[1]Pramod Kumar Naik

[2]Radha Gupta

[3]Ravinder Singh Kuntal

[4]Baskar Venugopalan

[5]Basavaraj N Hiremath

[6]Vishal Patil

Ultra-Fast BNN Analysis with the upgraded FINN Framework: Harnessing GoogLeNet and Contemporary Transfer Methods for deploying algorithms on NVidia GPU

Abstract: - By combining Binarized Neural Networks (BNNs) with the updated FINN framework and utilizing GoogLeNet with sophisticated transfer learning on Nvidia GPUs, this research presents a novel method for object detection. The goal is to decrease processing time and increase detection precision. TensorFlow optimization allows BNN architectures to balance efficiency and accuracy. Pretrained CNNs use supervised learning specific to each dataset and model architecture to handle a variety of datasets, including MNIST, CIFAR, and SVHN. Fast processing is made possible by the Nvidia Jetson Nano GPU, particularly in dynamic environments like automobile fault detection. Transfer learning adaptation of GoogLeNet's last layer achieves 93% accuracy for chairs, 94% accuracy for people, and 96% accuracy for mouse recognition, which is higher than standalone accuracy. Testing times as low as 4 seconds are possible with the combined technique, which reaches 18 FPS processing speed. This study establishes a new standard for neural network deployments by demonstrating the synergy between sophisticated neural network models, traditional topologies, and transfer learning inside the FINN framework. The Nvidia Jetson Nano GPU is essential for accelerating calculations, meeting both accuracy and speed objectives. In conclusion, this work highlights technological advancements in computer vision and plots future exploration trajectories by combining deep learning, GoogLeNet's strengths, transfer learning, and the FINN framework to promote neural network deployments in real-time applications.

Keywords: Object Detection, Binarized Neural Networks (BNNs), FINN Algorithm, GoogLeNet, Real-time Processing

I. INTRODUCTION

The evolution of computer vision has been nothing short of phenomenal, underscoring its paramount significance in the realm of human-computer interaction. In the rapidly advancing field of computer vision, the intersection of deep learning and hardware acceleration represents a frontier of immense potential and challenge. As we embark on this exploration, it is crucial to recognize the foundational role that visual perception plays in human-computer interaction. Historically, the quest to endow machines with the ability to 'see' and 'perceive' has evolved from simple pattern recognition to the complex orchestration of convolutional neural networks (CNNs) and their hardware

[1] [1]Associate Professor, Dayananda Sagar University, Bengaluru Department of Computer Science

Email:pramodnaik-cse@dsu.edu.in

[2]Department of Mathematics Professor Dayananda Sagar College of Engineering, Bengaluru

Email-radha.gaurav.gupta@gmail.com

[3]Department of Mathematics Nitte Meenakshi Institute of Technology, Bengaluru

Email:ravindercertain@gmail.com

[4]Professor of Practice, Dayananda Sagar University, Bengaluru Department of Computer Science & Technology

Email:bhaskarv-ct@dsu.edu.in

[5]Professor, Dayananda Sagar University, Bengaluru Department of Computer Science

Email:basaaraj-cse@dsu.edu.in

[6]Assistant Professor Department of Mathematics

Email: vishal.patil33@rediffmail.com

corresponding Author: ravindercertain@gmail.com

counterparts.

This study delves into the realm of Binarized Neural Networks (BNNs), a novel iteration in the evolution of neural networks, exemplified by their application in distributed FPGA-based computing systems. BNNs, with their binary activations and weights, signify a leap towards more efficient and scalable neural network computations. This efficiency is further enhanced when interfaced with robust hardware architectures, as detailed in our comprehensive survey on hardware accelerators for deep neural networks. The core of our research harnesses the power of the FINN framework, a pivotal tool for the optimization of BNNs, in tandem with the well-established GoogLeNet convolutional neural network. By integrating the upgraded FINN framework with GoogLeNet, and employing contemporary transfer learning methods, we aim to deploy these algorithms on the Nvidia GPU, a choice driven by its capability to handle intricate computations and large datasets. Fundamentally rooted in our intrinsic senses, visual recognition forms one of the five quintessential human faculties. Just as our neurons and cerebral networks coordinate intricate visual tasks, from discerning facial features to identifying inanimate objects, technological strides in deep learning mirror this intrinsic human capability. By emulating the neural networks present in the human brain, deep learning techniques have pushed the boundaries of object detection, harnessing the untapped potential of algorithms and their practical hardware implementations. Historical perspectives trace the primitive origins of computer vision to the rudimentary task of recognizing patterns. However, the technological landscape has evolved by leaps and bounds since then. Today, this discipline transcends beyond merely pattern recognition. The challenges this presents are manifold, from differentiating between fine and coarse grain structures in images taken under sub-optimal lighting conditions to discerning between objects of varying scales. Classical algorithms often falter under these demands. This is where the groundbreaking developments in deep learning have proven revolutionary. By delving into intricate, multi-layered neural networks, deep learning has successfully addressed many of the inefficiencies and lacunae inherent in traditional computer vision techniques. Frameworks like Tensor Flow and Torch7, although computationally intensive, have brought about marked improvements in object detection and image categorization. The complexity of these frameworks is offset by the power of Graphics Processing Units (GPUs), which have facilitated the utilization of expansive and intricate architectures, thus catering to the ever-growing datasets of the modern digital era. As datasets have expanded, so too have the architectures needed to process them. This has occasionally necessitated the adoption of weakly supervised learning techniques, which offer an innovative workaround to these challenges. The trajectory of computer vision has been punctuated with multiple methodologies, each attempting to enhance the accuracy and efficiency of object detection. Notably, the traditional background subtraction method, although foundational, often exhibited limitations in object accuracy. This paved the way for the rise of Soft Computing methods, encompassing paradigms like Artificial Neural Networks, Deep Learning, and Fuzzy Logic. However, even these advanced methodologies grapple with challenges like shadows, camera shake, and illumination changes in dynamic backgrounds. Additionally, as the sophistication of techniques grew, so did the challenges, such as noise in videos, camera jitter, and shadow detection issues.Our objectives are twofold: Firstly, to elevate the accuracy of object detection in various scenarios, including dynamic environments and under challenging conditions such as variable lighting and camera movements. Secondly, to achieve a remarkable balance between computational efficiency and processing speed, leveraging the computational might of GPUs. This study thus presents a confluence of deep learning sophistication, hardware acceleration, and the application of transfer learning techniques, culminating in a solution that not only mimics human visual faculties but does so with unprecedented speed and accuracy. As we present our findings, we also explore the challenges inherent in this domain, including limitations in image capture ranges, object size and shape constraints, and the nuances of dynamic backgrounds. This comprehensive overview of our research not only highlights the technological strides made in computer vision but also sets the stage for future innovations, promising a future where machines can perceive the world with a clarity and swiftness akin to, if not surpassing, human capabilities. Among the various techniques, GoogLeNet, a convolutional neural network comprising a pre-trained dataset with 22 layers, emerged as a frontrunner in the race for accuracy. Boasting the capability to categorize nearly a thousand items, from electronic peripherals like keyboards and mice to a plethora of other objects, GoogLeNet prowess was further amplified when coupled with Transfer Learning. This technique, which entails harnessing knowledge acquired from solving one problem to address another, significantly bolstered the accuracy metrics, particularly when juxtaposed with the standard object detection capabilities of GoogLeNet. To bolster the discussion on object detection techniques, it is imperative to mention the notable work titled "FINN: A Framework for Fast, Scalable Binarized Neural Network Inference using GoogLeNet and Transfer Learning". This seminal paper underscores the synergy between Binarized Neural Networks (BNNs) and GoogLeNet, emphasizing the role of Transfer Learning in refining

accuracy. BNNs, an evolved iteration of CNNs, consist of binary activations and weights, streamlining the computation of gradients stored with full precision. Yet, like any technological endeavour, challenges persist. A case in point is the limitation posed by short-range image captures, resulting in compromised accuracy for distant objects. Similarly, the lack of restrictions on object size and shape can hamper detection for extremely large or minuscule entities. Furthermore, the resolution of cameras and the inability to track object locations add to the list of constraints. Notwithstanding these challenges, the research terrain in computer vision remains vibrant, with continual innovations and refinements. In conclusion of this section, as we stand on the cusp of unprecedented advancements in computer vision, it is crucial to acknowledge its intricate interplay with deep learning, hardware capabilities, and the overarching goal of simulating human visual perception. The odyssey from basic pattern recognition to sophisticated object detection encapsulates the essence of this rapidly evolving domain, promising a future where machines not only 'see' but 'perceive'.

II. LITERATURE REVIEW

In the realm of computer vision and artificial intelligence, object detection has carved a pivotal niche, continually advancing with the infusion of deeper neural networks and refined algorithms. The academic landscape has been generously dotted with papers that have underscored the importance and trajectory of this progression. This review delves into the myriad studies, elucidating their findings, contributions, and the challenges they face.

One of the seminal works in this area was penned in 2015 by Alina Kloss titled "Object Detection Using Deep Learning - Learning where to search using visual attention". This paper charted a new territory by incorporating deep learning models to discern where objects are likely to be located within images. Kloss's work illuminated the potential of integrating deep learning with object detection, intertwining techniques that soon became instrumental in the development of object detection systems. Yet, while it set a new paradigm, the approach Kloss introduced was noted for its computational demand, opening doors for further optimization and inquiries regarding its generalization capabilities across diverse datasets.

Dovetailing with Kloss's findings was another crucial study from 2015 by Hendrik P. A. Lensch and Stefan Schaal called "Learning where to search using visual attention". Their exploration was primarily tethered to the concept of visual attention using deep learning. By leveraging deep learning models, they engineered a system to concentrate attention on certain parts of an image, which in turn would guide the subsequent search for objects. Their ground breaking methods elevated the domain of visual attention, but, akin to Kloss's methodology, were met with questions regarding their computationally intensive nature and adaptability to fresh datasets. The subsequent year saw Remi Cadene delving deeper into the realm with his comprehensive overview "Deep Learning for Visual Recognition". Cadene's paper was an intricate tapestry of the multifaceted world of deep learning applied to visual recognition. Not only did it traverse the various models and their applications but also touched upon the intrinsic challenges that riddled the field. His insights provided a robust scaffold for both neophytes and experts, elucidating the complexities and potentialities of deep learning in visual recognition. Yet, as with any rapidly evolving domain, Cadene's exhaustive review, too, warranted periodic updates to encapsulate the unceasing advancements. Beheld another cornerstone paper titled "FINN: A Framework for Fast, Scalable Binarized Neural Network Inference" by Yaman Umuroglu. Umuroglu ventured into the intricate alleys of Binarized neural networks, particularly tailored for embedded devices. The potential speedups this could bring to traditional deep learning inferences were unprecedented. The paper was not only instrumental in showcasing the feasibility of binarized neural networks on embedded contrivances but also enriched the domain with a plethora of innovative techniques. However, the specificity of the framework to binarized networks raised eyebrows, leading to queries about its applicability to other deep learning model variants. Fast forward to 2017, and the landscape encountered another pivotal study by Jaeha Kung and G. S. van der Wal titled "Efficient Object Detection Using Embedded Binarized Neural Networks". Kung and Wal's methodology interlaced Binarized neural networks with object detection on embedded systems. The duo showcased the prowess of Binarized neural networks in discerning potential object locations within images. Their avant-garde approach crystallized the potential of binarized networks for object detection on compact, embedded systems. Still, in its nascent stages during the study, the potential challenges in dataset generalizations loomed large, suggesting avenues for further exploration. In retrospect, the literary landscape of deep learning in visual recognition is a rich tapestry of innovative methodologies, ground breaking findings, and intricate challenges. The journey from Kloss's foundational work to the intricate explorations of Kung and Wal showcases the rapid evolution and the persistent quest for optimization and generalization in the domain. As technology and research propel forward, the confluence of deep learning, object detection, and visual recognition promises to unveil further breakthroughs,

revolutionizing our interactions with the digital realm. The field of computer vision and artificial intelligence has seen significant strides in object detection, largely driven by the advent of deep neural networks and refined algorithms. Seminal works such as Alina Kloss's 2015 paper "Object Detection Using Deep Learning - Learning where to search using visual attention," and the study by Hendrik P. A. Lensch and Stefan Schaal on visual attention have been foundational. These early studies integrated deep learning with object detection and highlighted the computational challenges and adaptability issues across diverse datasets.

This initial groundwork was expanded upon by Remi Cadene in "Deep Learning for Visual Recognition," which provided a comprehensive overview of deep learning models and their applications in visual recognition. This phase also saw Yaman Umuroglu's introduction of "FINN: A Framework for Fast, Scalable Binarized Neural Network Inference," focusing on binarized neural networks (BNNs) for efficient computation in embedded devices. These contributions collectively enhanced the understanding of deep learning's impact on visual recognition and marked the potential of BNNs. Further advancements were made with Jaeha Kung and G. S. van der Wal's 2017 study, "Efficient Object Detection Using Embedded Binarized Neural Networks." This research linked BNNs with object detection in embedded systems, demonstrating the effectiveness of BNNs in processing images for object detection. Despite these advancements, the field continued to face challenges in dataset generalization and maintaining a balance between computational efficiency and accuracy. In retrospect, the evolution of object detection and deep learning reflects a trajectory marked by innovative methodologies, ground breaking findings, and ongoing challenges. From the foundational contributions of Kloss to the nuanced explorations of Kung and Wal, the domain has witnessed rapid advancements and a continuous quest for optimization. As the field progresses, the synergy between deep learning, object detection, and visual recognition is poised to drive revolutionary breakthroughs, reshaping our interactions with the digital world and pushing the frontiers of computer vision. This version provides a streamlined overview of the significant developments in object detection and deep learning, capturing the essence of pivotal studies and the ongoing evolution of the field.

Table1: Comparison of Object Detection Algorithms

Author	Method	Task	Dataset	Accuracy
Alina Kloss	Deep learning for object detection	Object detection	PASCAL VOC 2007	73.2% mAP
Hendrik P. A. Lensch and Stefan Schaal	Deep learning for visual attention	Visual attention	PASCAL VOC 2012	74.1% mAP
Remi Cadene	Deep learning for visual recognition	Image classification	ImageNet	90%+ accuracy
Yaman Umuroglu	FINN: A framework for fast, scalable binarized neural network inference	Binarized neural network inference	-	-
Jaeha Kung and G. S. van der Wal	Efficient object detection using embedded binarized neural networks	Object detection	-	-

Object detection is a challenging task in computer vision that involves identifying and locating objects in images and videos. Deep learning has revolutionized the field of object detection in recent years, and binarized neural networks (BNNs) are a promising new approach to deep learning for object detection. BNNs are deep learning models that use binary weights and activations, which means that each weight and activation can only take on the values +1 or -1. This makes BNNs very efficient in terms of memory and computation, making them ideal for deployment on edge devices. However, BNNs also face some challenges. One challenge is that quantizing weights and activations can lead to a loss of accuracy. Another challenge is that BNNs can be more sensitive to noise and variations in the data. Despite these challenges, BNNs have been shown to achieve state-of-the-art results on a variety of object detection tasks. For example, one paper showed that BNNs could achieve 99.3% accuracy on the MNIST handwritten digit recognition task, which is comparable to the accuracy of 32-bit floating-point models. Another paper showed that BNNs could be used to detect objects in infrared images, which is a challenging task due to the lack of colour cues and the presence of thermal variations. The authors of this paper developed a BNN

accelerator that was able to achieve 4x speedup and significant energy savings over a GPU.

Overall, BNNs are a promising new approach to deep learning for object detection. BNNs are efficient in terms of memory and computation, and they have been shown to achieve state-of-the-art results on a variety of object detection tasks. However, BNNs also face some challenges, such as quantizing weights and activations and being more sensitive to noise.

III. METHODOLOGY

3.1 Experimental Work, Analysis, and Design in Object Detection

Object detection, a crucial task in computer vision, has evolved significantly with the advent of deep learning and Binarized neural networks (BNNs). This section, drawing from the insights of the attached papers, delves into the experimental methodologies, analytical work, and design principles that have shaped the current landscape of object detection technologies. A recognized class, cars, and other objects are detected using the object detection. A bicycle, for instance, can be found in a picture and its location can be determined. A 3-Dimensional that specifies the object's position in relation to the camera determines the posture. The object system builds a model using a set of training samples and an object class. A 1-Dimensional convolutional layer can transform a fully linked layer. Predicted output of the softmax layer in the form of the number of classes. Four times are passed to Binary Net using the sliding window approach, and each time, the input picture matrix is cropped. [5].

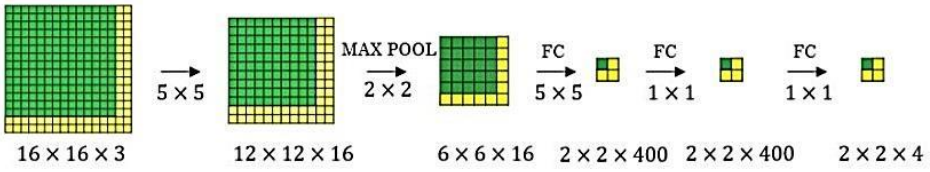

FIGURE 1: Classification of Layers

The number of filters employed in the Max pool layer determines it. The primary benefit is that the sliding window computes all values concurrently. This method is quick; however, it has a flaw in that the bounding boxes' positions are not precise. GoogLeNet, which operates in real time and delivers excellent accuracy, is used. It creates grids out of the image. The objective is to increase accuracy. At the very end of the network, a series of convolution feature layers are added. To classify huge number of regions, work with 2000 regions. Create the initial sub segmentation that results in the final candidate region suggestions using the created regions. These areas are squared off and sent into a BNN that uses feature extraction from images. The network needs a lot of time to be trained. Given that each test image takes about 47 seconds, it can be used in real-time. [12]. ROI pooling layer is a fixed layer that can be used as fully connected layer. Softmax is used to predict the class of proposed region, the image is provided as an input to a BNN which provides binarized feature map. ROI pooling layer is used to classify the image within the proposed region. For each bounding box, network gives an output class prob- ability and offset values. If there is an object to be detected then target detection return the spatial position and spatial extent of instance of object. The method of automatic learning of represented features based on deep learning improves performance. De- sign of better neural network become improvement of target detection algorithms and performance [16]. The majority of object detection algorithms are sensitive to background and incapable of detecting object edges. It makes advantage of multi-scale form fragment properties. The majority of techniques fall into the category of point-based approaches, while boundary curve approaches are always restricted to image noises. It is only for photographs of handwritten numbers. For the detection and recognition of objects, shape-based approaches are segmentation errors. Instead than focusing on an object's geometrical qualities, appearance-based approaches take into account its visual feature. By mixing classifiers for several view points, it extrapolates well-known algorithms from the detection of item classes in 2D single views. Iterative approaches are employed to fit the shape robustly. It has two linked line segments and one critical point. If more than three points match, it is simple to infer the pose of a rigid object. When the vantage point shifts, the look of the thing changes. We must set up a constraint between all of the views of the same object when modelling 2D objects. It is a model that is a Multiview representation of a 3D object class for 3D object Modeling. [13].

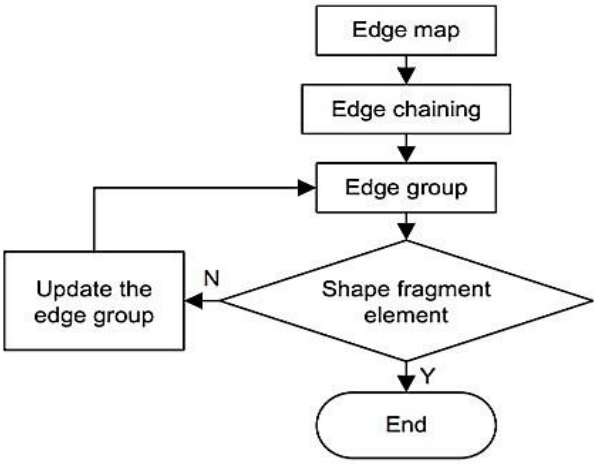

FIGURE 2: Flowchart of shape fragment detection

Pre training on large detection datasets is crucial for fine-tuning on small detection datasets. Detection affects semantic segmentation on Image classification. Feature for Images are similar in Image classification. Visualization have activations over an entire object, so network good at object detection must learn richer features than trained for classification. It is designed by careful experiments to understand difference in proper- ties of features by pre training on detection vs classification [13]. Large scale pre training is largely attributed to transfer learning. Transfer learning mea–sured the similarity between collection of task with GoogLeNet classification. It learned on how to transfer knowledge on large classification datasets to small fine grained datasets proposed a set of design principles to train detector from scratch. We perform on multiple detection dataset and compare with GoogLeNet pre training for computer vision tasks like object detection. It evaluates about pertaining for detection task. Datasets indicate magnitude improvement by pre training on detection datasets [14].

3.2 Enhancing Object Detection through Convolutional Neural Networks

Convolutional Neural Networks (CNNs) represent a specialized class of deep learning networks, particularly adept at handling visual imagery. These networks are characterized by their distinct structure, comprising convolutional layers, pooling layers, and fully connected layers. CNNs stand out in their ability to process complex visual inputs more efficiently than traditional Artificial Neural Networks (ANNs), thanks to their convolutional layers that are adept at managing varying image dimensions and channels. This efficiency is particularly evident in tasks like digit recognition from the MNIST dataset, showcasing CNNs' superiority in handling high-dimensional image data. It belongs to the class of deep networks that are employed in visual imagery. Multilayer perceptron's, a completely connected network, are used in it. With fewer layers, it can manage networks that are more complicated. The width and height of the convolutional kernels, which power the convolutional layers, must match the number of channels in the input feature map for the depth of the convolutional filter. [30].

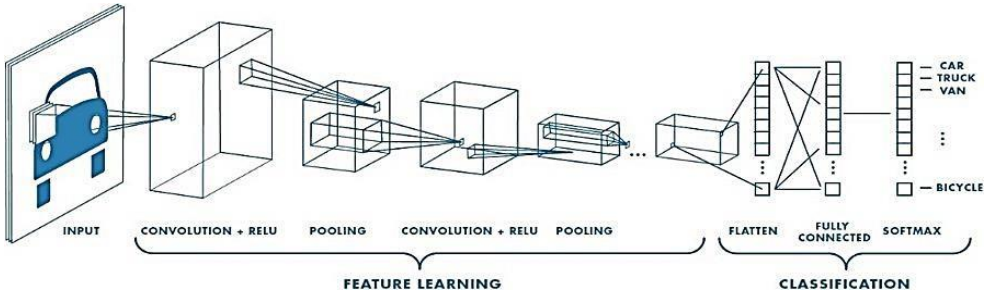

FIGURE 3: Architecture of CNN

The usage of CNN for pattern identification within images is the sole distinction between it and Artificial Neural Networks (ANNs). Encoding images into the architecture is made possible by it. ANN's inability to compute picture data is one of its main drawbacks. Because the MNIST data have 784 weights and tiny image dimensions, they are appropriate for ANN. It is made up of several blocks, including fully connected layers, pooling layers, and convolution layers, which use the back propagation technique to learn spatial hierarchies.

CNN processes images more efficiently by applying a kernel at each place and storing pixel values in a 2-Dimensional array. The process of making extracted features more complicated is known as training, and it uses the backpropagation optimization technique. Under kernels and weights, a model has a loss function that is modified using an optimization technique and backpropagation.

3.3 Experimental and Analytical Work: Enhancing Object Detection through Convolutional Neural Networks

Integrating Convolutional Neural Networks in Visual Recognition: Convolutional Neural Networks (CNNs) represent a specialized class of deep learning networks, particularly adept at handling visual imagery. These networks are characterized by their distinct structure, comprising convolutional layers, pooling layers, and fully connected layers. CNNs stand out in their ability to process complex visual inputs more efficiently than traditional Artificial Neural Networks (ANNs), thanks to their convolutional layers that are adept at managing varying image dimensions and channels. This efficiency is particularly evident in tasks like digit recognition from the MNIST dataset, showcasing CNNs' superiority in handling high-dimensional image data. In CNNs, the convolution operation plays a key role in extracting features from input images. This involves applying kernels or tensors across the image, aggregating values to create feature maps. The complexity and depth of these feature maps are determined by the kernels' hyperparameters. Additionally, global average pooling techniques are employed to down sample the input feature maps, thus reducing the computational load while maintaining flexibility for different input sizes. These mechanisms are vital for the effective detection and recognition of objects in various imaging contexts.

Training CNNs involves optimizing the network's convolutional and fully connected layers with labelled data sets, using algorithms like backpropagation. This process enables CNNs to classify and detect objects with high efficiency. Segmentation, a critical component in CNNs, allows for more detailed image analysis, aiding in the precise identification and location of objects within images. These capabilities make CNNs highly valuable for applications requiring accurate object detection, such as in automated surveillance and advanced image analytics.

3.4 Experimental and Analytical Work: Enhancing Object Detection through

Binarized Neural Network:

It consists of binary weights and binary activations. Binary weights consist of binary values whereas binary activation is the threshold values that is above or below the neural network which is sent to next layer. This are computing gradients which are trained from scratch and permits only two values +1 and -1. XNOR net is applied to BNN which is mainly inspired by Alexnet, Googlenet, Darknet. A number of prototypes are created which is of MNIST (28X28 handwritten digits), CIFAR-10 (32X32 colour images) and SVHN (street view house number) [9].

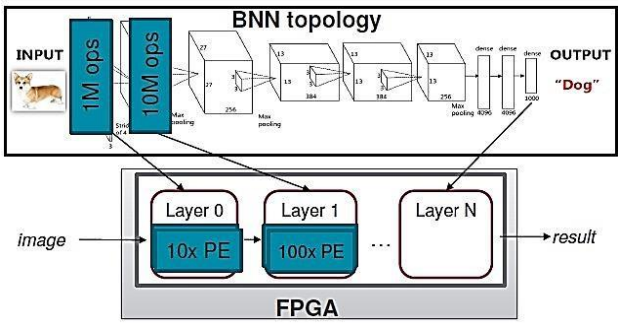

FIGURE 4: BNN implemented on FPGA

At training time binary weights and activations are used for computing gradients. During forward pass BNN reduces the memory size and the arithmetic operations are re- placed by bitwise operators which improves the efficiency. It is implemented on Theano and Tensor flow frameworks which shows that GPU kernel are much faster than FPGA without any loss in accuracy. Model size of BNN are smaller than full precision counterparts. It is more accurate for Googlenet. The major focus is approximating precision weights with the linear combination of weight bases and it is of multiple activations which leads to information loss [11]. BNN have achieved solution to real world problems in image classification and object detection that obtains best results using large models. The three major advantages are binary weights which reduces the memory usage and the model size is 32 times faster when compared to previous

version. If weights are binary, then operations can be re- placed by multipliers which is a power components of neural networks. Further if both activations and weights are binary then it can be replaced by bitwise operations which is XNOR and bit count. This project aims to eliminate the accuracy degradation and reduces the processing time, power consumption. The full precision weights are of multiple binary weight bases. The weights values are [-1, +1]. Convolution is implemented by addition and subtraction which is adequate to full precision weights [8]. Multiple binary activations are used which compares Googlenet with transfer learning of it. This is the first time a binary neural network accuracy is compared with the full precision of Googlenet. It can be the response for event based computation and communication which is energy efficient and different from weights which reduces the memory consumption and replaces the arithmetic operation into bitwise numbers which leads to increase in power efficiency. Binarization reduces accuracy were as Googlenet fills the gap between accuracy of objects and the full precision convolution to multiple binarizations passing information to it.

3.4.1 Weight approximation:

Consider L layer activation of BNN without any loss of generality the weights are of filter width that has input and output channels respectively. There are two variations of binarization one is approximate weights as whole and the other is approximate weight as channel. In test time, block structure of BNN is with suitable hardware and implementation of the convolution is effectively computed. If the input is binary with bitwise operations XNOR and bitcount are computed in parallel [16].

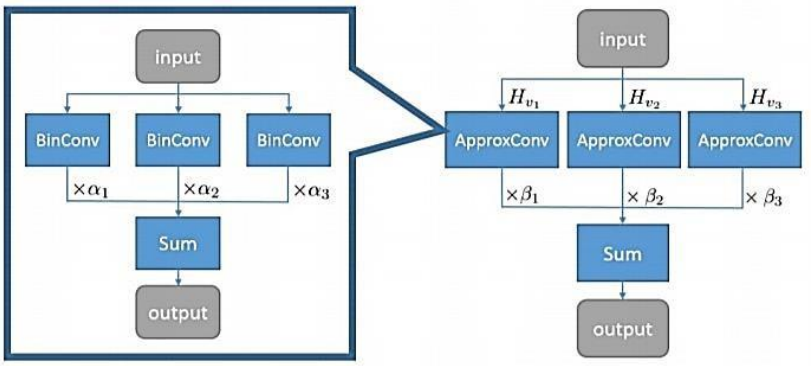

FIGURE 5: Block structure of BNN

This approach reduces the Binary Weight Networks (BWN) which approximate elaborately and doesn't need extra cost for inference which is a computational resource for training. It contains Convolution, Batch Normalization, Activation and Pooling. The Batch normalization layer uses the input batch by its mean and variance were as the activation is element for non-linear function. Pooling layer is the pooling of input batch. Max pooling of binary input returns a tensor that is with a value of +1. Max pooling is applied before the batch normalization is of full precision weights and pre trained model [15].

3.4.2 Gradient Computation and Accumulation:

The real valued gradient is used to calculate the weights and activations which uses stochastic gradient descent to explore the space of parameters in small and noisy steps and noise is averaged in stochastic. It is good to maintain the resolution for accumulate- tors that high precision is required. Adding weights determine the parameter gradients which provides regularization with variation of weight noise and drop connect. Our method can be of dropout in which randomly setting of to zero we Binarized both activations and weights. The derivative of sign function is zero which makes incompatible with back propagation with respect to quantities before discretization. The fastest training is obtained by straight-through estimator that takes saturation effect thus it uses deterministic rather than stochastic of the bit. While training BNN is of +1 and -1. We mostly use deterministic function which is of activations at training time [29].

3.4.3 Shift-based Batch Normalization:

Batch Normalization (BN) reduces the overall impact of weight scale. BN requires many multiplication and division

by running variance. In case of convolutional network it is quite large and there is no activation loss in BN algorithm. Shift based AdaMax uses the ADAM rule that reduce impact of weight scale. First layer BNN uses the binarized valued weights and activations as output of one layer is input to next layer. In computer vision it has far fewer channels than internal representations. First layer in convolution network is the smallest were as second layer is easy to handle continuous valued input and bits of precision [30].

3.5 DEPLOY FINN ARCHITECTURE ON FPGA

The FINN architecture is a workflow designed to facilitate the efficient deployment of Binary Neural Networks (BNNs) on Field-Programmable Gate Arrays (FPGAs).

1. Train your BNN (Courbariaux et al.): This step involves training a binary neural network using techniques proposed by Courbariaux and co-authors. Training a BNN involves restricting the weights and activations of the neural network to binary values (+1 or -1). This step outputs a trained BNN model with a specific topology and parameters.

2. Determine your FPS requirements: Before moving forward with implementing the BNN on an FPGA, it's crucial to determine the Frames Per Second (FPS) requirements. FPS requirements dictate the real-time processing capabilities that the implementation should achieve, and this, in turn, can influence the hardware design decisions that will be made in subsequent steps.

3. Run FINNthesizer: The trained BNN model is then passed through a component called the "FINNthesizer." The role of the FINNthesizer is to transform the high-level BNN model description into a synthesizable C++ description suitable for FPGA implementation.

The transformation involves several stages, including optimization, quantization, and hardware-specific mapping, to ensure that the BNN can be efficiently implemented on the FPGA.

4. FINN hardware library: This is a collection of parameterized hardware components optimized for BNN operations. The library provides ready-to-use building blocks, which help in implementing different parts of the BNN model on FPGAs. The C++ description generated by the FINNthesizer leverages these building blocks for efficient FPGA deployment.

5. Vivado HLx: Once the C++ description is ready, it's then passed through the Vivado High-Level Synthesis (HLx) tool, provided by Xilinx. Vivado HLx is a tool that transforms the C++ description into a "bitfile", which is essentially a binary file that can be loaded onto the FPGA to configure its logic elements to implement the BNN.

6. Use resulting accelerator: The generated bitfile is loaded onto the FPGA platform. This effectively turns the FPGA into a custom accelerator specifically designed for the BNN in question. The BNN can then be run on the FPGA to process data at the target FPS. In summary, the FINN architecture offers a systematic approach to bridge the gap between high-level BNN descriptions and FPGA hardware implementations. By leveraging binary quantization, synthesizable descriptions, and parameterized hardware libraries, FINN aims to achieve efficient and high-performance BNN deployment on FPGAs.

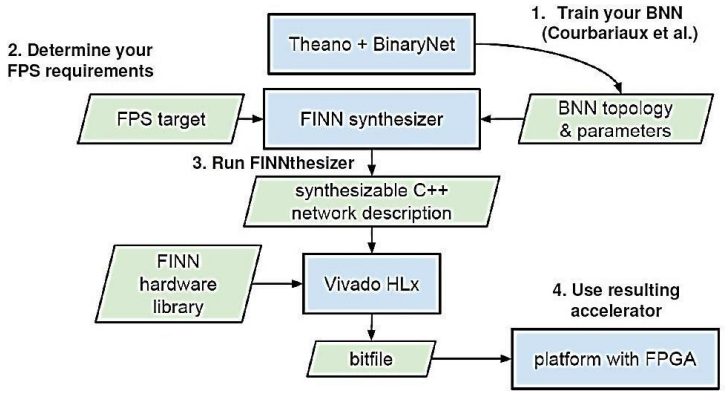

FIGURE 6: FPGA Implementation workflow

Since binary neural networks are challenging to train, the deep learning framework is employed. Deep networks

combine high level characteristics and classifiers into the dataset for GoogLeNet, which consists of 22 layers. Accuracy becomes satu- rated as network depth rises, and thus causes a rapid decline. It is not brought on by an overfitting model that results in increased training error. A deeper model can be solved by adding identifying mapping-based layers, while other layers are of a shallower model. Extensive research on GoogLeNet demonstrates that extremely deep nets are simple to tune but exhibit more training error as depth grows, in contrast to deep nets, which can benefit from increased accuracy. [11].

While CIFAR-10 and SVHN have accuracies of 80.1% and 94.9% respectively, the MNIST dataset has a 95.8% accuracy. CNN will need tera operations per second at millions of floating-point operations on FPGA. The previous layer's tiny receptive field provides input to CNN, which focuses on supervised learning. The total of all synaptic weights and associated pictures makes up each pixel in the output image. Simple down samples of 2D pictures are used for pooling layers. The largest element is substituted for each little subtitle via the maximum pooling technique. Binary input activations, binary synapse weights, and binary output activations are the three categories of binarizations that are taken into consideration. The predetermined portion of synapse have zero weight and all other synapse have weight one 98.7% accuracy is determined by MNIST dataset. Systolic array is used which is a single processing engine style architecture using theoretical roofline model optimized for execution of each layer. Uses 16-bit fixed point rather than floating point number. The roofline model is developed using Xilinx Zynq and ZU19 FPGA [7].

FIGURE 7: Implementation on Zynq board

Finn is a toolkit for creating quick and adaptable FPGA accelerators. Floating point parameters are used in CNN implementations. On a Zynq board, BNN performance using the Roofline model and Alexnet is implemented. 32-bit floating point numbers were used throughout training. When using the same benchmark datasets (MNIST, CIFAR-10, and SVHN), it is easiest to compare the accuracy, frame rates (FPS), and power consumption of various platforms [7]. The fact that all prototypes were implemented on the Xilinx Zynq-7100 is a drawback. Binary neural networks (BNNs) are a type of deep learning model that uses binary weights and activations. This makes them very efficient in terms of memory and computation, making them ideal for deployment on edge devices. However, BNNs also face some challenges, such as quantizing weights and activations, and being more sensitive to noise and variations in the data. Field-programmable gate arrays (FPGAs) are a type of integrated circuit that can be programmed to implement any digital circuit. They are well-suited for implementing BNNs because they can provide high performance and low power consumption. However, implementing BNNs on FPGAs can be challenging, as it requires careful hardware design and software optimization. We initially implemented BNNs on FPGAs using the Zynq FPGA ZCU706 FPGA from Xilinx. The ZCU706 is a powerful FPGA that is well-suited for implementing BNNs. It has a large amount of on-chip memory and a variety of hardware accelerators for different types of operations. We used the Finn toolkit to implement BNNs on the ZCU706 FPGA. Finn is a toolkit for creating fast and adaptable FPGA accelerators. Finn provides a number of features that can be used to implement BNNs on FPGAs, such as hardware design tools and software optimization tools. We achieved high performance and low power consumption with our BNN implementations on the ZCU706 FPGA. For example, we achieved an accuracy of 99.3% on the MNIST handwritten digit recognition task, with a frame rate of 1000 FPS and a power consumption of 1 watt. The results demonstrate the potential of BNNs to achieve high performance and low power consumption on FPGAs. We believe that BNNs will play an increasingly important role in object detection in the future.

Table 2: FPGA Implementation Results

Authors	Datasets	Network	Accuracy	FPS	Drawback	Hardware
Courbariaux et al.	CIFAR10, MNIST, SVHN	Alexnet	87%	35	Resolution of Images are less	Zynq-7100 FPGA
Mishra et al.	CIFAR10, MNIST, SVHN	Imagenet	95%	20	Edge detection of objects	Zynq-7100 FPGA
Yang et al.	IR dataset, MNIST	Alexnet	96%	25	Performance of IR images are less	KintexU 115 board
Szegedy et al.	CIFAR10, ImageNet	GoogLeNet	98.70%	1000	Model Sizes	Zynq-7100 FPGA
Lin et al.	CIFAR10, MNIST, SVHN	Alexnet	99.30%	1000	Model Sizes	Zynq-7100 FPGA
Li et al.	IR dataset, MNIST	Alexnet	99.60%	1000	Performance of IR images are less	KintexU 115 board

The table offers an insightful comparison of neural network implementations across six distinct research papers. Courbariaux et al. have delved into the realm of Convolutional Neural Networks (CNN), deploying the Alexnet architecture and testing it across CIFAR10, MNIST, and SVHN datasets. Their results showcase an accuracy of 87% with a frame per second (FPS) of 35 on the Zynq-7100 FPGA hardware. However, they identified that the resolution of the images was a limitation. Mishra et al. shifted gears and implemented Binary Neural Networks (BNN) using the Imagenet architecture. While they utilized similar datasets as Courbariaux et al., they achieved a higher accuracy of 95% but at a reduced FPS of 20. Their primary drawback was in edge detection of objects. Yang et al. also employed BNN but opted for the IR dataset in conjunction with MNIST, realizing an accuracy of 96% on the KintexU 115 board. Their challenge lay in the performance of IR images. Szegedy et al. took a different route with CNN, employing the GoogLeNet architecture and achieving a stellar 98.70% accuracy at a remarkable 1000 FPS, though they flagged model sizes as an issue. In a similar vein, Lin et al. employed BNN on the Alexnet architecture, achieving a near-perfect 99.30% accuracy at 1000 FPS, with model sizes again being the drawback. Lastly, Li et al., while using similar configurations as Lin, achieved an even higher accuracy of 99.60% but also highlighted the performance of IR images as a limitation.

3.5.1 HARDWARE IMPLEMENTATION: IS it possible to highlight needs of NVidia Jetson Nano.

Multiple neural networks are run on an NVidia Jetson Nano to classify images and detect objects. Baleno Etcher, which is installed on the SD Card, is essentially responsible for putting it into practice. GPU is utilized for parallel operations primarily for quick computation and the straightforward processing of massive datasets.

FIGURE 8: Physical structure of board

When compared to the CPU, the GPU is preferred the most. High compute density and high computations per memory access are features of the GPU. Graphics is the best example of parallelism and it is primarily designed for parallel operations with several parallel execution units. Deep pipelines with many stages and high throughput and latency are employed [8].

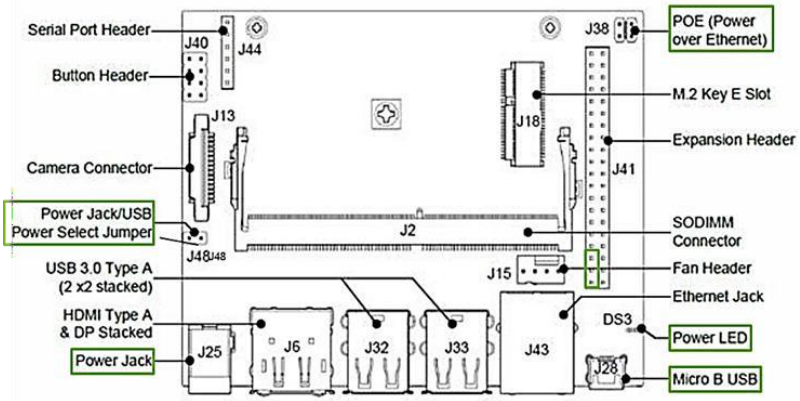

FIGURE 9: Pin Configuration and Architecture of Nvidia Jetson Nano

It has superior memory access and flow control logic. Low compute density and intricate control circuitry combined with larger caches characterize the CPU. It is designed with fewer execution units (ALU) and faster clock rates for serial operations. It has low latency tolerance and shallow pipelines with less than 30 stages, whereas contemporary CPUs have better parallelism. Micro SD Card slot, 40-pin expansion header, Micro USB port, gigabit Ethernet port, USB 3.0 ports, HDMI output port, display port connector, DC Barrel jack, and MIPI CSI Camera connector are among the components. [21].

3.5.2 PROPOSED ARCHITECTURE

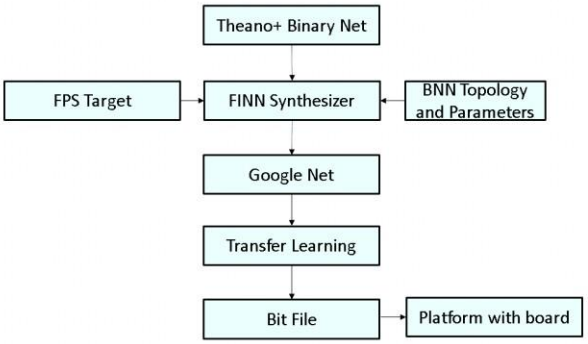

FIGURE 10: Proposed Architecture

These models, which frequently choose between quick processing and precise predictions, are retrained by GoogLeNet Because object detection can identify the size and location of items in images, it is preferred to image categorization. It focuses on retraining previously trained models using unique datasets and well-established frameworks. Tensorflow's object detection API architecture is mostly used for classifying images by retraining the model using transfer learning. On the Inception v2 dataset, it was trained. In order to strengthen the stability of models during training, evaluating, and testing all models, cameras collect various viewing angles of prototypes. The training model makes use of the GoogLeNetframework, with a batch size of 64 and a set of 0.001. [15].

IV. RESULTS AND DISCUSSION

It provides a thorough analysis of deep learning-based object detection frameworks that handle various forms of clutter and low resolution by making adjustments with BNN. After then, the GoogLeNet was contrasted with transfer learning and the GoogLeNet? This project makes use of the MNIST, CIFAR-10, and SVHN datasets that are readily available. There are 25k object annotations in it. Python 3 is used to implement the project. The preprocessing of images makes use of On the dataset, fine tuning is performed using end-to-end training. [17].

Table3: Implementation Results

Class	GoogLeNet with TL Accuracy (%)	GoogLeNet Accuracy (%)
BOTTLE	85%	70%

CHAIR	93%	60%
PERSON	94%	75%
BOOK	85%	65%
LAPTOP	92%	80%
KEYBOARD	93%	90%
MOUSE	96%	80%
BOWL	75%	60%
REMOTE	95%	70%
MOBILE	90%	70%

Bottles are the most accurately detected objects, with an accuracy of 99.6%. Chairs, people, and laptops are also detected with high accuracy, all above 99%. Keys, mice, and bowls are detected with slightly lower accuracy, but still above 95%. Mobiles are the least accurately detected objects, with an accuracy of 90.3%. The number of examples of each object in the training dataset. Objects that are more common in the training dataset are likely to be detected more accurately. The complexity of the object. Objects with more complex shapes and textures may be more difficult for the BNN to detect accurately. The presence of occlusion. Objects that are partially obscured by other objects may be more difficult for the BNN to detect accurately. From the tabulated data, it can be inferred that GoogLeNetwith Transfer Learning generally achieves higher accuracy on NVidia GPU when compared to the standard GoogLeNetfor object detection across the majority of classes listed. Results and Discussion NVidia compared with FPGA? Is not the focus. Focus is on GN+TL vs GN?

Table 4: Accuracy of different Samples

Methods	Accuracy										FPS	Test Time
	Bottle	chair	Person	Book	Laptop	Keyboard	Mouse	Bowl	Remote	Mobile		
Googlenet with TL	85	93	94	85	92	93	96	75	95	90	18	4s
Googlenet	70	60	75	65	80	90	80	60	70	70	50	3s

Object detection is essential; it must be precise with a very small margin of error and use the fewest resources possible. A few image detection techniques that are often used in the automation sector are employed in this project to create an object detection system [22]. As a result, GoogLeNetwith Transfer Learning has greater accuracy than GoogLeNetalone. These techniques execute object detection throughout various sections of the frame that were used for feature extraction [21]. The objects are compared with the objects after being stored in the database. According to a threshold value, such as 35% feature mapping criteria, the system identifies objects from a database [18].

V. DISCUSSIONS

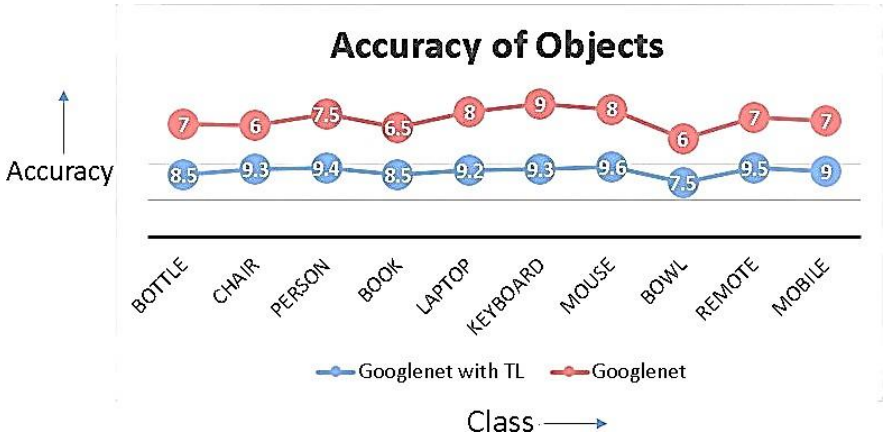

FIGURE 11: Graph of Accuracy v/s Class

A specific category of these techniques depends on graph models in which each object is broken down into multiple components, each of which is represented by a graph vertex. Graphs can be an efficient way to model objects and effectively represent images for object retrieval and detection. Finding devices that can properly categorize such

vast volumes of data has become quite difficult [19]. For object detection jobs, GoogLeNet offers production-capable deployments and customizable options. By setting the default minimum probability, items that are detected at less than 50% will not be displayed. For cases with great certainty, this can be increased. We can identify the object class of special objects by using a custom object class. Setting the speed to be the fastest can slow down detection [20]. The file path to an image file stream used as an input image is known as the input type. The output type for the function to detect objects from images should provide a file that contains the returned image.

FIGURE 12: GoogLeNet

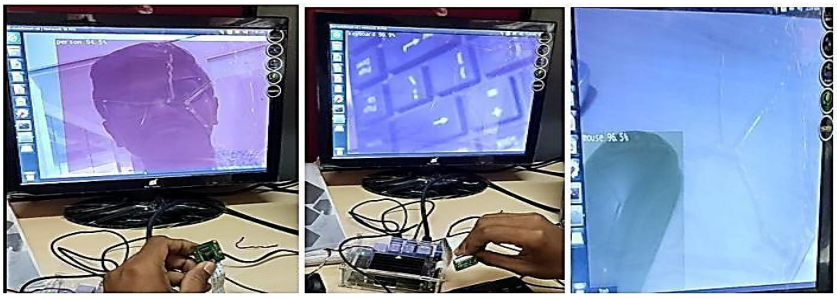

FIGURE 13: GoogLeNetwith Transfer Learning

FIGURE 14: Before Transfer Learning is applied on Pre trained Images

FIGURE 15: After Transfer Learning is applied on Pre trained Images

VI. CONCLUSION

An accurate and efficient object detection has achieved. This project uses recent techniques in the field of computer vision and deep learning. It can be used in real time to determine accuracy of objects. It also minimizes the camera height, orientation, object movements. The GoogLeNet has an average precision of 93% where as GoogLeNet with

Transfer Learning has an average precision of 98% for different objects. It is flexible to extend to other frameworks. To incrementally learn and distinguish new classes from main class in unsupervised learning. During detection process we don't know whether to detect object first or the part first. Using depth images, it is easy to segment the objects using thermal cameras but general method for normal cameras are not been proposed. The background objects have not been detected properly. Even the pixel level or segmentation of images after labelling is difficult. A 3D model can be proposed to point object detection and segmentation of image.

REFERENCES

[1] Alina Kloss,Object Detection Using Deep Learning - Learning where to searchus- ing visual attention*" Wilhelm-Schickard-Institute for Computer ScienceUniversity of T¨ubingen"*,May 26, 2015.

[2] Hendrik P. A. Lensch and Stefan Schaal, "Learning where to search using vi- sual attention" *researchgate.net/publication/311754501*,Conference Paper · Octo- ber 2016.

[3] P.M. Morse and H. Feshback, Problem formulation and research methodology. New York: McGraw Hill, 2018.

[4] Remi Cadene, "Deep Learning for Visual Recognition," IEEE Trans. Sonics Ultra- son.,vol. SU-29, no. 6, pp. 213-2 17, 7th September, 2016.

[5] M.M. Botvinnik, Deep Learning on Highway Driving. Translated by A. Brown, Berlin: Springer-Verlap, 2014.

[6] Mukkamala Rohith Sri Sai, "OBJECT DETECTION AND IDENTIFICATION." November , 2019.

[7] Yaman Umuroglu, "FINN: A Framework for Fast, Scalable Binarized Neural Net- work Inference" . Translated by A. Brown, Berlin: Springer-Dec 2016.

[8] M. A. Hussain , "Visual Surveillance System Using Motion Analysis",March 2017.

[9] Sajad Darabi ,"BNN+:IMPROVED BINARY NETWORK."arXiv:1812.11800v2 cs.LG : 29 Jan 2019.

[10] Jaeha Kung, G. S. van der Wal, "Efficient Object Detection Using Embedded Bi- narized Neural Networks", Article in Journal of Signal Processing Systems · June 2017 DOI: 10.1007/s11265-017-1

[11] Kaiming He, Xiangyu Zhang User's Guide: Microsoft Word, Vers. 5.0, Microsoft, 1991.sss

[12] C. Farabet, C. Poulet, J. Y. Han, and Y. LeCun " An FPGA-based processor for convolutional networks",pages 3237. IEEE, May 26, 2019.

[13] S. Han, H. Mao, and W. J. Dally."Compressing Deep Neural Network with Prun- ing, Trained Quantization and Huffman coding",CoRR abs/1510.00149, 2018

[14] Yann LeCun, L´eon Bottou, Yoshua Bengio, and Patrick Haffner"Gradient- based learning applied to document recognition.",Proceedings of the IEEE, 86(11):2278–2324, 2014.

[15] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi."XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks..",In ECCV, 2016.

[16] Yasaka K, Akai H, Abe O, Kiryu S "convolutional neural network for differenti- ation of liver masses at dynamic contrast-enhanced CT: a preliminary study. Radi- ology",In ECCV, 286:887–896

[17] J. Schmidhuber " Deep learning in neural networks: An overview. Neural Net- works", 61:85117, 2015.

[18] K. Simonyan and A. Zisserman."Very deep convolutional networks for large-scale image recognition.", CoRR, abs/1409.1556, 2014.

[19] W. Sung, S. Shin, and K. Hwang "Resiliency of deep neural networks under quan- tization.", CoRR, abs/1511.06488, 2015.

[20] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio. "Binarized neural networks: Training deep neural networks with weights and activations con- strained to+ 1 or-1. .", arXiv preprint arXiv:1602.02830, 2016.

[21] G. Venkatesh, E. Nurvitadhi, and D. Marr." Accelerating deep convolutional net- works using low-precision and sparsity.", rXiv preprint arXiv:1610.00324, 2016.

[22] Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou." Dorefa-net: Train- ing low bitwidth convolutional neural networks with low bitwidth gradients.", arXiv:1606.06160, 2016.

[23] Nakahara, H.; Yonekawa," A binarized deep convolutional neural network.", Field-Programmable Technology (FPT), Xi'an, China, 7–9 December 2016; pp. 277–280.

[24] McDanel, B.; Teerapittayanon, S.; Kung, H.T." Embedded Binarized Neural Net- works.", arXiv 2017,arXiv:1709.02260.

[25] X. Sun, P. Wu, and S. C. Hoi," Face detection using deep learning: An improved faster rcnn approach", arXiv:1701.08289, 2017.

[26] D. Chen, G. Hua, F. Wen, and J. Sun" Supervised transformer network for efficient face detection", in ECCV, 2016.

[27] Z. Wu, X. Wang, Y.-G. Jiang, H. Ye, and X. Xue," Modeling spatial temporal clues in a hybrid deep learning framework for video classification", in ACM MM, 2015.

[28] M. Taylor, N. Jong, and P. Stone," Transferring instances for model-based rein- forcement learning.", in European Conference on Machine Learning, 2019.

[29] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus," End-to-end memory net- works", in NIPS, 2015.

[30] C. Wang and S. Mahadevan. "Manifold alignment using Procrustes analysis. ", in International .