***Image Classification Using Convolutional Neural Network (CNN)***

**Ferdousi Rahman1, Israt Jahan Ritun1, Nafisa Farhin1**

*1Students of B.Sc. in Computer Science and Engineering,Department of Computer Science & Engineering.*

*BRAC University, 66 Mohakhali, Dhaka, Bangladesh.*

*{ferdousirahman707, isratritun, nafisafarhin }@gmail.com*

***Abstract —*** ***Most modern convolutional neural networks (CNNs) used for object recognition are built using the same principles: Alternating convolution and max-pooling layers followed by a small number of fully connected layers. We re-evaluate the state of the art for object recognition from small images with convolutional networks, questioning the necessity of different components in the pipeline. We find that max-pooling can simply be replaced by a convolutional layer with increased stride without loss in accuracy on several image recognition benchmarks. Following this finding – and building on other recent work for finding simple network structures – we propose a new architecture that consists solely of convolutional layers and yields competitive or state of the art performance on several object recognition datasets (CIFAR-10, CIFAR-100, ImageNet). To analyze the network we introduce a new variant of the “Deconvolution Approach” for visualizing features learned by CNNs, which can be applied to a broader range of network structures than existing approaches.***

***Keywords—Convolutional Neural Network, CIFAR-10, Deep Learning, Image Classification***

1. INTRODUCTION

The goal of this project is to classify images to a respective category that it belongs which a major branch of image processing. Previously many diverse approaches were taken for handling images and extract data or result information for different usage. Classification of pictures has been an important task in many fields as well as a challenging factor for robust systems. Machine learning has been a essential footstep for this sector [1]. But training the system with specific personal dataset gives a partial success. Our proposed project uses a dataset call CIFAR-10. CIFAR-10 is a multi-class dataset consisting of 60,000 32×32colour images in 10 classes, with 6,000 images per class There are 50,000 training images and 10,000 test images .In this paper, we explore different learning classifiers for the image-based multi-class problem.CIFAR-10 presents a challenging classification problem.32×32 images don’t contain enough information for most classifiers to draw clear decision boundaries. A clear example of this is the confusion between “cat” and “dog” classes. The images objects are different in scale, rotation, position, and background. Some of the images are very unclear and hard to classify (even for human beings).

1. LITERATURE REVIEW

Neural networks are widely used in solving image recognition problems. There is a wide variety of architectures that serve different purposes. Some publications aim at simple architectures to achieve decent results. [2] Presents a shallow neural network that is, unlike multi-layered (i.e. deep) architectures, fast to train and more suitable for real-time applications. Their network achieved 75.86% test accuracy. To learn better feature representations and to transform the image space into a linearly separable feature space, a standard linear classification algorithm (e.g. SVM) is used. It aims at learning features by training a convolutional neural network using only unlabeled data. Another experiment with network pooling suggests regulating existing pooling functions. It proposes a flexible parameterization of the spatial pooling step and learn the pooling regions together with the classifier that replaces the conventional deterministic pooling operations with a stochastic procedure (randomly picking the activation within each pooling region) has formulated a fractional (stochastic) version of maxpooling (where non-integer multiplicative factors are allowed). This helps reduce overfitting, and achieves state-of-the-art accuracy (with 96.53% test accuracy). The system learns new pooling functions by combination of max and average pooling functions, or tree-structured fusion of pooling filters.

Optimize network activation functions present new randomized leaky rectified linear units (RReLU). [4] designed a novel form of piecewise linear activation function that is learned independently for each neuron using gradient descent (and outperforms rectified linear units. introduce an exponential linear unit (ELU) which speeds up learning in deep neural networks and leads to higher classification accuracies defines a simple new model called maxout to improve the accuracy of dropout fast approximate model averaging technique. present a probabilistic variant (probout) of the recently introduced maxout unit to improve its invariance properties. shows that replacing the softmax layer with a linear support vector machine (SVM) consistently improves accuracies proposes a deep neural network architecture for object recognition based on recurrent neural networks (ReNet). The proposed network replaces the ubiquitous convolution + pooling layer of the deep convolutional neural network with four re-current neural networks that sweep horizontally and vertically in both directions across the image. [4] Proposes a recurrent CNN (RCNN) for object recognition by incorporating recur-rent connections into each convolutional layer to enhance the ability of the model to integrate the context information (which is important for object recognition).This introduced a novel architecture that decreases depth and increases width of residual networks. Wide residual networks (WRNs) are superior to their deep residuals counter parts.it also proposes a simple method for weight initialization for deep net learning, called Layer-sequential unit-variance (LSUV) initialization to improve test accuracies.

1. PROPOSED METHOD

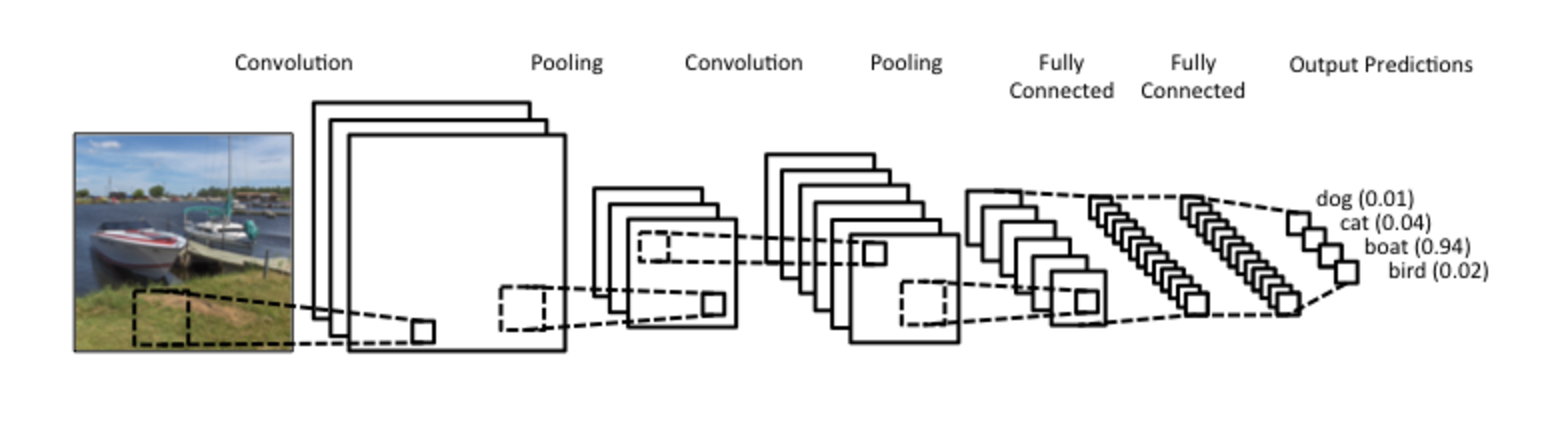
* CONVOLUTINAL NEURAL NETWORK (CNN):

We used the best available model Models from other publications with higher accuracies were either not publicly available or not training properly. The architecture of our deep neural network consists of 8 convolutional layers in addition to 3 linear layers .It achieves, on average, a test accuracy of 73.13%. Ensemble Weight Estimation proposes multiple approaches for ensemble parameter estimation in a weighted voting system. However, we opted out for a simple exhaustive search. Assume,C1, C2, ..., Cn, are experts. We define a sequence of possible weights,Wn+1=Wn + S and W0= 0, where S is a step between two consecutive weights. Let R={W0, W1, ..., Wk},where k is the number of possible weights.

E(C1, C2) = argmax wi ,wj ∈ Rwi ×C1+wj×C2

We estimate all parameters in a chain rule style (i.e. E(E(C1, C2), C3)).

Convolutional networks almost always incorporate some form of spatial pooling, and very often it is alpha times alpha max-pooling with alpha=2. Max-pooling act on the hidden layers of the network, reducing their size by an integer multiplicative factor alpha. The



*Fig.: Convolutional Neural Networking*

amazing by-product of discarding 75% of your data is that you build into the network a degree of invariance with respect to translations and elastic distortions. However, if you simply alternate convolutional layers with max-pooling layers, performance is limited due to the rapid reduction in spatial size, and the disjoint nature of the pooling regions. We have formulated a fractional version of max-pooling where alpha is allowed to take non-integer values. Our version of max-pooling is stochastic as there are lots of different ways of constructing suitable pooling regions. We find that our form of fractional max-pooling reduces overfitting on a variety of datasets: for instance, we improve on the state-of-the art for CIFAR-100 without even using dropout.

* LAYER-SEQUENTIAL UNIT-VARIENCE (LSUV) INITIALIZATION:

A simple method for weight initialization for deep net learning - is proposed. The method consists of the two steps. First, pre-initialize weights of each convolution or inner-product layer with orthonormal matrices. Second, proceed from the first to the final layer, normalizing the variance of the output of each layer to be equal to one. Experiment with different activation functions (maxout, ReLU-family, tanh) show that the proposed initialization leads to learning of very deep nets that-

1. produces networks with test accuracy better or equal to standard methods and
2. is at least as fast as the complex schemes proposed specifically for very deep nets such as FitNets.

* DROPOUT:

Dropout (Hinton et al., 2012) provides an inexpensive and simple means of both training a large ensemble of models that share parameters and approximately aver-aging together these models’ predictions. Dropout applied to multilayer perceptrons and deep convolutional networks has improved the state of the art on tasks ranging from audio classification to very large scale object recognition (Hinton et al., 2012; Krizhevsky et al., 2012). While dropout is known to work well in practice, it has not previously been demonstrated to actually perform model averaging for deep architectures.

* MODEL:

The Sequential model is a linear stack of layers. It converts the input into a linear stack. In our project we have used Sequential model. As we know neural network takes one input and for that one input only one specific output is there. This singularity is maintained by sequential model. We have created this Sequential model by passing a list of layer instances to the constructor. We have used 2D convolution layer like Conv2D(nb-filters, (nb-conv, nb- conv), input\_shape=(nb-conv, img-row, img-col), padding='same', activation='relu',kernel\_constraint=maxnorm(3). This layer creates a convolution kernel that is convolved with the layer input to produce a tensor of outputs. Conv2D is for the first convolutional layer. The first convolutional layer will shape the image into filter (32x32) and a Dot product is done with filter and input pixel. During using this layer as the first layer in a model, we have provided some keyword arguments such that input\_shap(), activation='relu',kernel\_constraint=maxnorm(3) etc. Here we have used input\_shape=(3, 32, 32) for 32x32 RGB pictures.

There are many activation functions. Sigmoid, Tanh and Relu are some basic activation functions. To get better result we have used the relu activation function. Relu looks like this

f(x)= max(0,x)

The range of this activation function is (0, inf), and it’s not differentiable at zero. The nicest thing about relu is that it’s gradient is always equal to 1, this way we can pass the maximum amount of the error though the network during back-propagation.

Moreover we have used kernel\_constraint =maxnorm(3). Here 3 is the highest weight of the neuron so weight cannot be more than 3. It is a constraint. This constraint regularizes directly. Here we have also used Dropout(0.2) for creating our model. Dropout is a simple and powerful regularization technique for neural network. Dropout can be used in input layer, hidden layer and also in output layer. Dropout is a technique where randomly selected neurons are ignored during training.

After thatAI we have also used MaxPooling2D(pool\_size = (2, 2). MaxPooling reduces the featured map by choosing the highest value. For example:

0 2 1 4

6 9 6 7

From these featured maps it will choose 9 and 7 and, in this way, it will reduce the map.Also we have used Flatten() because we need to convert four dimension to two dimension. It is used to reshape. Moreover, we have used activation='softmax'. Here at the output, the number can be too big or too small. So, we have used softmax classifier to fix the numbers from 0 to 1.

[1] https://towardsdatascience.com/machine-learning-with-ibm-powerai-getting-started-with-image-classification-part-1-6219e3c6a9fa

[2] R. Suguna, P. Anandhakurnar, "Concept Discoveryin Image Databases using Orthogonal Polynomial Transformation", *Advance Computing Conference 2009. IACC 2009. IEEE International*, pp. 1382-1387, 2009.