ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

An Efficient Approach for Image Retrieval Based on Xor Distance Using Binary Hash Codes

[1] Prateek Prince Thakur, [2] Arpit Chaudhary, [3] Vibhav Prakash Singh, [4] Rajeev Srivastava
[1] Department of Computer Science And Engineering Iit(Bhu), Varanasi

Abstract: Deep learning has given very promising results in the field of image retrieval. These results are generally built on the already existing deep learning systems that are used for the standard classification problem. A popular techniques in this domain employees binary hash codes which are used as the semantic representation of the image. The approach has a major bottleneck relating to the time of computation. Our idea is built around combining this approach and performing clustering based upon the hash code. The relevant clusters are then obtained by both direct matching as well as appropriate clusters within some small hamming distance.

Index Terms— Deep Learning, Machine Learning, Image retrieval, binary hash codes.

I. INTRODUCTION

Content-based image retrieval (CBIR) is a technique of searching for images in a large databases. This involves indexing all the images present in a database. This uses features which are directly extracted from the image instead of features that are obtained from the metadata associated with the image. Also known as query by image content (QBIC) and content-based visual information retrieval (CBIR) is the application of computer vision techniques to the image retrieval problem, that is, the problem of searching for digital images in large databases. The most common method for comparing two images in content-based image retrieval is using an image distance measure. An image distance measure compares the similarity of two images in various dimensions such as color, texture, shape, and others. Traditional approach in obtaining features of an image are using color histogram texture features and BoF using SIFT.

II. DATASETS

We have used ImageNet dataset which is a largescale dataset with over 15 million labeled high resolution images belonging to roughly 22, 000 categories. We have used 50,000 images of 1000 class (50 images of each class to index. ImageNet: is a large-scale dataset with over 15 million labeled high-resolution images belonging to roughly 22, 000 categories. The images were collected from the web and labeled by human labelers using Amazon's Mechanical Turk crowd-sourcing tool. Starting in 2010, as part of the Pascal Visual Object Challenge, an annual competition called the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) has been held. ILSVRC uses a subset of ImageNet with roughly 1, 000 images in each of 1,000 categories. In all, there are roughly 1.2 million training images, 50, 000 validation images, and 150, 000 testing images. The Deep CNN model in our framework is trained based on ILSVRC2012.

III. MATERIALS AND MODELS

CNN as a Feature Extractor: VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman. Its main contribution was to show that the depth of the network is the critical component for good performance. The model achieves 92.7% top-5 test accuracy in ImageNet. In this architecture, input to the ConvNets is a fixed size 224x224 RGB image.The image is passed through a stack of convolutional layers, where filters of size 3x3 are used. Binary Hash Codes: Binary hash codes are computed by binarizing each of the continuous values of H length image representation to (0/1).

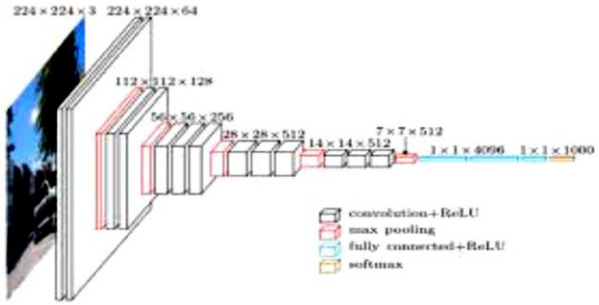

Figure1. VGG16 Model

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

IV. PROPOSED IMAGE RETRIEVAL APPROACH

A. 4.1 Convolutional Neural Networks (CNNs)

CNNs are very similar to ordinary neural networks ie they are made up of neurons that have learnable weights and biases.Each Neuron receives input and it produces output and passes it as input for next layer but the difference in ConvNets is that it assumes that the input is image and this allows us to make certain changes in network which makes it more efficient and the amount of parameters is reduced. The regular Neural Networks consists only fully connected layers and this can cause overfitting in case of images.For example in CIFAR-10, images are only 32x32x3(32 wide,32 high and 3 color channels), so a single first fully connected neuron will have 32x32x3 weights which is manageable but if the image size is 200x200x3 then the single first fully connected neuron will have 200x200x3=120,000 weights and their will more than one neuron.So number of parameters will increase quickly.Such a network with huge amount of parameters will lead to overfitting. ConvNets uses the fact that the inputs consists of images, so they constrain the architecture in a sensible way.In ConvNets, the layers having neurons are arranged in 3 dimensions (width,height and depth). Each layer transforms the 3D input to 3D outputs volumes of neurons.

4.2 Layers Used to build ConvNets:

There are four main types of layers to build ConvNets architecture:

1. Input: Every Image is represented in the form of matrix of Pixel values.For CIFAR-10 images, this matrix would we of size 32x32x3 (32 height, 32 width and 3 colors).
2. Convolutional Layer: It will compute the output of the neurons that are connected to the local regions of the input by computing dot product between weights of filter used and the small region of the image from which neuron is connected.
3. Non-Linearity(ReLu) Layer: ReLu stands for Rectified Linear unit and it is non linear operation. Its output is given by max(0,input). ReLu converts all negative values in a feature map to zero.The purpose of this layer is to introduce non-linearity in our ConvNet.
4. Pooling Layer: This layer reduces the spatial dimensions(width and height) of each feature map but retains the most important information. It can be of different types: Max pooling, Average pooling, Sum pooling.
5. Fully Connected Layer: It is the traditional Multi Layer Perceptron that uses a activation function in output layer(mainly softmax activation is used). In FC Layer every neuron of this layer is connected to the every neuron of previous layer. The purpose of FC layer is to use high level features from convolution and pooling layer for classifying image into various classes.

4.3 Models Used: VGGNet

VGG is a convolutional neural network model proposed by K. Simonyan and A. Zisserman in the paper “Very Deep Convolutional Networks for Large-Scale Image Recognition” .Its main contribution was to show that the depth of the network is the critical component for good performance. The model achieves 92.7% top-5 test accuracy in ImageNet , which is a dataset of over 14 million images belonging to 1000 classes. VGGNets are of two types : VGG16 (contains 16 weight layers) and VGG19(contains 19 weight layers).

4.4 Architecture of VGG model:

In this architecture, input to the ConvNets is a fixed size 224x224 RGB image. For preprocessing mean of RGB values over training data is subtracted from each pixel. The image is passed through a stack of convolutional layers,where filters of size 3x3 are used. Stride of the convolution is fixed to 1 pixel, the spatial padding of conv. layer input is such that the spatial resolution is preserved after convolution ie padding is 1 pixel for 3x3 conv layers. Max Pooling is performed over 2x2 pixel window with stride of 2. A stack of convolutional layers (which has a different depth in different architectures) is followed by three Fully-Connected (FC) layers: the first two have 4096 channels each, the third performs 1000-way ILSVRC classification and thus contains 1000 channels (one for each class). The final layer is the soft-max layer. The configuration of the fully connected layers is the same in all networks. The two types of VGG models we used are : VGG16 (contains 16 weight layers) and VGG19(contains 19 weight layers).

1. Feature Extraction using CNN: we take the activations of the two fully connected layers (FC1, and FC2) as the feature representations for CBIR tasks. In our experiments, we denote the feature vector of this direct feature generalization as “FC1” and “FC2”, respectively.FC2 is the features taken from the final

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

hidden layer, and FC1 is the activations of the layer before FC2. We do not evaluate features from lower convolutional layers in the network since the lower layers are unlikely to contain richer semantic representations than the later features which form higher-level hypotheses from the low-level to mid-level local information .The above direct feature generalization may work on the dataset used for training the CNNs model, but may not work well for CBIR tasks on a new dataset, as shown Figure 12, which may be very different from the original training data set. In the following, we discuss three kinds of feature generalization schemes in detail. We assume the retrieval domain is similar to the original dataset for training the CNN model. In this scenario, we will simply adopt one of the activation features FC1 FC2, directly. To obtain the feature representation, we directly feed the images in new datasets into the input layer of the pre-trained CNN model, and then take the activation values from the two layers. Since we only need to compute the feedforward network based on the matrix multiplication for one time, the whole scheme will be very efficient.

2. Clustering using Binary Hash of Features: The 4096 features vector obtained will be reduced to some much smaller number say H. Whenever a query image comes it's signature is computed and only images having same signature is searched for retrieving best k similar images so overall searching time is reduced. We have used hamming distance to match different binary hash signatures

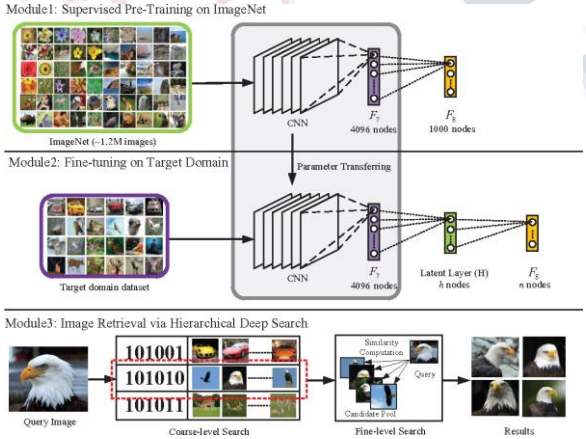

Figure 2. Model workflow

4.5 Final Approach

Step 1: We will be using a Hidden latent layer in network to learn representation from feature data. So the 4096 features vector obtained will be reduced to some much smaller number say H. This H1 dimension feature vector obtained can be binarized to get same dimension binary vector mapping to 2H unique memory slots. This binary vector can be used as an image signature so images having same signature can be indexed into same memory location.

Step2: Whenever a query image comes it's signature is computed and only images having same signature is searched for retrieving best k similar images so overall searching time is reduced.

Note: A query image can also be searched in the image clusters which are say 1 or 2 hamming distance away to corresponding to

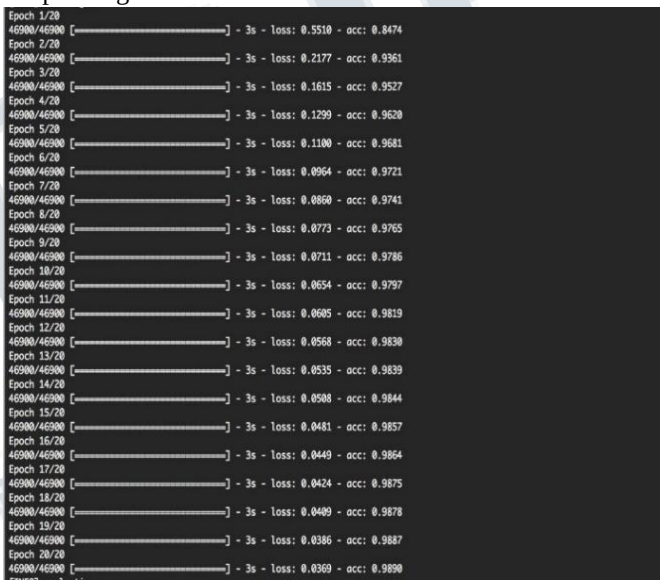

Figure 3: Training Epochs

V. RESULTS

Performance of our system for various hamming distance

RESULTS		
	Time(sec)	Accuracy(%)
No Hash	5.2	92.42
0 Hamming	0.02	78.89
1 Hamming	0.2	88.56
2 Hamming	0.5	90.22

Figure 4: Comparison between various hashes for accuracy and retrieval time

- Retrieval time is decreased by a large factor when

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

using binary hash codes.

• Increasing the hamming distance increases the accuracy but also increases the time.

• Highly scalable to different real world image datasets.

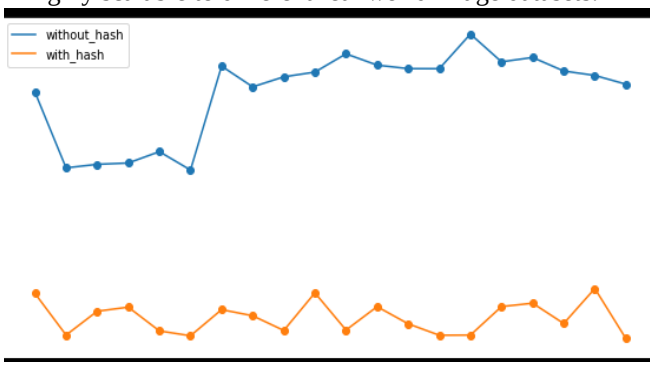

Figure 5 : Retrieval Time comparison

IV. CONCLUSION

Using Deep learning techniques with binary hash code reduces the search time considerably by reducing the search space for a given image. This approach can be utilized in a variety of settings like real time systems where slight loss in accuracy is a profitable trade off when weighed against significant reduction in time.

REFERENCES

[1] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, Q. V.Le, M. Z. Mao, M. Ranzato, A. W. Senior, P. A. Tucker, K. Yang, and A. Y. Ng. Large scale distributed deep networks. In NIPS , pages 1232–1240, 2012

[2] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations.In ICML , pages 11–18, 2003.

[3] H. Bay, T. Tuytelaars, and L. J. V. Gool. Surf: Speeded up robust features. In ECCV (1),pages 404–417, 2006.

[4] B. C. Becker and E. G. Ortiz. Evaluating open-universe face identification on the web. In CVPR Workshops, pages 904–911, 2013.

[5] Y. Bengio, A. C. Courville, and P. Vincent. Unsupervised feature learning and deep learning: A review and new perspectives. CoRR, abs/1206.5538, 2012.

[6] H. Chang and D.-Y. Yeung. Kernel-based distance metric learning for content-based image retrieval. Image and Vision Computing , 25(5):695–703, 2007.

[7] G. Chechik, V. Sharma, U. Shalit, and S. Bengio. Large sca le online learning of image similarity through ranking. Journal of Machine Learning Research, 11:1109–1135, 2010.

[8] D. C. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In NIPS, pages 2852–2860, 2012.

[9] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. Journal of Machine Learning Research, 7:551–585, 2006.