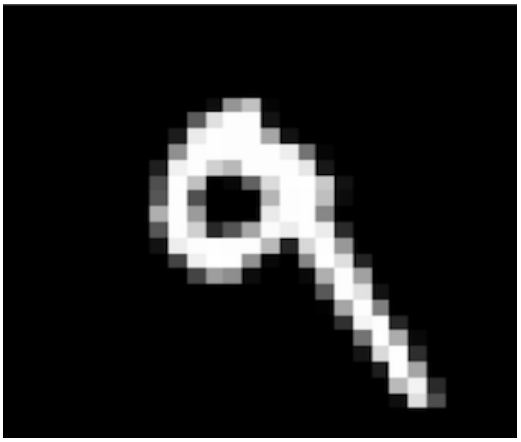

Figure 7 Query Image

Figure 7 is the query images with 9 written in it. This image is passed as an input to the system, the system starts comparing the query image code with the code present in the dataset and finds a similar image according to the nearest neighbor technique.

G. Result

The system retrieved 5 images from the dataset, and are amazed to see that all the images are similar to the query image.

Figure 8 result of query image

VII. CONCLUSION

In this proposed work, the basic concept of a simple image retrieval system through an autoencoder and the nearest-neighbor algorithm is studied. It is initiated with the training of autoencoder on a gigantic dataset so that it can be capable enough to encode effectively the visual content of each image. Then the code of the image is compared with the code of the images present in our handwritten dataset. The overall process does not use any labels or tags. Effective researches and processes are being performed to improve the results and quality of this model by using different auto-encoders other than convolutional. It has so many applications in many other fields in which this model helps a lot so for attaining the accuracy and maintaining the quality of this system more and more processes and researches are being performed so that it gets accurate results in other fields as well.

References:

[1] N. Hubens, "Build a simple Image Retrieval System with an Autoencoder," *Medium*, Aug. 24, 2018. https://towardsdatascience.com/build-a-simple-image-retrieval-system-with-an-autoencoder-673a262b7921 (accessed May 15, 2020).

[2] M. I. Daoud, A. Saleh, I. Hababeh, and R. Alazrai, "Content-based Image Retrieval for Breast Ultrasound Images using Convolutional Autoencoders: A Feasibility Study," in *2019 3rd International Conference on Bio-engineering for Smart Technologies (BioSMART)*, Apr. 2019, pp. 1–4, doi: 10.1109/BIOSMART.2019.8734190.

[3] "Autoencoders for Content-based Image Retrieval with Keras and TensorFlow," *PyImageSearch*, Mar. 30, 2020. https://www.pyimagesearch.com/2020/03/30/autoencoders-for-content-based-image-retrieval-with-keras-and-tensorflow/ (accessed May 15, 2020).

[4] J. Brownlee, "Your First Deep Learning Project in Python with Keras Step-By-Step," *Machine Learning Mastery*, Jul. 23, 2019. https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/ (accessed May 15, 2020).

[5] "Different types of Autoencoders," *OpenGenus IQ: Learn Computer Science*, Jul. 14, 2019. https://iq.opengenus.org/types-of-autoencoder/ (accessed May 15, 2020).

[6] I. A. Siradjuddin, W. A. Wardana, and M. K. Sophan, "Feature Extraction using Self-Supervised Convolutional Autoencoder for Content based Image Retrieval," in *2019 3rd International Conference on Informatics and Computational Sciences (ICICoS)*, Oct. 2019, pp. 1–5, doi: 10.1109/ICICoS48119.2019.8982468.

[7] M. A. Nielsen, "Neural Networks and Deep Learning," 2015, Accessed: May 15, 2020. [Online]. Available: http://neuralnetworksanddeeplearning.com.

[8] A. Sze-To, H. R. Tizhoosh, and A. K. C. Wong, "Binary codes for tagging x-ray images via deep de-noising autoencoders," in *2016 International Joint Conference on Neural Networks (IJCNN)*, Jul. 2016, pp. 2864–2871, doi: 10.1109/IJCNN.2016.7727561.

[9] "Keras: the Python deep learning API." https://keras.io/ (accessed May 15, 2020).

[10] A. Baaj, "Keras Tutorial: Content Based Image Retrieval Using a Convolutional Denoising Autoencoder," *Medium*, Dec. 06, 2019. https://medium.com/sicara/keras-tutorial-content-based-image-retrieval-convolutional-denoising-autoencoder-dc91450cc511 (accessed May 15, 2020).

[11] X. Yuan and C.-T. Li, "CBIR approach to building image retrieval based on invariant characteristics in Hough domain," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, Mar. 2008, pp. 1209–1212, doi: 10.1109/ICASSP.2008.4517833.

[12] K. Ramanjaneyulu, K. V. Swamy and C. S. Rao, "Novel CBIR System using CNN Architecture," 2018 3rd International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 2018, pp. 379-383, doi: 10.1109/ICICT43934.2018.9034389.