International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 04 Issue: 07 | July -2017 www.irjet.net p-ISSN: 2395-0072

Comparison of Various RCNN techniques for Classification of Object from Image

Radhamadhab Dalai[1], Kishore Kumar Senapati[2]

[1] *PHD Student ,Dept. of Computer science & Engineering, BIT Mesra, Ranchi, Jharkhand, India*
[2]*Professor, Dept. Of Computer science & Engineering, BIT Mesra, Ranchi, Jharkhand, India*

Abstract -Object recognition is a very well known problem domain in the field of computer vision and robot vision. In earlier years in neuro science field CNN has played a key role in solving many problems related to identification and recognition of object. As visual system of our brain shares many features with CNN's properties it is very easy to model and test the problem domain of classification and identification of object. Basically CNN is typically a feed forward architecture; on the other hand visual system is based upon recurrent CNN (RCNN) for incorporating recurrent connections to each convolutional layer. In middle layers each unit is modulated by the activities of its neighboring units. Here Various RCNN techniques (RCNN,FAST RCNN,FASTER RCNN)are implemented for identifying bikes using CALTECH-101 database and alter their performances are compared.

Key Words: DNN, CNN, RCNN, FAST-RCNN, FASTER RCNN

1. INTRODUCTION

Recently, techniques in deep neural networks (DNN) - including convolutional neural networks(CNN) [1]and residual neural networks - have shown great recognition accuracy compared to traditional methods (artificial neural networks, decision tress, etc.). However, experience reveals that there are still a number of factors that limit scientists from deriving the full performance benefits of large, DNNs. We summarize these challenges as follows: (1) large number of hyper parameters that have to be tuned against the DNN during training phase, leading to several data re-computations over a large design-space, (2) the share volume of data used for training, resulting in prolonged training time, (3) how to effectively utilize underlying hardware (compute, network and storage) to achieve maximum performance during this training phase. Fast R-CNN is an object detection algorithm proposed by Ross Girshick in 2015. Fast R-CNN builds on previous work to efficiently classify object proposals using deep convolutional networks. Compared to previous work, Fast R-CNN employs a region of interest pooling scheme that allows to reuse the computations from the convolutional layers. In just 3 years, we've seen how the research community has progressed from Krizhevsky et. al's original result to R-CNN, and finally all the way to such powerful results as FASTER R-CNN [2].

Seen in isolation, results like FASTER R-CNN seem like incredible leaps of genius that would be unapproachable. Yet, through this post, I hope you've seen how such advancements are really the sum of intuitive, incremental improvements through years of hard work and collaboration. Each of the ideas proposed by R-CNN, Fast R-CNN[1,3], Faster R-CNN. We describe how we compose hardware, software and algorithmic components to derive efficient and optimized DNN models that are not only efficient, but can also be rapidly re-purposed for other tasks, such as object in motion identification, or assignment of transverse momentum to these motions. This work is an extension of the previous work to design a generalized hardware-software framework that simplifies the usage of deep learning techniques in big data problems.

1.1 CNN Fundamentals

Convolutional neural networks are an important class of learnable representations applicable, among others, to numerous computer vision problems. Deep CNNs, in particular, are composed of several layers of processing, each involving linear as well as non-linear operators, which are learned jointly, in an end-to-end manner, to solve a particular tasks. These methods are now the dominant approach for feature extraction from audiovisual and textual data.

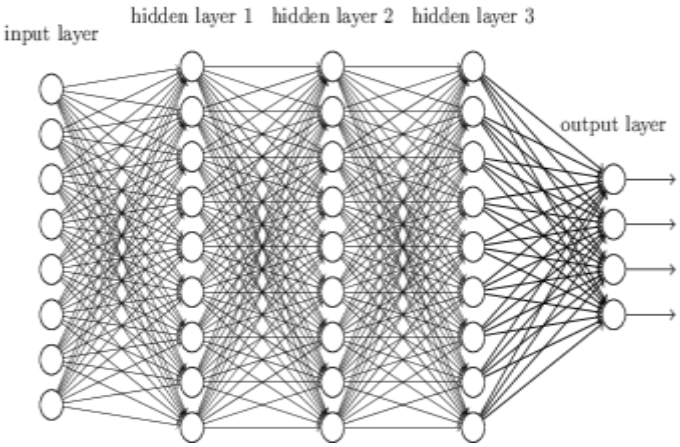

Figure 1: Diagram of common convolutional network

The first layer in a CNN is always a Convolutional Layer. First thing to make sure you remember is what the input to this conv (I'll be using that abbreviation a lot) layer is. Like we mentioned before, the input is a 32 x 32 x 3 array of pixel values. Now, the best way to explain a conv layer is to imagine a flashlight that is shining over the top left of the image. Let's say that the light this flashlight shines covers a 5 x 5 area. And now, let's imagine this flashlight sliding across all the areas of the input image. In machine learning terms, this flashlight is called a filter(or sometimes referred to as a neuron or a kernel) and the region that it is shining over is called the receptive field. Now this filter is also an array of numbers (the numbers are called weights or parameters). A very important note is that the depth of this filter has to be the same as the depth of the input (this makes sure that the math works out), so the dimensions of this filter are 5 x 5 x 3. Now, let's take the first position the filter is in for example. It would be the top left corner. As the filter is sliding, or convolving, around the input image, it is multiplying the values in the filter with the original pixel values of the image (aka computing element wise multiplications). These multiplications are all summed up (mathematically speaking, this would be 75 multiplications in total). So now you have a single number. Remember, this number is just representative of when the filter is at the top left of the image. Now, we repeat this process for every location on the input volume. (Next step would be moving the filter to the right by 1 unit, then right again by 1, and so on). Every unique location on the input volume produces a number. After sliding the filter over all the locations, you will find out that what you're left with is a 28 x 28 x 1 array of numbers, which we call an activation map or feature map. The reason you get a 28 x 28 array is that there are 784 different locations that a 5 x 5 filter can fit on a 32 x 32 input image. These 784 numbers are mapped to a 28 x 28 array.

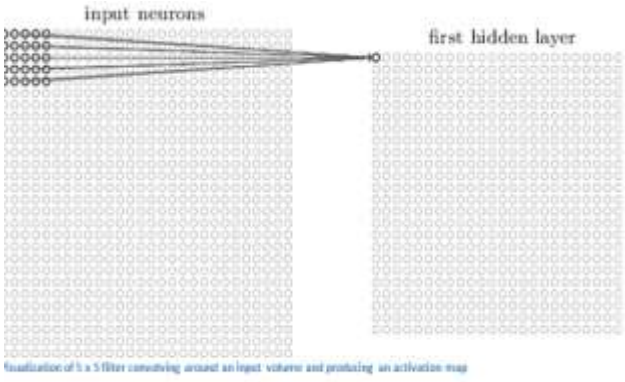

Figure 2: Basic CNN Model with hidden layer

1.2 RCNN Technique

As shown in Figure 1: the RCNN technique comprises of various steps as training set as mentioned below

Step -1 Use Selective search for region proposals

In selective search, we start with many tiny initial regions. We use a greedy algorithm to grow a region. First we locate two most similar regions and merge them together. Similarity S between region a and b is defined as:

S (a,b)=Stexture(a, b)+Ssize(a, b).

where Stexture(a ,b) measures the visual similarity, and SSize prefers merging smaller regions together to avoid a single region from gobbling up all others one by one. Regions are merged until everything is combined together.

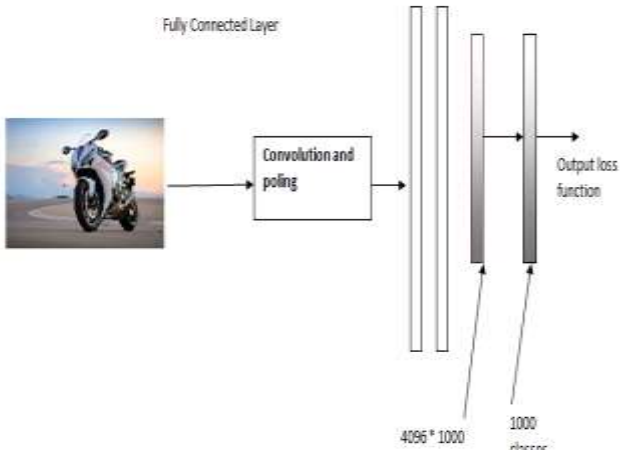

Figure 3: Step 1 for feature modelling

Step-2 Warping

For every region, CNN is used to extract the features. Since a CNN takes a fixed-size image, a region of size X x Y RGB images are taken.

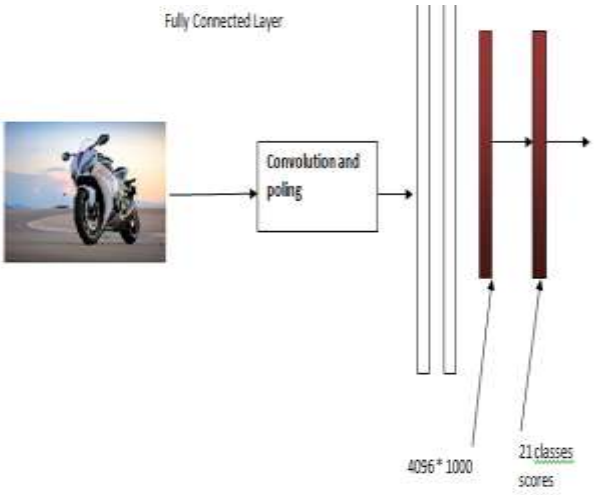

Figure 4: Step 2 for feature modelling and extraction

Step-3 Extracting features with a CNN

This will then process by a CNN to extract a XY-dimensional feature.

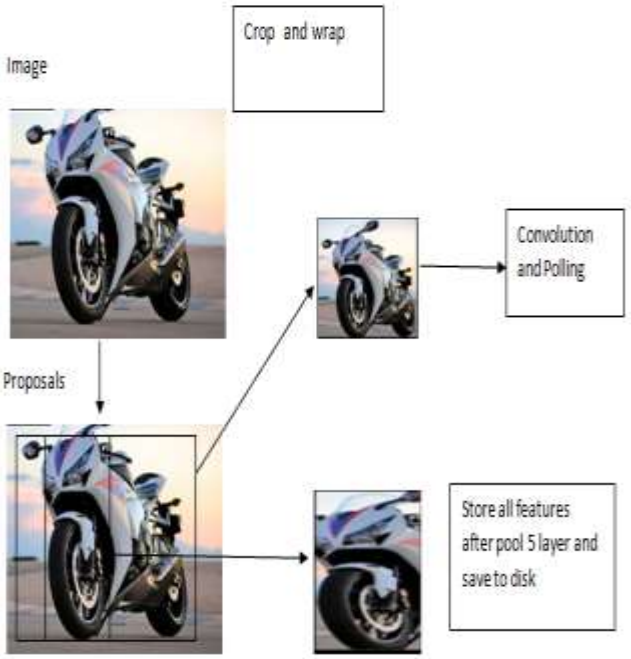

Figure 5: Step 3 for feature modelling training

Step 4 Classification

A SVM classifier is used to identify the object. In classification, there's generally an image with a single object as the focus and the task is to say what that image is. But when we look at the world around us, we carry out far more complex tasks. We see complicated sights with multiple overlapping objects, and different backgrounds and we not only classify these different objects but also identify their boundaries, differences, and relations to one another. We'll cover the intuition behind some of the main techniques used in object detection and segmentation and see how they've evolved from one implementation to the next. In particular, R-CNN (Regional CNN), the original application of CNNs to this problem, along with its descendants Fast R-CNN, and Faster R-CNN will be verified.

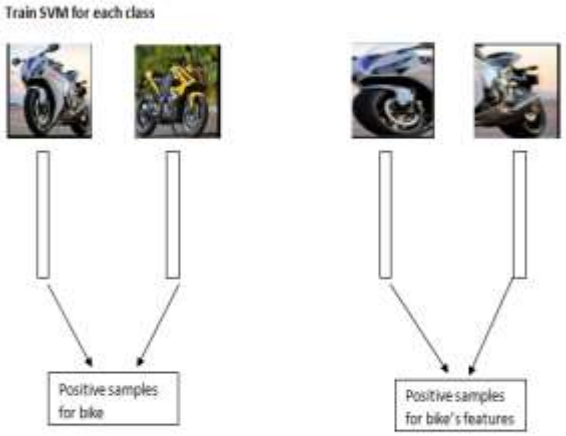

Figure 6: Diagram for feature classification stage

1.3 FAST RCNN

Fast R-CNN warps ROIs into one single layer using the RoI pooling. The RoI pooling layer uses max pooling to convert the features in a region of interest into a small feature map of H × W. Both H & W (e.g., 7 × 7) are tunable hyper-parameters. Instead of multiple layers, Fast R-CNN only uses one layer.

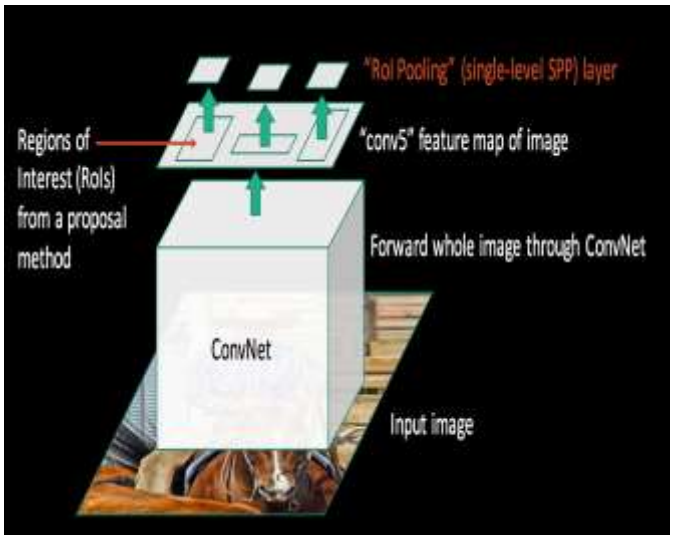

Figure 7: Diagram of FAST RCNN architecture

Fast R-CNN network takes as input an entire image and a set of object proposals. The network first processes the whole image with several convolutional and max pooling layers to produce a convolutional feature map. Then, for each object proposal a region of interest (RoI) pooling layer extracts a fixed-length feature vector from the feature map. Each feature vector is fed into a sequence of fully connected (fc) layers that finally branch into two sibling output layers:

one that produces softmax probability estimates over K object classes plus a catch-all "background" class and another layer that outputs four real-valued numbers for each of the K object classes. Each set of 4 values encodes refined bounding-box positions for one of the K classes.

1.3 FASTER RCNN

Fast R-CNN[1,3] warps ROIs into one single layer using the RoI pooling.. The region proposal network is a convolution network. The region proposal network uses the feature map of the "conv5" layer as input. It slides a 3x3 spatial windows over the features maps with depth K. For each sliding window, we output a vector with 256 features.

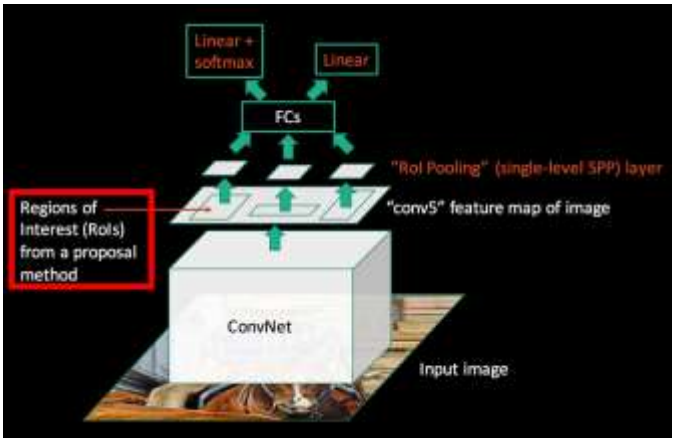

Figure 8: Diagram of FASTER RCNN architecture

3. Experiments and Results

The image set for bikes(as shown in figure given below) have been taken and trained using all RCNN (RCNN,FAST RCNN and FASTER RCNN).The images are trained and verified using caffe framework[4](c++,python) and in matlab also[5]. The following results as shown in table are compared. The best performance was obtained with the FASTER RCNN architecture and with ~80:00% accuracy. However, the difference with the system FASTER RCNN lies within the confidence interval. Therefore, the benefit of considering jointly the two modalities at the feature extraction stage need to be confirmed with additional experiments.

Table 1 Comparisons of RCNN, FAST RCNN, FASTER RCNN approaches reported for the CALTECH dataset (http://www.vision.caltech.edu/Image_Datasets/Caltech101/101_ObjectCategories.tar.gz .) Results are recognition accuracy in percent against classifier types (SVM,BAYESIAN, CNN)

Method	RCNN	FAST	FASTER
SVM	68.52 %	78.77 %	83.94 %
BAYESIAN	76.10 %	75.72 %	84.12 %
CNN-RNN	77.71 %	78.82 %	86.22 %

3. CONCLUSIONS

The use of CNN for extracting visual features from images of the dataset provided by various known classifiers techniques are experimented and their results have been shown. The comparison of various RCNN techniques has been conducted for which the two visual modalities are jointly processed. We derived different systems in which the CNN is used as a feature extractor and is combined with different classifiers techniques such as SVM and BAYESIAN. Experiments were conducted on a continuous image dataset the CNN over a previously published baseline. These techniques can be better implemented for challenging prediction tasks those can make better use the large learning capacity.

REFERENCES

[1] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks",IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 39, NO. 6, JUNE 2017

[2] Uçar, Yakup Demir,"Moving towards in object recognition with deep learning for autonomous driving applications",International Symposium on INnovations in Intelligent SysTems and Applications (INISTA),2016

[3] Jifeng Dai,Yi Li,Kaiming He,Jian Sun,"R-FCN: Object Detection via Region-based Fully Convolutional Networks",30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain

[4] Online resource for Caffe: https://github.com/rbgirshick/py-faster-rcnn

[5] https://in.mathworks.com/help/vision/examples/object-detection-using-deep-learning.html?requestedDomain=www.mathworks.com

[6] Ms. Vijayasanthi D., Mrs. Geetha S.,"Deep Learning Approach Model For Vehicle Classification Using Artificial Neural Network",International Research Journal of Engineering and Technology (IRJET) Volume: 04 Issue: 06 | June -2017

[7] Ross Girshick Microsoft Research,"Fast R-CNN",ICCV,2015

[8] Eric Tatulli, Thomas Hueber , "FEATURE EXTRACTION USING MULTIMODAL CONVOLUTIONAL NEURAL NETWORKS FOR VISUAL SPEECH RECOGNITION",ICASSP 2017

[9] Sandeep Kaur, Gaganpreet Kaur,"Content Based Image Retrieval and Classification Using Image Features and Deep Neural Network",International Research Journal of Engineering and Technology (IRJET),Volume: 03 Issue: 09 ,Sep -2016

[10] Cosmina Popescu1, Lucian Mircea Sasu, "Feature Extraction, Feature Selection and Machine Learning for Image Classification: A Case Study",pattern recognition and machine intelligence,2014

[11] Ross Girshick,"Fast R-CNNObject detection with Caffe",Microsoft Research