Chapter Title: Overview of Artificial Intelligence on Edge Devices.

Subtopics: Machine Learning & Deep Learning Applications, Internet of Things

1) Introduction:

Artificial Intelligence and Machine Learning models have been in development for a long time. As the demand for AI to solve complex problems increase the problem of high-performance computing also increases because developing AI requires significantly higher hardware capacities and use of external accelerators. This chapter aims to create a comprehensive and comparative outlook on the development of Artificial Intelligence on different Edge Devices with the outlook of edge computing based on their hardware architecture and their ability to develop an AI model. The comparison will be based on facts and some personal experiments. The chapter will also aim to look on different problem statements like object detection and try to test the same on some of the hardware categories and determine the performance of the model and draw a conclusion on which hardware will perform best for developing certain AI models and which will create optimal results. The study will be a review of different resources available on the topic with a touch of some personal conclusions drawn through Raspberry Pi 3 and Raspberry Pi 4.

Starting with Edge devices, Edge devices refer to a category of computing devices that are located close to the data source or the user, typically at the network's edge, rather than relying on centralized cloud-based computing. These devices are equipped with processing power and storage capabilities that allow them to perform data processing, analysis, and storage locally, which reduces the need for constant communication with remote data centers.

Common examples of Edge Devices can be found in regular places from smartphones to Surveillance devices such as CCTV's. Edge devices can also be categorized using their usage in the field:

- Smartphones and Tablets: These portable devices have significant processing power and are often used for local data processing and running various applications without relying solely on cloud services.

- Internet of Things (IoT) Devices: Connected devices such as sensors, cameras, smart appliances, and industrial machinery are often equipped with edge computing capabilities to process data locally before sending relevant information to the cloud.
- Edge Servers and Gateways: These devices act as intermediaries between the edge devices and the central cloud infrastructure, aggregating and preprocessing data from multiple edge devices before transmitting it to the cloud.
- Routers and Switches: Some modern networking equipment includes edge computing capabilities to perform functions like traffic optimization, security filtering, and network monitoring locally.
- Embedded Systems: Devices with embedded processors, such as in-car entertainment systems, medical devices, and industrial controllers, often have edge computing capabilities to handle specific tasks without relying on a constant internet connection.

2) Why is there a need for edge devices?

The need for edge devices is accountable to several factors from a storage and computing perspective to more security and privacy enhancement:

- Reduced latency: Data processing occurs closer to the data source, which can significantly reduce latency. This is important for applications that require real-time or near-real-time response, such as self-driving cars, industrial automation, and augmented reality.
- Bandwidth optimization: By processing data locally, edge devices can reduce the amount of data that needs to be transmitted over the network, optimizing bandwidth usage, and lowering associated costs.
- Data privacy and security: Keeping sensitive data locally on edge devices can enhance data privacy and security, as it reduces the exposure to potential data breaches during data transit to centralized servers.
- Offline capability: Some edge devices can continue to operate and provide essential functionality even when disconnected from the central network or cloud infrastructure. This is valuable in scenarios with intermittent or unreliable connectivity.

3) Artificial Intelligence Tasks that can be employed on Edge Devices:

Artificial Intelligence (AI) tasks that can be employed on edge devices leverage the capabilities of localized processing and real-time inference. By performing AI tasks directly on edge devices, various benefits such as reduced latency, improved privacy, and enhanced reliability can be achieved. Here are some AI tasks that can be deployed on edge devices:

Image and Video Processing:

- Object detection and recognition: Identifying and classifying objects within images or video streams in real-time.
- Facial recognition: Authenticating or identifying individuals based on facial features.
- Gesture recognition: Detecting and interpreting hand gestures from images or video.
- Image and video compression: Reducing the size of media files to optimize storage and transmission.

Natural Language Processing (NLP):

- Speech recognition: Converting spoken language into text.
- Speech synthesis: Generating human-like speech from text.
- Language translation: Translating text or speech from one language to another.
- Sentiment analysis: Determining the sentiment (positive, negative, neutral) of text data.

Anomaly Detection:

- Identifying unusual patterns or behaviours in data from sensors or monitoring systems.
- Predictive maintenance: Predicting potential failures or maintenance needs in industrial machinery or equipment.

Edge Machine Learning Models:

- Running pre-trained machine learning models locally for inference tasks.
- Transfer learning: Fine-tuning existing models on edge devices for domain-specific tasks.
- Federated learning: Collaborative learning across multiple edge devices without sharing raw data.

4) Real Time Use case performed using Raspberry Pi 3 and Raspberry Pi 4:

For understanding the concept of AI on Edge devices it is essential to know a little about the hardware of most widely used Edge devices like Raspberry Pi 3 and Raspberry Pi 4.

The latest Raspberry Pi 3 B+ has a 64-bit quad-core ARM Cortex-A53 @ 1.4 GHz processor with a RAM of 1 GB LPDDR2-900 SDRAM and can run the Raspbian OS with the availability of programming languages such as Python, C/C++, and Java. Though the Pi 3 does not have a very powerful CPU but is able to handle lightweight object detection models like EfficientDet lite and YOLO V3.

Raspberry Pi 4 has a 64-bit quad-core ARM Cortex-A72 @ 1.8 GHz processor and comes in different RAM variants such as 1 GB, 2 GB, 4 GB or 8 GB LPDDR4-3200 SDRAM, this also comes with a GPU and performs faster on Mid to High level object detection algorithms like Faster RCNN and YOLO v7.

The reason Raspberry Pi is widely used is the ease to use and setup the devices according to your use and, they have large community of users and developers who can help you with your projects.

For this use case an Object detection model was created to be fitted with both the models Pi 3 and Pi 4 and check their performance based on the FPS and accuracy they provide on detection.

To deploy a Deep Learning model on edge devices it is proposed to use the TensorFlow Lite library as it has already pretrained models and weights:

4.1 TensorFlow Lite

TensorFlow Lite is a deep learning framework that is open-source and can be used on multiple platforms. It takes pre-trained models from TensorFlow and converts them into a format that is optimized for either speed or storage.

This specialized format can then be used on edge devices such as Android or iOS mobile phones, as well as embedded devices such as Raspberry Pi or Microcontrollers to enable inference at the edge.

In this proposed model we are performing the task of object detection, The first thing is to decide the model for this task. There are different options for that:

- Creating a custom model.
- Using a pre-trained model like YOLO, CNN, ImageNet, EfficientDet.
- Applying Transfer Learning on a pre-trained model.

The TF Lite model is optimized for accuracy and is a lightweight version, which makes it an ideal option for mobile and embedded devices in this case Raspberry Pi where space is limited. Its efficiency and compactness make it a perfect fit for such devices.

4.2 Object Detection Model:

For this chapter a latest object detection model is used called Efficient Det lite which was developed by google to improve the performance of edge computing.

EfficientDet Lite:

The proposed model uses EfficientDet lite as its model for object detection. EfficientDet lite was proposed by google in 2020 in their paper titled "EfficientDet: Scalable and Efficient Object Detection".

Like any Object detection model EfficientDet also has three components: Backbone, Feature Extraction, Detection Head.

1) Backbone:

The work of a backbone in any deep learning model is to extract features from the given images. EfficientDet uses EfficientNet as its backbone which is another model by Google. Using EfficientNet as backbone the model achieves much better efficiency than previous detectors like ResNets, VGG16, ImageNet, etc.

EfficientNet works by scaling the Depth, Width and Resolution. This helps the model to extract more crisp features from the image and better features provide better detection results.

2) Feature Network:

A Feature Network is the second step in object detection, it combines all the incoming feature and outputs feature maps at multiple level. Most of the object detection models use some kind of Feature Pyramid Network(FPN) for this task but the problem with traditional FPN is it is time consuming and requires high memory for training and implementing.

So, to reduce this problem EfficientDet uses a Bi-directional Feature Pyramid Network (BiFPN). A weighted BiFPN is a type of feature pyramid network which performs multi-scale feature fusion at fast speeds and with an ease. It uses top-down and bottom-up approach with regular and efficient approach.

Traditional approach gives all the features equal importance even if they are of different resolutions. However different resolution features lead to contribution of unimportant features in the output layer.

Therefore, BiFPN adds additional weight to individual inputs allowing the network to learn importance of each feature which in sense provides the model with more refined features.

3) Detection Head:

Like any object detection model the EfficientDet uses a detection head to represent the output. EfficientDet uses the same detection head as its predecessors.

Detection head predicts the class of the object and the bounding box required, EfficientDet uses a regressor and classifier. The regressor is used for drawing a bounding box around the object and the classifier uses a SoftMax function for predicting and displaying the class.

4.3 Dataset Employed:

The dataset used here is a combination of different animal classes such as Bear, Elephant, Hyena, Leopard, Lion, Pig, Tiger, and Wolf.

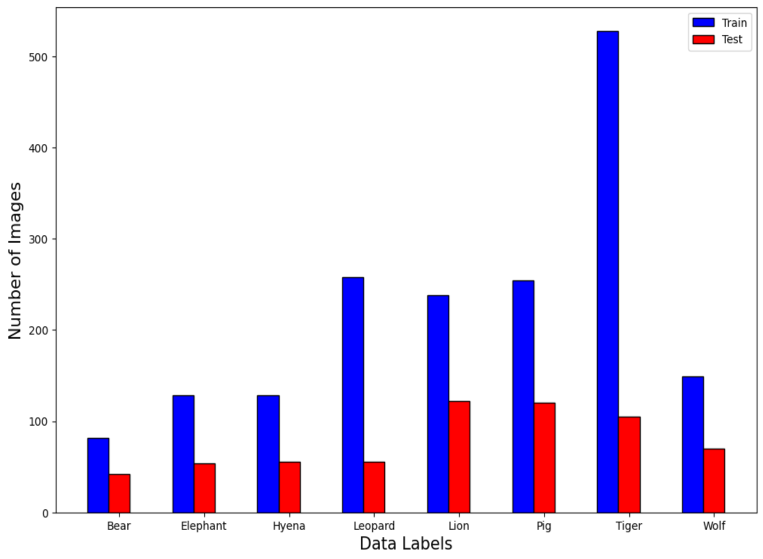


Fig. 1. Dataset representation with train and test split numbers.

4.4 Model Training and Deployment

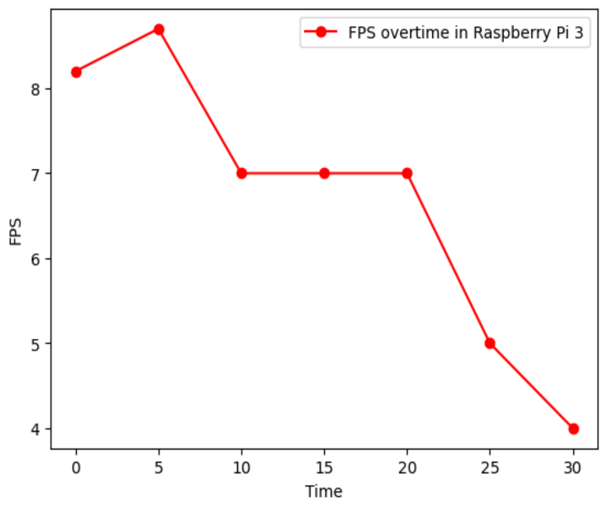
Model Training for edge devices can be performed on Google Colab which overcomes the performance problem of the edge devices with cloud GPU but simultaneously training can also be done using the devices computing power but it will require greater time to handle big datasets

For deployment Raspberry Pi with the pi camera module is the main detection module. To perform detection the raspberry pi command prompt is opened in the TensorFlow lite environment where the program is stored and to run the program the user needs to input the commands same as a windows terminal.

Once the setup is complete, the camera starts working on the live video feed and detects the objects around and matches it with the trained model like any other object detection model.

5) Results:

Apart from the accuracy of different classes which is a factor that depends on the dataset quality and training period, to compare the edge devices Raspberry Pi 3 and Raspberry Pi 4 we can consider the FPS (frame per second) parameter to compare the behaviour of these two edge devices.



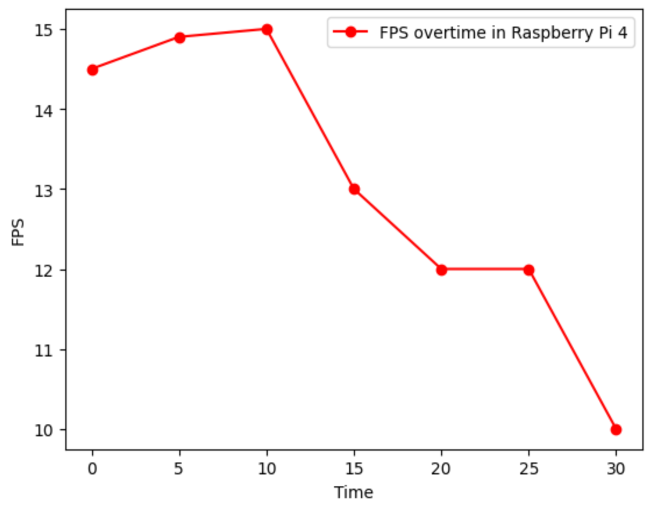


Fig. 2. *Fig 3.*

FPS given by Raspberry Pi 3 and Raspberry Pi 4 overtime in minutes.

- From simple graphs in Fig 2 and Fig 3, one can understand that Raspberry Pi 4 is giving greater FPS than Pi 3 even at its lowest.
- Let's explore the result more, Raspberry Pi 3 starts with great FPS of 8 and continues to give constant FPS of 7 for a long time but later drops to 4 after 30 minutes due to the heating of the components.
- Raspberry Pi 3 in no way gives the performance as we see in a normal PC or laptop, but for its size it performs satisfactorily and gives understandable FPS at beginning.
- Coming to Raspberry Pi 4 the device starts at a great FPS of around 14 to 15 and even touches 15 after sometime but after that there is a gradual decline to 10 at the end of the testing cycle.
- Unlike Pi 3 which maintains a constant FPS for 3 test points, Pi 4 maintains only maintains a constant FPS for 2 test points.
- If we compare these edge devices to traditional hardware's like PC and laptops with GPU's the results are disappointing since 24 FPS are normal for any good PC and laptops.
- But the accessibility of these edge devices gives them an advantage and increase use cases foe similar kind of tasks.

6) Conclusion:

Edge Devices are still at their growing stage and are getting more powerful with external accelerators like Coral Edge TPU developed by google or FPGA boards which are developed by intel.

Edge devices are good for small tasks like object detection on a controlled dataset and provide accurate and understandable results but they still have a long way to go in comparison to the traditional computers.

A conclusion can be drawn that edge devices are easy to use and even more easy to program with the preferred tasks but they will not produce results like their traditional computing counterparts due to their size and processing power. The results obtained from Edge devices are satisfactory and since they are smaller in size and more accessible they also increase the use cases of Artificial Intelligence.