IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Building An Image Search Engine With Deep Learning

,K. Praveen Kumar[1], V. Prakash Reddy[2] G. Indra Karan Reddy[3,] N.S. Ganesh[4]

SreeNidhi Institute of Science and Technology, Hyderabad

[1,2,3] UG Students,[4]Assistant Professor Hyderabad, Telangana

Abstract: In this paper we are presenting an Image search engine build using deep learning. Automation is everybody's dream whether it is to ease their efforts or to make machines learn how to do something. In this paper we are presenting a way to search for similar images in local disk when an input image is applied. We used CNN for feature extraction and Flask for implementing the program functionality in the local webpage. The model is used to extract the features as NumPy arrays and store them locally and be able to search for similar images when and input is applied.

Keywords: Image, Image search, convolutional neural network, Feature vector, Artificial intelligence.

I. INTRODUCTION:

With the rise of users having access to high quality cameras whether its from their phones or from a external camera source and access to high speed internet for searching for images or text or videos and be able to download them right away from the internet, this demands for filtering,sorting and organising these images which can save up alot of time which is used up if manually searching for the images we want,it is possible to manually filter images but it becomes impossible when dealing with huge datasets hence we need a machine that can sort large datasets.

In this Paper we are proposing a deep learning model which is capable of returning similar images when a input image is applied. A pretrained CNN model known as VGG16 is used. CNN is used to extract features and Flask is used to deploy the model in to a local webpage. The images that are getting returned should be similar to the input [2].Most of the previous methods are based on extracting pixel information and manually saving all the data and search through all the dataset which can be relaced by using a deep learning for better feature extraction.

In order to get similar images as output. We will be designing a single model which takes an input image and then extract features and save the extracted features in the database as NumPy arrays and have same name as the image. This project is divided in to 3 parts.First making an offline file which will preprocess the data and secound is making a feature extractor file which can be imported whenever we need to perform feature extraction.Third file consist of web application code which describe how the webpage works.

The paper proposes a model which returns similar images as input. The dataset used in this project contains 8000 images of different types of flowers. We are going to use VGG16 which consists of 16 hidden layers which is used for feature extraction. Flask is used to deploy the model in the webpage and acts as an interface between user and the programm.

Fig.1 Example of input

II. APPROACH:

In this paper we are purposing a model which contains a CNN and flask. CNN is used for image preprocessing also called as encoding and Flask is used to deploy the model in thelocal webpage. All these neural networks can be mathematically represented for better understanding and easier analysis.

A. *Convolutional Neural Network:*

Convolutional neural networks are best suited for image processing and visual analysis. The reason they are used is that the image can be easily manipulated by convoluting image data with a filter data for easier manipulations. CNN contains multiple layers, layers after CNN are connected as multilayer neural networks. The design of CNN allows it to be able to take a 2D image as an input. The output can be achieved by using multiple layers and weights just like in a neural network by convoluting one layer to another for manipulations and with various pooling techniques to get the feature vector.

In this Model we used VGG16 which consists of 16 hidden layers for its model, which is a deep learning CNN model. This model is a pretrained model capable of categorizing 1million types of images. This model is used for feature extraction which can be further utilized for training.

Preprocessing using CNN:

Preprocessing of data implies that the input data is converted in a form that the computer can understand that easy training. Preprocessing is used to improve the performance by manipulating the image so that it can be easily understood by the machine. As VGG16 accepts only 224 x 224 images it is important to first resize the images so it can be fed in to VGG16.This reduction in size greatly enhances the performance as it has less data to work on.

Filtering:

In CNN, the main application of Convolution is done in filtering. In filtering the input image vectors are multiplied with the filter to get the modified output that we need. There are several types of filters such as sharpening, grayscale, blur and etc.

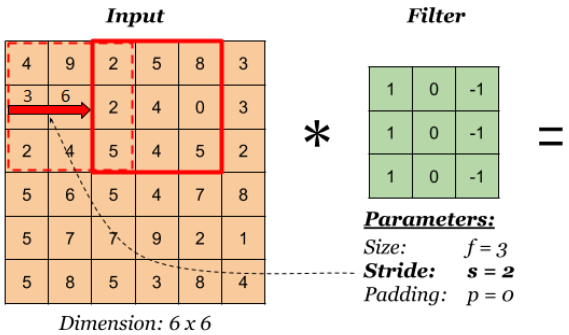

Fig 2: Filter

Feature extraction:

Just like the above filters CNN filters are used to extract features from an image,when compared to the usual filters CNN filters does not have any predefined values and these values are determined during the training period this helps the model to make filters by its own which can result in pretty amusing filters that humans can never think of manually. A 2D convolution filter is commonly used and it is referred as Conv2D.This filter adds up all the inputs and a single output is obtained from the image.

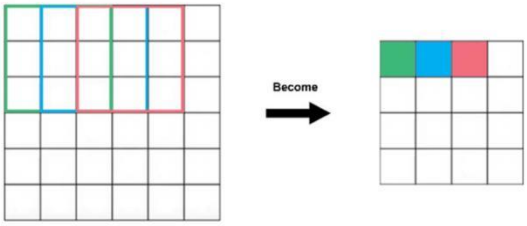

Fig 3. Feature Extraction

B. Flask:

Flask is a Python-based microweb framework. It is referred to as a microframework because it does not necessitate the usage of any specific tools or libraries. [2] It doesn't have a database abstraction layer, form validation, or any other components that rely on third-party libraries to do typical tasks. Extensions, on the other hand, can be used to add application functionalities as if they were built into Flask itself. Object-relational mappers, form validation, upload handling, different open authentication protocols, and other framework-related tools all have extensions..

II. METHEDOLOGY:

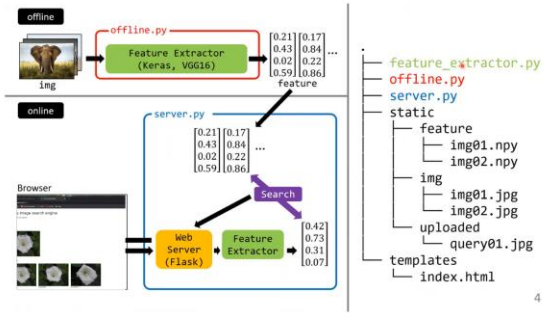

Fig.4 Methedology.

We divided the project into 3 parts:

1. Offline.Py (Save features into numpy arrays)
2. Feature extractor.Py (Contains VGG16 for preprocessing)
3. Server.Py (contains Flask for web interactions)

Offline.py:

- This Python file consists of code that is used to get the images from local directory
- It is used to initialise Features folder which contains all the feature as Numpy arrays.
- This File is able to extract features by calling the feature extractor class.

Feature Extractor.py:

- This file consists of code required for preprocesssing the images.
- Here we are using VGG16 which is a pretrained CNN model for feature extraction by eleminating the classification layer from the model
- This model imports weights from imagenet.
- We apply Normalization and then return numpy arrays.
- This model converts the image info to numpy arrays that are stored locally.

Server.py:

- This file contains code for web implimentation of our model.
- We are using flask web framework for interfacing between the model and the user.
- We make use of feature extractor class for features of user input image.
- This file is responsible to host the web page locally and display output on the webpage

IV. RESULT:

A. Dataset:

The data set consists of 8000 images with various types of flowers.

B. Results:

The model returns first 30 closest images based in the score from distance between the images.

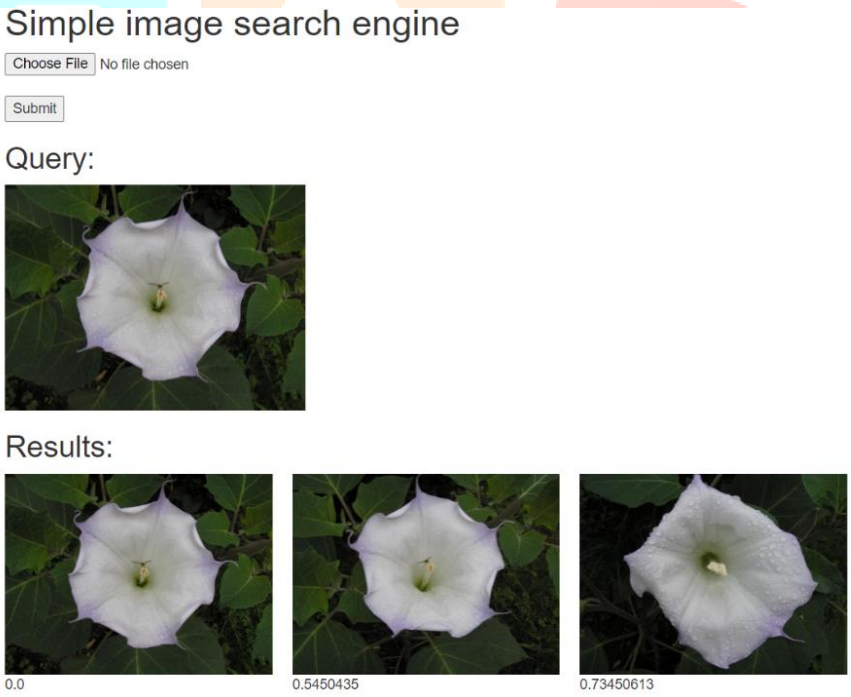

Fig 5. Results

ACKNOWLEDGEMENT:

The authors would like to express sincere gratitude to the management of SreeNidhi Institute of Science and Technologyfor their continuous support and encouragementin this work.

REFERENCES:

[1] Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma, "A survey of content-based image retrieval with high-level semantics," Pattern recognition, vol. 40, no. 1, pp. 262–282, 2007.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[3] C. Szegedy,W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.

[4] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2818– 2826.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097–1105.

[6] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," in European conference on computer vision. Springer, 2014, pp. 584–599.

[7] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan, "Supervised hashing for image retrieval via image representation learning." in AAAI, vol. 1, 2014, p. 2.

[8] K. Lin, H.-F. Yang, J.-H. Hsiao, and C.-S. Chen, "Deep learning of binary hash codes for fast image retrieval," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2015, pp. 27–35.

[9] J.-C. Chen and C.-F. Liu, "Visual-based deep learning for clothing from large database," in Proceedings of the ASE BigData & SocialInformatics 2015. ACM, 2015, p. 42.

[10] N. Khosla and V. Venkataraman, "Building image-based shoe search using convolutional neural networks," CS231n Course Project Reports, 2015.

[11] A. Iliukovich-Strakovskaia, A. Dral, and E. Dral, "Using pre-trained models for fine-grained image classification in fashion field," 2016.

[12] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition." in Icml, vol. 32, 2014, pp. 647–655.

[13] G. Shrivakshan, C. Chandrasekar et al., "A comparison of various edge detection techniques used in image processing," IJCSI International Journal of Computer Science Issues, vol. 9, no. 5, pp. 272–276, 2012.

[14] A. Maurya and R. Tiwari, "A novel method of image restoration by using different types of filtering techniques," International Journal of Engineering Science and Innovative Technology (IJESIT) Volume, vol. 3, 2014. [15] R. Kandwal, A. Kumar, and S. Bhargava, "Review: existing image segmentation techniques," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 4, no. 4, 2014.

[16] K. Roy and J. Mukherjee, "Image similarity measure using color histogram, color coherence vector, and sobel method," International Journal of Science and Research (IJSR), vol. 2, no. 1, pp. 538–543, 2013.

[17] J. Shlens, "Train your own image classifier with Inception in TensorFlow," https://research.googleblog.com/2016/03/train-your-ownimageclassifier-with.html, 2016.

[18] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," arXiv preprint arXiv:1312.6229, 2013.

[19] P. Wu, S. C. Hoi, H. Xia, P. Zhao, D. Wang, and C. Miao, "Online multimodal deep similarity learning with application to image retrieval," in Proceedings of the 21st ACM international conference on Multimedia. ACM, 2013, pp. 153–162.

[20] S. Liu, Z. Song, G. Liu, C. Xu, H. Lu, and S. Yan, "Street-toshop: Cross-scenario clothing retrieval via parts alignment and auxiliary set," in Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012, pp. 3330–3337.

[21] K. Yamaguchi, M. H. Kiapour, L. E. Ortiz, and T. L. Berg, "Retrieving similar styles to parse clothing," IEEE transactions on pattern analysis and machine intelligence, vol. 37, no. 5, pp. 1028–1040, 2015.

[22] J. Wan, P. Wu, S. C. Hoi, P. Zhao, X. Gao, D. Wang, Y. Zhang, and J. Li, "Online learning to rank for content-based image retrieval," 2015.