Performance Analysis of Deep Neural Networks Using Computer Vision

Nidhi Sindhwani[1], Rohit Anand[2], Meivel S.[3], Rati Shukla[4], Mahendra Pratap Yadav[5], Vikash Yadav[6,*]

[1]Amity University, Noida, India
[2]G.B.Pant DSEU Okhla – 1 Campus, New Delhi, India
[3]M.Kumarasamy College of Engineering, Karur, Tamil Nadu, India
[4]MNNIT Prayagraj, Allahabad, Uttar Pradesh, India
[5]VIT Bhopal University, Madhya Pradesh, India
[6]Department of Technical Education, Uttar Pradesh, India

Abstract

INTRODUCTION: In recent years, deep learning techniques have been made to outperform the earlier state-of-the-art machine learning techniques in many areas, with one of the most notable cases being computer vision. Deep learning is also employed to train the neural networks with the images and to perform the various tasks such as classification and segmentation using several different models. The size and depth of current deep learning models have increased to solve certain tasks as these models provide better accuracy. As pre-trained weights may be used for further training and prevent costly computing, transfer learning is therefore of vital importance. A brief account is given of their history, structure, benefits, and drawbacks, followed by a description of their applications in the different tasks of computer vision, such as object detection, face recognition etc.

OBJECTIVE:. The purpose of this paper is to train a deep neural network to properly classify the images that it has never seen before, define techniques to enhance the efficiency of deep learning and deploy deep neural networks in various applications.

METHOD: The proposed approach represents that after the reading of images, 256x256 pixel image's random parts are extracted and noise, distortion, flip, or rotation transforms are applied. Multiple convolution and pooling steps are applied by controlling the stride lengths.

RESULT: Data analysis and research findings showed that DNN models have been implemented in three main configurations of deep learning: CNTK, MXNet and TensorFlow. The proposed work outperforms the previous techniques in predicting the dependent variables, learning rate, image count, image mean, performance analysis of loss rate and learning rate during training, performance Analysis of Loss with respect to Epoch for Training, Validation and Accuracy.

CONCLUSION: This research encompasses a large variety of computer applications, from image recognition and machine translation to enhanced learning. DNN models have been implemented in three main configurations of deep learning: CNTK, MXNet and TensorFlow. Extensive research has been conducted using the various deep architectures such as AlexNet, InceptionNet, etc. To the best of authors' knowledge, this is the first work that presents a quantitative analysis of the deep architectures mentioned above.

Keywords: Computer Vision, Object Detection, Deep Learning, Deep Neural Network, Image classification, Artificial intelligence, Machine learning, Convolutional neural network, Accuracy

Received on 26 July 2021, accepted on 27 September 2021, published on 13 October 2021

doi: 10.4108/eai.13-10-2021.171318

*Corresponding author. Email: vikas.yadav.cs@gmail.com

1. Introduction

Deep Convolutional Neural Networks (CNNs) [1,2] are feed-on networks, sparsely connected to the neural networks. The regular structure of CNNs is made up of a line of layers. Each layer inputs a data set known as the Function Map (FM) and generates a new set of FMs with higher semantics [3]. Like regular machine learning algorithms, CNNs have been implemented in two phases. First of all, the stage of training is focused on a known list of annotated subjects. When a model is fine-tuned, the weights of a previously trained network are used to initialise the constraints of a new training. These weights are then modified for a new restriction, such as a different data set or a decreased accuracy. The second step, known as inference, uses the model learnt to identify the new data samples (i.e., inputs that were not previously seen by the model). In a typical system, CNNs are trained/fine-tuned only once to the large clusters of Graphics Processing Units (GPUs). On the other hand, an assumption is made every time a new data sample is to be categorised. As a result, the research lights more on accelerating the inference process. Our discussion outlines the key approaches used to accelerate the inference [4-6]. In addition, as most of the CNN accelerators test their performance against the models trained for image classification, this paper focuses on such applications. However, the methods detailed in this survey can be used to enhance the use of CNNs for other applications such as object detection, image segmentation and speech recognition etc. [7-9].
Machine learning works by creating a network of simple computing nodes that perform "output = F (weight*input + bias)" calculation and find the optimal weights and biases using known data to recognise the trends in inputs that correlate with outputs. The model is trained on tagged (i.e. training) data sets. The trained model can be used to predict the output for untagged (i.e. inferential) input data sets. Deep neural networks (DNNs) can be represented as tissues of nodes [10], where nodes represent the computational operations on multi-dimensional data arrays [11-13]. Deep learning, a branch of machine learning inspired by how the human brain operates, takes place in two stages: Training and Inference. The manual definition of these characteristics or programming the parameters of a good or bad product are not required. The neural network trains by itself. In the inference phase, the trained neural network after being presented with fresh images provides an inference as to the quality of the object and the neural network's confidence in its assessment [14]. A popular computational intelligence numerical scheme for solving second order nonlinear singular functional differential equations (FDEs) is presented, which employs artificial neural networks (ANNs) [15-17].
The machine learning may be modelled in classic mode or deep learning mode as shown in figure 1.

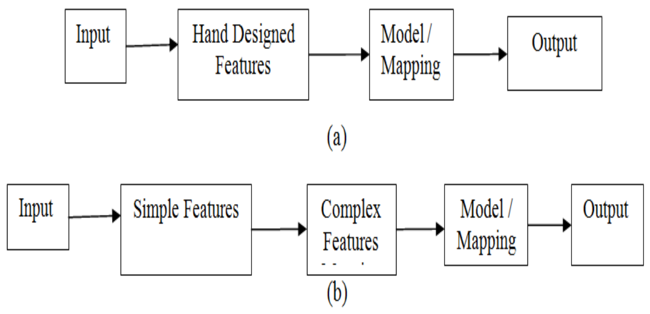


Figure 1. (a) Classic Machine Learning Model (b) Deep (End-to-End) Machine Learning Model

The training process includes determining the number of neurons and layers (that would make up the neural network) and exposing the network to the labelled training data, usually good images of objects (that would pass the inspection) and poor images of objects (that would fail). The neural network then sets out the properties of each grade, such as scale, shape, colour, colour consistency and so on [18]. The recent popularity of deep neural networks (DNNs) has generated significant interest in the efficient performance of DNN-related computing. However, the main emphasis is typically very specific and restricted to (i) inference i.e. how to effectively execute already trained models and (ii) image classification networks as the primary benchmark for assessment. The benchmark suite for DNN training is called Training Benchmark for DNNs, comprising of a representative set of eight DNN models and covering six major machine learning applications: image classification, machine translation, speech recognition, object detection, adversarial networks, reinforcement learning and detailed performance review of these models across three major deep learning frameworks (TensorFlow, MXNet & Cognitive Toolkit) across the various hardware configurations (single-GPU, multi-GPU, and multi-machine) [19]. Most of the machine learning codes is written in Python, invoking compiled system libraries and hardware-specific math libraries like Cuda & MKL (Math Kernel Library) etc. As python is an interpreted language, dynamic tracing is required for most of the analysis. In some instances, there are native python tracers, tracers built into frameworks, and hardware-based tools [20, 21].
Rest of the article is systematized as follows. Section 2 includes a brief survey on the related work. Section 3 discusses the various types of Networks for training and deployment. Section 4 provides the different steps involved in training and deployment of a Neural Network and section 5 discusses the performance assessment and results. At the end, the work is summarized in Section 6.

2. Related Work

The authors in [22] reviewed three major groups of computer vision deep learning. The three primary categories of computer vision deep learning are Convolutional Neural

Networks, the Boltzmann family and Stacked Denoising Auto-encoders (SdAs). All of these groups have been used to achieve the substantial output rates in numerous kinds of visual comprehensions. CNNs have a special ability to learn the features, that is, they automatically comprehend the features on the basis of data collection. They are also stable for the transformation which is very beneficial for many of the applications related to computer vision. However, they depend excessively on the presence of labelled data, in contrast to Deep Boltzmann Machines (DBMs) and Deep Belief Networks (DBNs) (both DBMs and DBNs come under Boltzmann family) and SdAs, which is able to run in an unsupervised manner. Out of all the models explored, both DBNs/DBMs and CNNs are very challenging from the computational point of view in the training but SdAs might be trained under the various situations in real-time. In [23], the authors explored the efficiency of the state-of-the-art deep learning systems, like Amazon Web Services (AWS) P3, NVIDIA DGX-2, IBM Power System Accelerated Computation Server AC922, and TensorEX GPU, a consumer-grade server. The research focuses on workloads in deep learning from the natural language processing and the computer vision. The performance review is carried out in association with a variety of essential features. The efficiency of communication and large throughput models of machine learning are considered. Many future usage designs for the various systems as a singular and as a part of the cloud are also investigated. The impact of different types of machine learning optimization models and machine layouts are also included in the study. In [24], the authors discussed the analysis of configuration for the object detection in case of deep learning. This analysis starts with a concise prologue to the background of deep learning and its illustrative method, CNN. Subsequently, the standard universal frameworks for the object detection have been discussed along with a few alterations and gainful tricks to further enhance the efficiency of detection. As the various particular detection exercises reveal the distinct features, a variety of peculiar tasks, including salient object detection and facial detection have been surveyed. The real-time test analyses are also performed to differentiate the different approaches and to draw solid cessations. At last, a range of encouraging activities is presented to guide the potential work in object detection and training systems of neural networks. In [25], the authors carried out a large number of demonstrations on cell phones (NVIDIA TX2) to explore the efficiency of numerous models related to deep learning. Some standard conventional tools are introduced to find the behaviour of machine learning models on the cell phones in view of memory and hardware usage and power injected. The various tools can compare the behaviour with compact operations in machine learning models, offering the potential to apprehend the performance variation and fine-grained problems. Perceiving the behaviour and features is crucial to the design and implementation of machine learning models on the cellphones.

The authors in [26] provided an overview of the background and ultra-modern techniques to text, audio and video processing; social network survey and natural language processing, along with an in-depth investigation of ground-breaking developments in the various applications of machine learning. The research also explored the problems facing online learning and unsupervised learning and further demonstrated how the various provocations could be turned into fertile potential avenues for study. In [27], the authors reviewed the most recent advances in the use of computer vision, image recognition and deep neural networks for the study and analysis of particle collision events in the Large Hadron Collider (LHC). The relation between LHC data analysis and computer vision techniques is focused on the principle of jet-images. It has been demonstrated that the modern image classification methods based on neural network frameworks of deep learning greatly boost the recognition of strongly stimulated electroweak particles with respect to the present methods. In addition, some novel techniques are implemented for visualising and understanding the high-level features that deep neural networks have learned to differentiate beyond physically derived variables, incorporating a new ability to understand physics, and creating more efficient LHC classification methods. The authors in [28] solved the problem of the self-regulating categorization of the various agricultural products, with no requirement for any human involvement. A comparative analysis of the well-known methods is used in the classification of images using Machine Learning. The proposed technique led to have the excellent results, as seen in the observational field. Further, the authors in [29] analysed the behavioural features of GPU for five different common machine learning frameworks: Caffe, Theano, TensorFlow, CNTK, and Torch from the frame of mind of the indicative model of CNN called AlexNet. Based on the attributes acquired, some methods of optimization have been suggested to improve the CNN model developed by the supporting configuration. The GPU output characteristics of the various convolution techniques have also been displayed, each of which uses General Matrix Multiply (GEMM), Fast Fourier Transform (FFT) and direct convolution. The scalability of CNN models have also been examined with the machine learning configurations in the context of multi-GPU as well as their overhead. The results show that the training rate of the AlexNet model can be increased two times by simply adjusting the alternatives given by the configurations.

In [30], the authors presented an integrated visual analytics framework to provide a wide variety of the performance assessment methods for different machine learning models in case of image classification problems. The proposed framework aims to address the problems by offering visual performance analysis at various levels and visualising instances of misclassification. The method, consisting of the views like ranking, projection, matrix and instance list views, enables the comparison and analysis of different models through the user interaction. Many cases of the use of multiple machine learning models have been identified in the Modified National Institute of Standards and Technology (MNIST) dataset to demonstrate the effectiveness of the method proposed. The authors in [31] discussed a drastic transformation of the computer systems from basic data processing to deep learning due to the reachability of large amount of data acquired from the internet and the sensors. The study found that deep learning techniques in computer vision are supervised, unsupervised, and semi-supervised. Commonly used algorithms are based on k-means clustering and neural networks. The latest applications of deep learning in the field of computer vision are the identification of objects, the recognition of objects and the derivation of the concerned details from pictures, visuals and graphical records. In [32], the authors encouraged a debate as to whether awareness of the conventional techniques of computer vision might be retained or not. They also discussed how the different sides of the computer vision can be united. Various kinds of latest techniques based on the combination of the various methodologies have been tested. All of these have shown the potential to enhance the computer vision efficiency and fix issues not matched with the machine learning. For instance, merging conventional techniques of computer vision with machine learning has been quite common in the various arising domains such as three-dimensional vision, for which the machine learning models have not been fully optimised and tested. In [33], a deep insight has been provided into the analysis of the objective detection approach using Convolutional Neural Networks that is applicable in all the three approaches (salient, objectivity, and category-specific) of object detection. Deep learning configurations and principles that are common for object detection tasks are also checked. In [34], an in-depth analysis has been done based on the latest advancements in deep neural networks. The four types of deep learning architectures, namely the small Boltzmann system, the deep faith networks, the auto-encoder and the convolutional neural network, are discussed in depth. Based on the deep learning methods, the unsupervised learning algorithms may be used to process the unmarked data. In addition, the trade-off between precision and computational complexity can also be adjusted flexibly in the most sophisticated learning algorithms. With the rapid growth of hardware resources and computational technologies, everyone is quite optimistic that deep neural networks will gain the wider attention and find the broader applications in the future.

3. Types of Networks for Training and Deployment

Graphic processing units (GPUs), Digital Signal Processors or more silicone frameworks designed for high efficiency and small energy are often met by significant computational resources requirements for training and assessment of Convolutional Neural Networks when the idiosyncratic patterns of computation are performed. In reality, the modern processors like Tensilica Vision P5 Digital Signal Processor from Cadence have a practically supreme collection of computing as well as memory resources needed to run neural networks with high quality performance.

Many studies with image recognition have proved that the conventional algorithms based on single level (also called shallow algorithms) are less effective than those based on multi-level (also called deep algorithms) where the various types of filter operations are performed at each level, and the outcomes of the former levels can be used at each of the deeper levels; the final outcome can be obtained by integrating the unique outcomes of all the individual levels. The explanation for improving the behavior of deep algorithms is that the different filters explicitly customized to each level can be applied. Some common examples are multi-resolution filters that can reveal the features of image on a variety of scales. When the different kinds of satellite images in urban settlements are considered, the commercial district typically has more high-rise buildings & wider streets than the residential districts. Several forms of networks have arisen that prove their strongness for the various images to be interpreted semantically. The different forms of networks are listed below that have proved to be quite useful for the analysis of satellite images.

Deep Neural Networks (DNNs): As defined in [35], these networks are made up of many layers and consist of an input layer, at least one hidden layer and an output layer. The processing of pixels is performed by each layer by itself. The subsequent process of training may be referred to as deep learning.

Recursive Neural Networks (RNNs): They should not be addled with the recurring neural networks. When the input data has been organized, these networks that are also used for the processing of speech and comprehension can be used effectively. Further, these networks may also be utilized for the genuine scenes, such as the images having the recursive layout [5]. Thus, RNNs can be used for segmentation and annotation of semantic scenes.

Convolutional Neural Networks (CNNs): They are developed for the image classification (with smaller error) of large images with a sizeable number of groups. As mentioned in [3], more than one million images can be categorised and more than 1,000 different classes are allocated. This is achieved centrally by five convolutional layers, three entirely interconnected layers, and the innumerable number of internal parameters. In order to minimise the overfitting, the system performs regularisation by avoiding the offending elements.

Generative Adversarial Networks (GANs): These networks enable the reciprocal training of two rival models G and D based on multilayer Perceptron following an oppositional process: G decides the dissemination of data, whereas D calculates the likelihood that a sample will be obtained from the training data rather than D. Further, D depicts the input data of multi-dimensions for semantic category labels.

3.1 Popular Types of DNNs

DNN is a fully-connected and feed-forward neural network based on Multilayer Perceptron (MLP) model while CNN is a feed-forward sparsely connected weight sharing neural network and RNN is a feedback based neural network based on long-term memory and storage.
In the sense of visualisation, there are six different categories of network frameworks. Certain frameworks are more unique based on either the learning algorithm or the kind of connection between the layers or the type of layers. Deep neural networks basically come under the multi-mode feed-forward neural networks.
DNN may be categorized into following types:

(i) **Deep Convolution Neural Network (DCNN)**: It is a network with an exceptional framework having eight layers [36, 37]. The first five layers are convolutional layers and the last three layers are entirely interconnected. Deep CNNs are based on supervised learning mechanisms, so large amounts of annotated data are required for proper learning.

(ii) Deep Belief Network (DBN): It is made up of the Restricted Boltzmann Machines (RBMs) and are defined by a special training technique [11].The top two layers have undirected connections, while the other layers are directly connected to the upper layers. RBM architecture consists of a bidirectional connection between hidden and visible layers. This feature enables the weight to be connected exclusively, allowing for deeper extraction between neurons.

(iii) Convolutional Deep Belief Network (CDBN): It is just like DBN, comprising of Convolutional RBMs stacked together [38]. Training is done similar to DBN using a greedy layer-wise learning process, i.e. the weights of the learned layers are set and regarded to be the feedback for the next layer. The CDBN model uses a hierarchical structure to extract the high-level features from low-level ones: lower layers extract low-level features and feed them into higher layers. The CDBNs are hierarchical generative models made up of stacked Convolutional Restricted Boltzmann Machines (CRBMs).

(iv) Multi-Column Deep Neural Networks (MCDNN): It **is** essentially a mixture of many DNNs stacked in the form of column [29]. The input is processed by all deep neural networks and their output is combined to the final output of the deep neural network. RELU is a Rectification Layer special implementation.

4. Steps Involved in Training and Deployment of a Neural Network

The proposed approach represents that after the reading of images, 256x256 pixel image's random parts are extracted and noise, distortion, flip, or rotation transforms are applied. Multiple convolution and pooling steps are applied by controlling the stride lengths. Pooling entails applying a mask to each pixel and then selecting a single value (for example, maximum) from the mask. A gradient optimizer is used to train the model weights by feeding the final output into a SoftMax function. The model is able to automatically learn the decision boundaries required to classify the images into one of the two classes by introducing non-linearities during learning using a Rectified Linear Unit (ReLU) and random dropouts. As the data flow through the network, the size of the filters and the outputs changes, allowing for training and detection of the similar features with different scaling. Because the convolution filters are applied to the entire image for training data, the CNNs are insensitive to feature transformations such as rotation and translation. Furthermore, the pooling layers effectively make the CNNs resistant to the various feature distortions.
The different steps for the training and deployment of a neural network are mentioned in the figure 2.

Step 1: Identify the appropriate deep learning function

An activation function's purpose is to add some kind of non-linear property to the function. The neural network could only perform the linear mappings from inputs x to output y if the activation functions were not present. We only use Softmax in the final layer when we want the neural network to predict the probability scores during the classification tasks. Simply put, the Softmax activation function forces the values of output neurons to be between zero and one, allowing them to represent the probability scores. Another thing to keep in mind is that when we classify the input features into different classes, these classes are mutually exclusive. This means that each feature vector x is assigned to a single class. The four most important deep learning tasks include classification, identification and localization, segmentation and detection of anomalies. Classification requires sorting the images into various types of images. Grouping of images is based on the different properties, most often in Pass and Fail categories [39].

Step 2: Select a framework

A structure, or tool set used to build a neural network, typically includes a neural network starter and network training and testing tools. Free, easy-to-use frameworks such as PyTorch, TensorFlow, and Caffe2 provide great documentation and examples to allow the inexperienced users to train and deploy the neural networks with minimal effort. It provides flexible development and deployment and also facilitates the mobile deployment. However, it has a higher learning curve compared to PyTorch. One of the oldest frameworks, Caffe2 has widely supported libraries for the convolutional neural networks and computer vision applications and is better adapted for the mobile devices using OpenCV. The optimal structure for a task ultimately depends on the complexity and speed of the inference needed. The more layers the neural network has, the slower the inference is [40].

Step 3: Preparing training data for the neural network

The number of images needed for the training depends on the type of data that the neural network can analyze.

Generally, a collection of training images is needed to test all the characteristics and grades of each characteristic of the neural network. More the number of images presented for each category, more finely the neural network learns to evaluate these categories. Companies such as Cvedia construct the synthetic datasets that are annotated and optimised for the neural network training. In the absence of other choices, self-produced and labelled images need to be made. Switching a single image to multiple images by rotating, resizing, stretching, and brightening or darkening saves time. Several developers in the deep learning industry open source their image labelling solutions and distribute them free of charge. Labelling, which is especially useful for unmarked datasets, offers a graphical image annotation tool that helps label the artefacts in the bounded boxes inside photos. Alternatively, third parties can be interested in the labelling process. Preparing the training data can become even more important in the light of particular hardware limitations or preferences, as some deep learning tools support only a limited range of hardware [41].

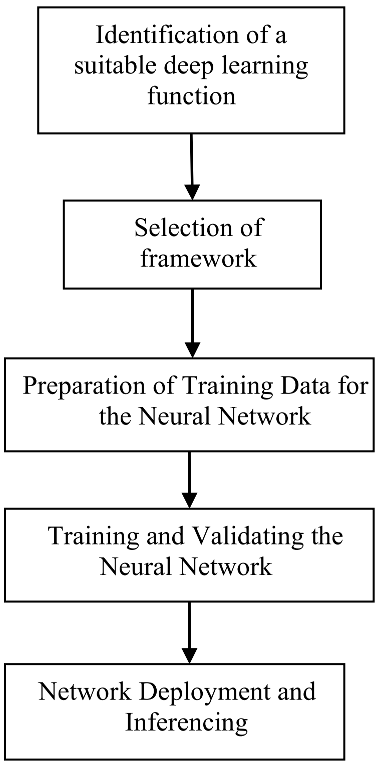

Identification of a suitable deep learning function
Selection of framework
Preparation of Training Data for the Neural Network
Training and Validating the Neural Network
Network Deployment and Inferencing

Figure 2. Steps in Training and Deployment of a Neural Network

Step 4: Train and validate the neural network to ensure accuracy

This stage involves configuring and running the scripts on a device until the training process offers an appropriate degree of accuracy for a particular use. Separating training and test data means that the neural network does not mistakenly train the data later used for evaluation. By taking advantage of the transfer of learning or the use of a pre-trained network and re-use it for another mission, this process can be accelerated. For example, a neural network already trained for feature extraction can only need a new collection of images to recognise a new feature. Frameworks like Caffe2 and TensorFlow provide open, pre-trained networks [42]. Neural networks learn the features directly from the data from which they are trained, so that the specialists do not need to bring out the features manually. There is a range of graphical user interface-based software options for the neural network training, such as Matrox Imaging Library (MIL) X, which works with the different frameworks and hence make the training and deployment process very intuitive even for less time [36].

Step 5: Deploy the neural network and run inference on new data

This last step includes deploying a qualified neural network on the selected hardware to assess the output and collect the field data. The first few phases of inference, preferably used in the field to gather the additional test data, can provide the training data for the future iterations. Cloud deployment provides the substantial savings on hardware costs and the opportunity to easily scale and install and distribute the changes at different locations. However, internet connectivity problems can cause the critical errors, and cloud deployment has a high latency compared to the edge deployment. In order to successfully deploy a trained model, there are two initial jobs.
1) First job is to provide our model an input that it expects.

2) Second job is to provide our end user an output that is useful.

The input that the network expects is determined both by its architecture and by how it was trained. The deployment requires writing code to convert the input we have to the input the network expects. The output that our network generates is determined by its architecture and what it learned. Deployment also requires writing code to convert the output that is generated to the output our end user expects.

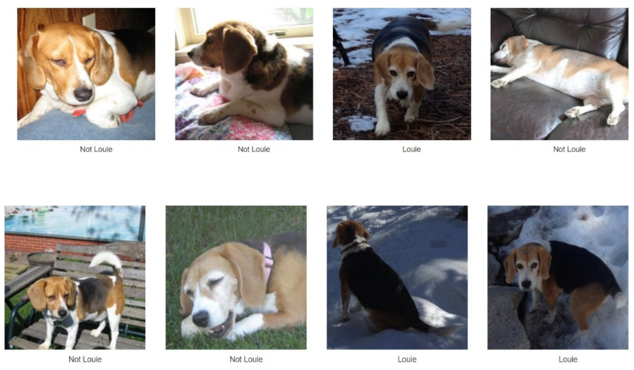
5. Performance Analysis and Results

A dataset is a set of related examples used to train and test a model. A dataset is a collection of examples related to a specific topic or domain, and it generally aims to serve one or more applications. Because a dataset can be labelled, it is ideal for training and testing the supervised models. Unlabelled datasets, on the other hand, are used to train unsupervised models.

Image-Net is a well-known computer vision dataset that aided in the rise of deep learning. Image-net is commonly used for object recognition and classification. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) is a large-scale evaluation of algorithms for object detection and image classification. One high-level motivation is to enable the researchers to compare progress in detection across a broader range of objects, making use of the costly labelling effort. Another reason is to track the advancement of computer vision for large-scale image indexing for retrieval and annotation. The most popular challenge is the image classification and localization task. It is available on Kaggle.

Deep learning presents additional challenges, despite the expert knowledge required in this multidisciplinary field of performance analysis. Deep learning models are diverse, encompassing a wide range of problem domains, models, and datasets. The pace of this field is quick, and cutting-edge models evolve every few months. Deep learning software and hardware systems are diverse and rapidly evolving, as are performance metrics such as accuracy, time to train, and inference latency. Metrics must be carefully chosen. To avoid the bias and extract deep insights, performance analysis must be carried out with care. A comprehensive set of benchmarks representing the state-of-the-art and emerging workloads and datasets should be included in a proper performance analysis methodology. The researchers must have a solid understanding of the resource requirements and bottlenecks associated with the cross-stack workloads. Meanwhile, researchers must be aware of the new workloads.
Image classification and training model is created by choosing (a) a dataset of images as shown in figure 3 (b) a neural network, and (c) how many training epochs to run. This highlights two concepts. Deep Neural Networks are the adaptable techniques motivated by the human mind that allow the specialists to use the training strategies motivated by the human learning. Deep Neural Networks fall very short of the human brain. After one epoch, the proposed model is predicting no better than chance. Image Classification is the process of recognizing and categorising the images in either of the various pre-decided separate classes. As a result, image classification applications can define what is shown in a picture and what can differentiate one object from the other.

In image classification with position, an image is placed in any class and then a bounding box is drawn in the vicinity of an object to indicate where it is placed in that image. The detection of objects categorizes the several objects in an image and displays the position of every object with those bounding boxes. So, it is basically a modification of the image recognition with the localization for a number of objects. The data collection "Beagles Images" satisfies this criterion. After preparation, the data set Model is obtained.



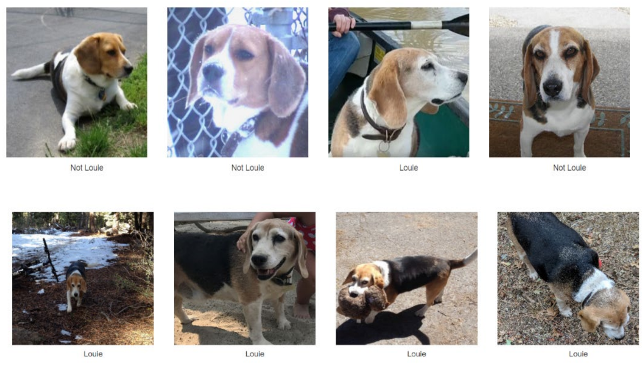


Figure 3. Dataset of Images

Each time the image is displayed, the output or prediction is produced as to whether the image contains a specific puppy or not. The model then produces a forecast that doesn't look any better than a chance. This time, the preparation takes a little longer (between 3 and 4 minutes).

While it trains, looking at the two graphs (figure 4 and figure 5), the two main concepts are illustrated - Learning Rate and Loss Rate. Learning rate is the rate at which each weight changes during training. Each weight is moving in the direction that reduces the loss at a value multiplied by the learning rate. The learning rate decreases throughout the training session because the network is getting closer to its ideal solution. At first, the network being trained knows nothing about what it may see. After a few epochs, the weights are getting better and hence it is important for the network to be less reactive to each image it sees. A training job is set up from this pre-trained model starting at the learning rate of 0.0001 as shown in figure 4. Training loss is calculated in the course of every epoch as shown in figure 5. While validation loss is computed after each epoch, training loss is regularly outlined during the whole epoch. But the validation metrics are calculated over the validation set only after the completion of the present training epoch. Sum () is a NumPy function that returns the sum of all array elements in the NumPy array. This sum () function can be used to count the number of pixels based on the criteria specified which is shown in figure 6 .The mean of all the images is taken (Mean is chosen for the prediction mainly because it can utilize all of the possible values present in the large dataset) as shown in figure 7 to identify the correct prediction about Louie image which is shown in figure 8. In figure 7, mean of image is calculated by dividing the sum of pixel values by the total count and subtracting the number of pixels in the dataset computed. In order to predict the Louie image as shown in figure 8, load the data into the memory. Just like training and testing the data, pre-process the data to meet the input requirements of our model and feed it to the network after pre-processing. The model will generate a prediction.

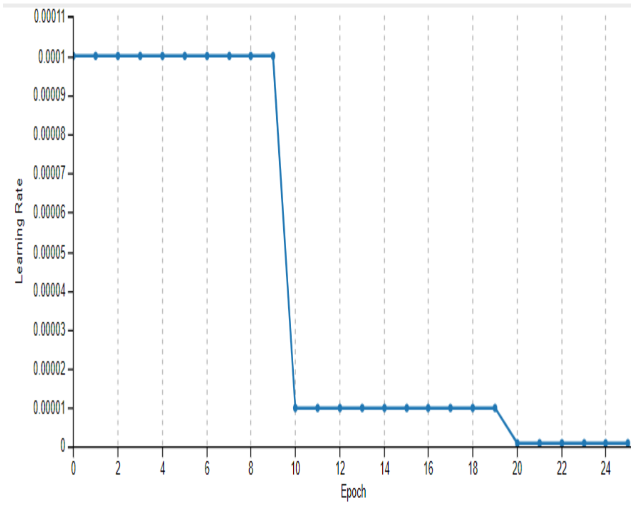


Figure 4. Performance Analysis of Learning Rate during Training

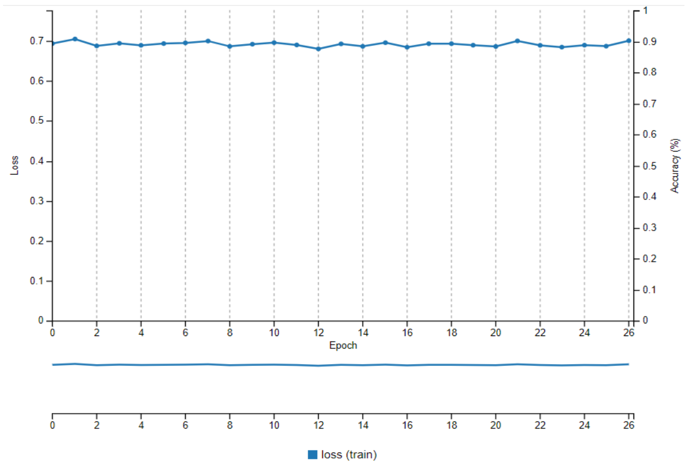
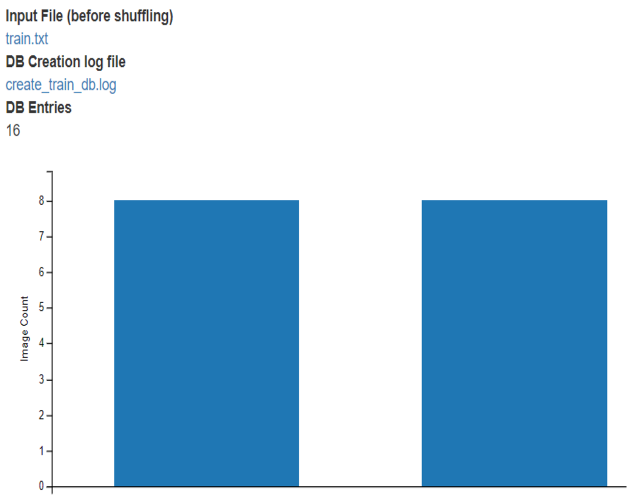


Figure 5. Performance Analysis of Loss rate during Training



Input File (before shuffling)
train.txt
DB Creation log file
create_train_db.log
DB Entries
16

Figure 6. Creation of Image Count

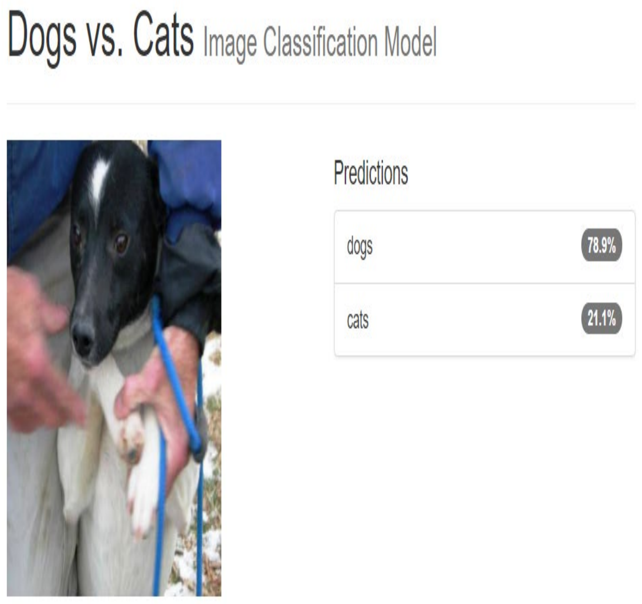
Figure 7. Image Mean

Figure 8. Predicted Image of Louie

After that, a neural network model is trained by exposing the neural network to data. When exposed to data, the neural network changes to create an accurate map between inputs and outputs. An input of 256 x 256 color images to an output specifies the likelihood that each image belongs to a *class*, in this case case *dog* or *cat.* There is training of a neural network model to classify dogs from cats. After that, the trained model is deployed into a simulator. With image classification, the proposed network takes an *image* and generates *classification.* More specifically, the input in this case is a 256 x 256 x 3 tensor of pixel values and output is a 2-unit vector (because of 2 classes) of probabilities. A trained network consists of two components - A description of the architecture of the untrained network and the weights that were learnt while the network was trained. Deployment is the work of taking a trained model and putting it to work as a part of an application. For example, edge devices can be deployed such as robots or autonomous vehicles. There may also be the deployment to a server in order to play a role in any piece of software. Using this model, one can identify whether cats or dogs are existing in a particular image.

The goal is to avoid a cat from entering into the house through a doggy door. For the purpose of this simulation, a program is created that does one thing when the network detects a dog and something different when the network detects a cat. Essentially, there is a need to combine deep learning with the traditional programming. An application could also have been created to see without deep learning but what deep learning can create to build an application has been used here. This step is called as d*eployment. The* trained neural network model can classify dogs and cats from DIGITS to create an application around it. This image classification model is shown in figure 9 that shows the predictions between cats and dogs. In the predication there is more probability of dogs.

The learning curve obtained from the training dataset provides a notion whether the model is learning properly. The learning curve on the basis of the hold-out validation dataset that provides an idea of how well the model generalizes is called Validation Learning Curve. The



Dogs vs. Cats Image Classification Model

Predictions

dogs	78.9%
cats	21.1%

Figure 9. Image Classification Model

Learning curve measured on the metric from which the various variables of the model are being optimized is called Optimization Learning Curve [43] for example - loss. The learning curve measured on the metric by which the model is accessed and chosen is called Performance Learning Curve, for example - accuracy. From the plot of loss as shown in figure 10, it might be observed that the proposed model has commensurate behaviour on training as well as validation datasets. If these similar plots begin to take-off harmoniously, it is an indication to terminate the training promptly. From the graph of accuracy, it may be seen that the model could possibly be upskilled somewhat more as the inclination for accuracy is increasing on both of the datasets since past few epochs. It can also be observed that the model has not learnt more so far in view of the training dataset that shows that the model is equivalently skilled on both the datasets.

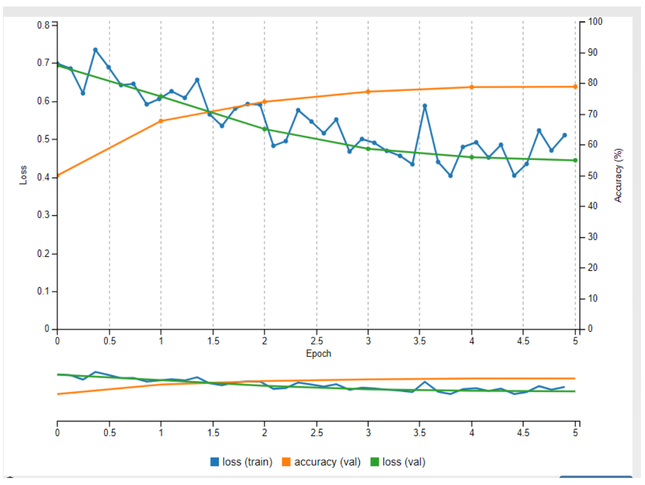


Figure 10. Performance Analysis of Loss with respect to Epoch for Training, Validation and Accuracy

6. Conclusion and Recommendations

Deep learning networks are emerging as a result of rising computing capacity on mobile hardware & the benefits of allowing excellent user experience. The behaviour and characteristics of deep-learning model training devices are largely non-transverse but realizing the performance characterization is crucial to the designing as well as execution of deep-learning models. The primary goal of this work is to investigate the performance of cutting-edge deep convolutional networks. In this work, a benchmark suite has been proposed for the training and deployment of Deep Neural Network. This encompasses a large variety of computer applications, from image recognition and machine translation to enhanced learning. DNN models have been implemented in three main configurations of deep learning: CNTK, MXNet and TensorFlow. Extensive research has been conducted using the various deep architectures such as AlexNet, InceptionNet, etc. To the best of authors' knowledge, this is the first work that presents a quantitative analysis of the deep architectures mentioned above. These models have been used to conduct the comprehensive behavioural survey and outlined to provide the effectiveness of Deep Neural Network training for the various configurations of hardware. The effectiveness of these methods has been evaluated using metrics such as learning rate, accuracy and epoch. The future scope of this research includes layer-by-layer fine-tuning of the ResNet50 architecture with larger datasets to improve the accuracy even further.

References

[1] Sindhwani, N.; Verma, S.; Bajaj, T.; and Anand, R.; "Comparative Analysis of Intelligent Driving and Safety Assistance Systems Using YOLO and SSD Model of Deep Learning." *International Journal of Information System Modeling and Design* 12, no. 1 (2021): 131-146.

[2] Kamalraj, R.; Neelakandan, S.; Kumar, M. R.; Rao, V. C. S.; Anand, R.; and Singh, H.; "Interpretable filter based convolutional neural network (IF-CNN) for glucose prediction and classification using PD-SS algorithm." *Measurement* 183: 109804.
[3] Ren, A.; Li, Z.; Ding, C.; Qiu, Q.; Wang, Y.; Li, J.; ... and Yuan, B.; "Sc-dcnn: Highly-scalable deep convolutional neural network using stochastic computing." *ACM SIGPLAN Notices* 52, no. 4 (2017): 405-418.
[4] Gupta, A.; Yadav, V.; "Hybrid Intelligence Model on the Second Generation Neural Network", *International Journal of Advanced Intelligence Paradigms*, Vol. 18, No. 3, pp. 398-416, February 2021.
[5] Rhu, M.; Gimelshein, N.; Clemons, J.; Zulfiqar, A. and Keckler, S. W.; "vDNN: Virtualized deep neural networks for scalable, memory-efficient neural network design." In *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO),* pp. 1-13. IEEE, 2016.
[6] Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; .... and Fei-Fei, L.; "Imagenet large scale visual recognition challenge." *International journal of computer vision* 115, no. 3 (2015): 211-252.
[7] Anand, R.; Singh, B.; and Sindhwani, N.; "Speech Perception & Analysis of Fluent Digits' Strings using Level-By-Level Time Alignment." *International Journal of Information Technology and Knowledge Management* 2, no. 1 (2009): 65-68.
[8] Juneja, S.; and Anand, R.; "Contrast Enhancement of an Image by DWT-SVD and DCT-SVD." In *Data Engineering and Intelligent Computing,* pp. 595-603. Springer, Singapore, 2018.
[9] Anand, R.; Shrivastava, G.; Gupta, S.; Peng, S. L.; and Sindhwani, N.; "Audio Watermarking With Reduced Number of Random Samples." In *Handbook of Research on Network Forensics and Analysis Techniques,* pp. 372-394. IGI Global, 2018.
[10] Umar, M.; Sabir, Z.; Raja, M. A. Z.; Baskonus, H. M.; Yao, S. W.; and Ilhan, E.; "A novel study of Morlet neural networks to solve the nonlinear HIV infection system of latently infected cells." *Results in Physics* 25 (2021): 104235..
[11] Parveen, H,; Yadav, V; "A Particle Swarm Optimization Strategy using QSAR modeling on the Second generation Neural Network", *IEEE International Conference on Advances in Computing, Communication Control & Networking (ICACCCN-2018)*, pp. 1188-1199, 2018.
[12] Umar, M.; Sabir, Z.; and Raja, M. A. Z.; "Intelligent computing for numerical treatment of nonlinear prey–predator models." *Applied Soft Computing* 80 (2019): 506-524.
[13] Sabir, Z.; Raja, M. A. Z.; Umar, M.; and Shoaib, M.; "Neuro-swarm intelligent computing to solve the second-order singular functional differential model." *The European Physical Journal Plus* 135 no. 6 (2020): 1-19.
[14] Schmidhuber, J.; "Deep learning in neural networks: An overview." *Neural networks 61*(2015): 85-117.
[15] Sabir, Z.; Guirao, J. L.; and Saeed, T.; "Solving a novel designed second order nonlinear Lane–Emden delay differential model using the heuristic techniques." *Applied Soft Computing* 102 (2021): 107105.
[16] Abdelkawy, M. A.; Sabir, Z.; Guirao, J. L.; and Saeed, T.; "Numerical investigations of a new singular second-order nonlinear coupled functional Lane–Emden model." *Open Physics*, 18, no. 1 (2020): 770-778.
[17] Sabir, Z.; Wahab, H. A.; Umar, M.; and Erdoğan, F.; "Stochastic numerical approach for solving second order nonlinear singular functional differential equation." *Applied Mathematics and Computation*, 363 (2019): 124605.
[18] Hordri, N. F.; Yuhaniz, S. S.; and Shamsuddin, S. M.; "Deep learning and its applications: a review." In *Conference on Postgraduate Annual Research on Informatics Seminar,* 2016.
[19] Shrestha, A.; and Mahmood, A.; "Review of deep learning algorithms and architectures." *IEEE Access* 7 (2019): 53040-53065.
[20] Dey, A.; "Machine learning algorithms: a review." *International Journal of Computer Science and Information Technologies* 7 no. 3 (2016): 1174-1179.
[21] Muhammad, I.; and Yan, Z.; "Supervised machine learning approaches: A survey." *ICTACT Journal on Soft Computing*, 5, no. 3 (2015): 946-952.
[22] Voulodimos, A.; Doulamis, N.; Doulamis, A.; and Protopapadakis, E.; "Deep learning for computer vision: A brief review." *Computational intelligence and neuroscience*, 2018. https://doi.org/10.1155/2018/7068349
[23] Ren, Y.; Yoo, S.; and Hoisie, A.; "Performance analysis of deep learning workloads on leading-edge systems." In *2019 IEEE/ACM Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS),* pp. 103-113. IEEE, 2019.
[24] Zhao, Z. Q.; Zheng, P.; Xu, S. T.; and Wu, X.; "Object detection with deep learning: A review." *IEEE transactions on neural networks and learning systems* 30, no. 11 (2019): 3212-3232.
[25] Liu, J.; Liu, J.; Du, W.; and Li, D.; "Performance analysis and characterization of training deep learning models on nvidia tx2." *arXiv preprint arXiv* 1906 (2019): 04278.
[26] Pouyanfar, S.; Sadiq, S.; Yan, Y.; Tian, H.; Tao, Y.; Reyes, M. P.; ... and Iyengar, S. S.; "A survey on deep learning: Algorithms, techniques, and applications." *ACM Computing Surveys (CSUR)* 51, no. 5 (2018): 1-36.
[27] Schwartzman, A.; Kagan, M.; Mackey, L.; Nachman, B.; and De Oliveira, L.; "Image Processing, Computer Vision, and Deep Learning: new approaches to the analysis and physics interpretation of LHC events." *Journal of Physics: Conference Series* 762, no. 1 (2016): 012035.
[28] Mouhssine,R.; Otman, A.; and Haimoudi, E.K; "Performance Analysis of Machine Learning Techniques for Smart Agriculture: Comparison of Supervised Classification Approaches." *International Journal of Advanced Computer Science and Applications,* 11, no. 3 (2020): 610-619.
[29] Kim, H.; Nam, H.; Jung, W.; and Lee, J.; "Performance analysis of CNN frameworks for GPUs." In *2017 IEEE*

International Symposium on Performance Analysis of Systems and Software (ISPASS), pp. 55-64. IEEE, 2017.

[30] Rahul, M,; Yadav, V,; "Zernike Moments based Facial Expression Recognition using Two staged Hidden Markov Model", *Advances in Computer Communication & Computational Sciences*, Vol. 924, pp. 661-670, 2019.

[31] Khan, A. I.; and Al-Habsi, S.; "Machine learning in computer vision." *Procedia Computer Science* 167 (2020): 1444-1451.

[32] O'Mahony, N.; Campbell, S.; Carvalho, A.; Harapanahalli, S.; Hernandez, G. V.; Krpalkova, L.; ... and Walsh, J.; "Deep learning vs. traditional computer vision." In *Science and Information Conference,* pp. 128-144. Springer, Cham, 2019.

[33] Suhail, A.; Jayabalan, M.; and Thiruchelvam, V.; "Convolutional neural network based object detection: A review" *Journal of Critical Reviews* 7*,* no. 11 (2020): 786-792.

[34] Liu, W.; Wang, Z.; Liu, X.; Zeng, N.; Liu, Y.; and Alsaadi, F. E.; "A survey of deep neural network architectures and their applications." *Neurocomputing* 234 (2017): 11-26.

[35] Sze, V.; Chen, Y. H.; Yang, T. J.; and Emer, J. S.; "Efficient processing of deep neural networks: A tutorial and survey." *Proceedings of the IEEE* 105, no. 12 (2017): 2295-2329.

[36] Bengio, Y.; "*Learning deep architectures for AI.*" Now Publishers Inc.(2009).

[37] Bakshi, G.; Shukla, R.; Yadav, V.; Dahiya, A.; Anand, R.; Sindhwani, N.; and Singh, H.; "An optimized approach for feature extraction in multi-relational statistical learning." *Journal of Scientific & Industrial Research* 80 (2021): 537-542.

[38] Wang, Z.; Liu, K.; Li, J.; Zhu, Y.; and Zhang, Y.; "Various frameworks and libraries of machine learning and deep learning: A survey." *Archives of computational methods in engineering*, (2019): 1-24. https://doi.org/10.1007/s11831-018-09312-w

[39] Sharma, D.; and Kumar, N.; "A review on machine learning algorithms, tasks and applications." *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)* 6, no. 10 (2017)): 2278-1323.

[40] LeCun, Y.; Bengio, Y.; and Hinton, G.; "Deep learning." N*ature* 521 (2015): 436-444.

[41] Abu, M. A.; Indra, N. H.; Abd Rahman, A. H.; Sapiee, N. A.; and Ahmad, I.; "A study on Image Classification based on Deep Learning and Tensorflow." *International Journal of Engineering Research and Technology* 12, no. 4 (2019): 563-569.

[42] Zhu, S.C.; and Mumford, D.; "A stochastic grammar of images. *Foundations and Trends in Computer Graphics and Vision* 2, no.4 (2006): 259-362.

[43] Sindhwani, N.; "Performance Analysis of Optimal Scheduling Based Firefly algorithm in MIMO system." O*ptimization* 2, no. 12 (2017): 19-26.