

INDUSTRIAL TRAINING REPORT

IN

RASPBERRY PI WITH PYTHON

**For project on
IMAGE SORTING ON RASPBERRY PI NAS USING MACHINE
LEARNING**

**Undertaken at
National Institute of Electronics & Information Technology**



**Under the Guidance of
Dr. SARWAN SINGH
(Deputy Director, NIELIT)**

ACKNOWLEDGEMENT

The success and final outcome of this project required a lot of guidance and assistance from many people and we are extremely privileged to have got this all through the completion of our project. All that we have done is only due to such supervision and assistance and we would not forget to thank them. We are thankful and fortunate enough to get a chance for undergoing training at National Institute of Electronics & Information Technology.

We respect and thank Dr. Sarwan Singh for providing us an opportunity to do the project work in National Institute of Electronics & Information Technology and giving us all support and guidance which made us complete the project duly. We are extremely thankful to him for providing such consistent support and guidance. Also, I would like to extend my sincere esteems to Mr. Jaspreet Singh who took keen interest in my project work and for his timely support.

We, as a team are fortunate enough to have such intriguing team members. We hail from different backgrounds, different part of countries and different colleges. We all are lucky enough to work as a team and efforts of each and every one counted for the completion of this project.

ABSTRACT

What is innovation? A simple Google search tells, “A product or an idea becomes **innovative** when it stands out from the rest and truly **makes** the customers' lives easier. A successful change that can convert knowledge and ideas into benefit – in the form of new or improved products / services is capable of being **innovative**.”

This was the thought process of our team. We wanted to do something innovative which can be beneficial to all and a project which can be modified for personal needs or can be extended further by someone else in future.

In our project we have tried to exploit the features of Raspberry Pi. The most fascinating part of Pi is that, it's a full fledged CPU with a complete Linux based operating system. To put it in simple words, we can potentially do everything that we do on a normal day-to-day computer on a Raspberry Pi. The only drawback is it's comparatively very less processing power. But when considering its size, power consumption and over-all cost value, it becomes perfect for most of the application where we might have needed a CPU. A Pi can be compared to 10-20 years old computers and in that comparison Raspberry Pi will emerge as a winner. There are many other Single Board Computers (SBCs) in the market and what makes Pi stand out from them is its ever growing community and its features per cost.

The title of our project is “Image Sorting on Raspberry Pi NAS Using Machine Learning”, the title itself tells much about the project work. We took a Pi, installed raspbian on it via NOOBS. Then we choose Open Media Vault to create a NAS on Pi using our regular pen drive. A hard-drive can also be connected and it would be advisable to use hard-drive which has a external power source and doesn't draws the power from the Pi.

In next step, we choose a pre-trained model, GoogleNet, it is quite famous among image recognition enthusiast and is based on Caffe framework. The tricky part was to implement it and to modify it to our requirements. After extensive modification, we were able to successfully run it on our Pi.

While making this project again, one should remember to use the correct versions of the Python and its packages. This is to ensure that there are no compatibility issues. All the necessary codes and the steps to follow are mentioned in the report. We have also mentioned the source for different parts of project which can be viewed later by the reader for deep understanding. The process and code mentioned in the report has been tested on three Raspberry Pi model B with (raspbian stretch) and have undergone necessary modifications.

Finally we can uploa, the image from mobile to the NAS and then the program written in Python 3 will sort the images in the respective folders. Currently we have limited ourselves to five categories viz: Vehicle, Animal, Food, Nature, and Miscellaneous. After this project one should be able to try and implement any other pre-trained model on Raspberry Pi.

TABLE OF CONTENTS

ABOUT NIELIT	7
<hr/>	
1 EVOLUTION OF SINGLE BOARD COMPUTERS	9-15
1.1 A BRIEF HISTORY	
1.2 PRESENT DAY SBCs	
1.3 TYPE OF SBCs	
1.4 APPLICATIONS	
1.5 THE FUTURE OF SBCs	
<hr/>	
2 KNOW YOUR RASPBERRY PI	15-21
2.1 WHAT IS A RASPBERRY PI?	
2.2 WHO INVENTED RASPBERRY PI?	
2.3 WHY IS IT CALLED RASPBERRY PI?	
2.4 WHEN WAS RASPBERRY PI LAUNCHED?	
2.5 WHY TO CHOOSE RASPBERRY PI?	
2.6 HOW IS IT SO CHEAP?	
2.7 WHAT IS DIFFERENCE BETWEEN RASPBERRY PI'S MODELS?	
2.8 WHERE ARE RASPBERRY PI'S USED?	
2.9 WHAT IS A RASPBERRY PI GOOD FOR?	
2.10 HOW MANY RASPBERRY PI HAVE BEEN SOLD?	
2.11 WHERE CAN YOU BUY RASPBERRY PI?	
<hr/>	
3 GETTING STARTED WITH RASPBERRY PI	22-24
3.1 HARDWARE SPECIFICATIONS	
3.2 INSTALLATION OF AN OPERATIONG SYSTEM	
<hr/>	
4 SETTING UP A NAS ON A RASPBERRY PI USING OMV	25-32
4.1 WHAT IS OPEN MEDIA VAULT (OMV)?	
4.2 INSTALLATION	
4.3 USAGE	
4.4 LOG-IN DETAILS	
4.5 FIRST STEPS IN OPEN MEDIA VAULT	

5	UPLOADING IMAGES TO NAS FROM ANDROID PHONES	26-29
	5.1 CHOOSING AN APP	
	5.2 PROCEDURE	
6	THEORY FOR IMAGE CLASSIFICATION	30-36
	6.1 INTRODUCTION TO PACKAGES USED	
	6.2 IMAGE CLASSIFICATION	
	6.3 CONCEPT OF IMAGE CLASSIFICATION	
7	UNDERSTANDING THE CODE	36-42
	7.1 UNDERSTANDING THE PYTHON CODE	
	7.2 UNDERSTANDING THE SHELL SCRIPT	
8	CONCLUSION	43



रा.इ.सू.प्रौ.सं
NIELIT

ABOUT NIELIT

National Institute of Electronics & Information Technology (NIELIT), (erstwhile DOEACC Society), is an Autonomous Scientific Society under the administrative control of Ministry of Electronics & Information Technology (MoE&IT), Government of India, which was set up to carry out Human Resource Development and related activities in the area of Information, Electronics & Communications Technology (IECT). NIELIT is engaged both in Formal & Non Formal Education in the area of IECT Besides development of industry oriented quality education and training programs in the state of the art areas. NIELIT has endeavored to establish standards to be the country's premier institution for Examination and Certification in the field of IECT. It is also one of the National Examination Body, which accredits institutes/organizations for conducting courses in IT in the non-formal sector.

At present, NIELIT has thirty five (35) offices located at Agartala, Aizawl, Ajmer, Aurangabad, Calicut, Chandigarh, Chennai, Chuchuyimlang, Churachandpur, Delhi, Gangtok, Gorakhpur, Guwahati, Imphal, Itanagar, Jammu, Jorhat, Kohima, Kolkata, Kokrajhar, Leh, Lucknow, Lunglei, Pasighat, Patna, Ranchi, Ropar (Rupnagar City Centre), Senapati, Shillong, Shimla, Silchar, Srinagar, Srikakulam, Tezpur, Tura with its Headquarters at New Delhi. It is also well networked throughout India with the presence of about 800 institutes.

Over the last two decades, NIELIT has acquired very good expertise in IT training, through its wide repertoire of courses, ranging from 'O' Level (Foundation), 'A' Level (Advance Diploma), 'B' Level (MCA equivalent), 'C' Level (M-Tech level), IT literacy courses such as CCC (Course on Computer Concept), BCC (Basic Computer Course) and other such long term and short term courses in the non-formal sector like courses on Information Security. The basket of activities of NIELIT is further augmented by the wide range of projects that it undertakes. NIELIT has demonstrated its capability and capacity to undertake R&D projects, consultancy services, turnkey projects in office automation, software development, website development etc.

1. EVOLUTION OF SINGLE BOARD COMPUTERS

1.1 A BRIEF HISTORY

In its truest sense, an SBC has referred to a single PC board with the processor, memory and some type of I/O that allowed it to function as a computer – an example is the mid-70s “dyna-micro”, one of the first true single board computers. This differs from a traditional mass produced motherboard in that these early boards had expansion slots for many of the key additional peripherals like audio, video and network cards. Today, most consumer motherboards would be considered SBCs as most of the necessary functionality exists on a single motherboard, with the added bonus of being able to upgrade the existing functionality through the use of add-on cards. This is an abbreviated version of the rise of the modern PC.

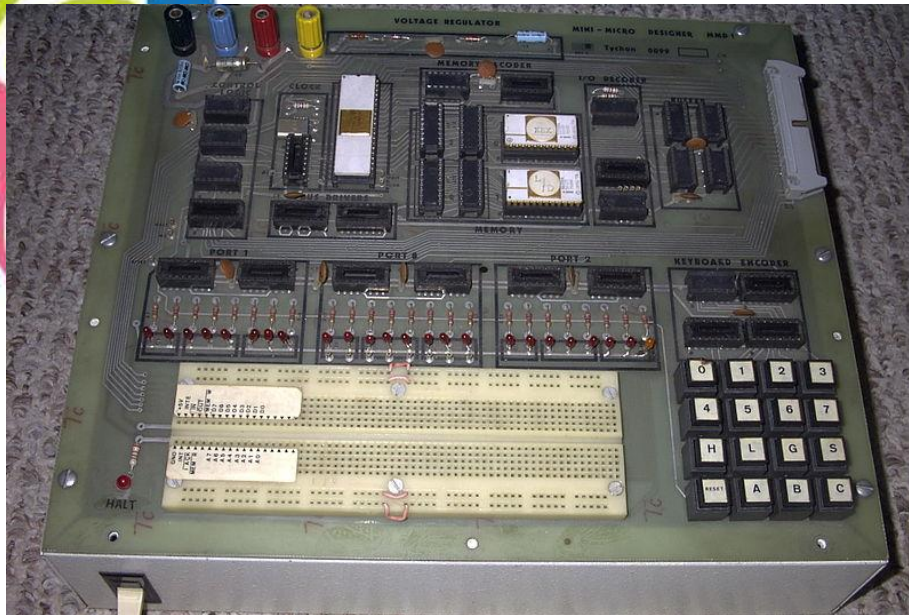


Figure 1.1

One of the first 10 MMD-1s, a prototype unit, produced by E&L Instruments in 1976. The "dyna-micro"/"MMD-1" was the world's first true single board computer. The MMD-1 had all components on a single printed circuit board, including memory, I/O, user input device, and a display. Nothing external to the single board except power was required to both program and run the MMD-1. The original design of the MMD-1 was called the "dyna-micro", but it was soon re-branded as the "MMD-1"

As stated previously, the term SBC generally refers to a microprocessor based board. But, just as in previous years, semiconductor manufacturers have often supplemented the launch of new products, especially microcontrollers, with development kits or demo boards that engineers could use to test out their newest silicon. These kits were generally used by professional engineers to test out the silicon for their next design. Once the necessary functionality of the microcontroller was confirmed, the designer went into laying out their own board to test out their proof of concept design.

In order to understand the rise of today's SBCs, we need only look back 10 years ago in Ivrea Italy where a team of designers were looking to develop a low cost, easy to use microcontroller based development kit that allowed people of all skill levels to make use of modern microcontrollers in their projects. This was the start of the Arduino platform. The rapid market acceptance of this prototyping platform paved the way for a new breed of designers entering into the electronics market. Today they are called makers, DIYers and hobbyists among other titles. As the microcontroller based Arduino market continued to grow, the cost of microprocessors and SOC's continued to drop dramatically thanks to the success of commercial processor platforms that integrated increasingly more functionality in a single package.



Figure 1.2

Beagleboard - The element14 BeagleBone Black is a full featured, internet enabled development platform that utilises the low cost Sitara AM3358 ARM Cortex-A8 processor from Texas Instruments and runs a variety of OS.

One of the next major developments in the rise of the modern SBC was on July 28, 2008 when the non-profit BeagleBoard.org. was formed to bring modern microprocessors development into the hands of engineers and designers through a low cost, open source community supported development platform known as the BeagleBoard.

With the Arduino and BeagleBone platforms in place and continuing to evolve in the hands of designers, developers and hobbyists, the world was about to experience one of the largest disruptions in both the consumer and industrial computing space - the introduction of the Raspberry Pi.

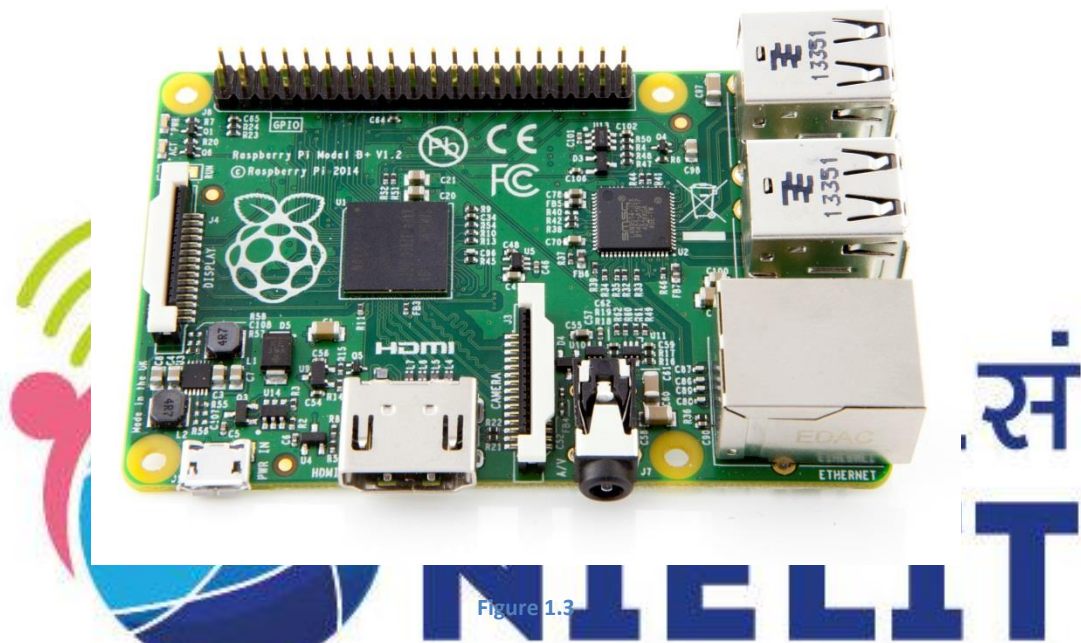


Figure 1.3

Raspberry Pi - The idea behind a tiny and affordable computer for kids came in 2006, when Eben Upton, Rob Mullins, Jack Lang and Alan Mycroft, based at the University of Cambridge's Computer Laboratory, became concerned about the year-on-year decline in the numbers and skills levels of the A-Level students applying to read Computer Science.

In 2006, a group from the University of Cambridge's Computer Laboratory, decided to address the need for a low cost computing platform that would allow students to learn how to program without the need for a home computer. The result was a \$35 (2,410.98 INR) single board computer named Raspberry Pi. While initially designed as a tool for students to learn to program, the Raspberry Pi was adopted by makers, designers, students and even professional engineers and helped to spark the current boom in interest in SBCs.

1.2 PRESENT DAY SBCs

Today, SBCs can be grouped into two main categories - open source and proprietary. Open source SBCs offer users access to both the hardware design and layout as well as access to the source code used on the board. This is ideal for all users as they can easily understand how the software and hardware operates and adopt the design to meet their end designs requirements or simply learn how a piece of hardware or software works. Proprietary SBCs on the other hand are generally designed for use in end applications or as a reference to be evaluated. They are often industrialised designs that have gone through the same type of testing that an end product requires and are often integrated into end product designs or installed in a rack mount configuration.

Current SBCs come with a wide variety of processor types, most with GPUs on-board. These processors range from X86 based processors from the traditional PC space (AMD and Intel) to ARM processors which have traditionally been used in the industrial and more recently mobile spaces. The most prevalent form of software used on SBCs is Linux with numerous derivations including Ubuntu, Fedora, Android, Debian and Arch Linux as well as FreeBSD and Windows CE.

The programming/debugging tools are also often free and open source, such as those based on the Eclipse IDE. Other tools that are tailored to a specific processor that are often used by professionals include ARM's DS-5 or vendor specific tools such as Texas Instruments' Code Composer Studio or Freescale's CodeWarrior.

1.3 TYPE OF SBCs

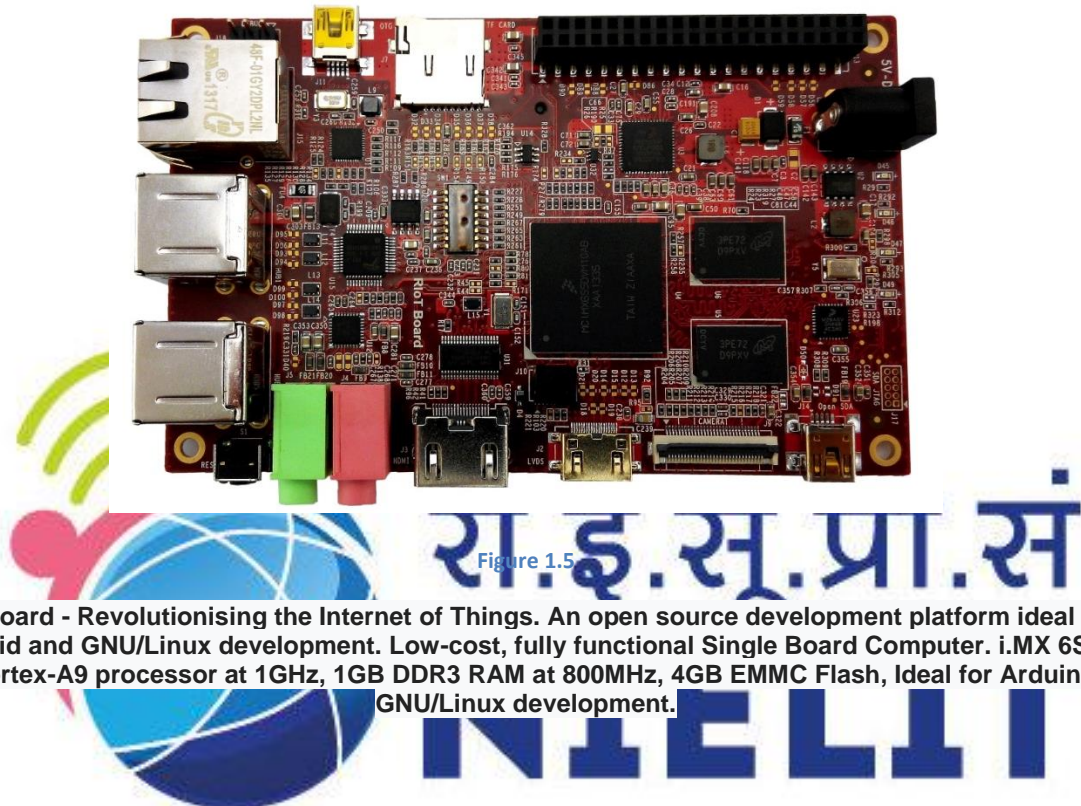
While SBCs can be used for a number of purposes, many have originally been designed for a specific application. A perfect example of this is the Raspberry Pi which was developed as an educational tool to help encourage and strengthen students programming skills. The BeagleBoard and BeagleBone were also developed to help educate and promote the benefits and usage of open source hardware and software in embedded computing.



Figure 1.4

Atmel Xplained - Fast prototyping and evaluation platform based on the SAMA5D3 ARM Cortex-A5. Powerful Atmel's SAMA5D36 Cortex-A5 MPU, 2Gb DDR2 RAM, 2Gb Flash, Dual Ethernet (GMAC + EMAC), Arduino R3-compatible header and LCD connector.

Numerous other SBCs have been developed in the past few years including Atmel's SAMA5D3 Xplained, which was designed for rapid prototyping development, and the RIOTboard which focuses on Android development to enable the development of the Internet of Things. Other well-known boards include the OlinuXino, PandaBoard, as well as a whole host of Allwinner ARM SoC based SBCs.



Riotboard - Revolutionising the Internet of Things. An open source development platform ideal for Android and GNU/Linux development. Low-cost, fully functional Single Board Computer. i.MX 6Solo ARM Cortex-A9 processor at 1GHz, 1GB DDR3 RAM at 800MHz, 4GB EMMC Flash, Ideal for Arduino and GNU/Linux development.

The long term success of an SBC, like most other products, relies heavily on the performance/price ratio. But what also weighs in just as heavily is the amount of available support for a particular board or range of boards. While some SBCs rely on a dedicated supplier or secondary support entity, most of the open source SBCs are supported through a community of developers. These communities strengthen the board's proposition by providing software updates from individuals as well as projects that showcase the features and many accessories that are often available to expand its functionality.

1.4 APPLICATIONS

One common variety of single board computer uses standardized computer form factors intended for use in a backplane enclosure. Some of these types are CompactPCI, PXI, VMEbus, VXI, and PICMG. SBCs have been built around various internal processing structures including the Intel architecture, multiprocessing architectures, and lower power processing systems like RISC and SPARC. In the Intel PC world, the intelligence and interface/control circuitry is placed on a plug-in board that is then inserted into a passive (or active) backplane. The end result is similar to having a system built with a motherboard, except that the backplane determines the slot configuration. Backplanes are available with a mix of slots (ISA, PCI, PCIX, PCI-Express, etc.), usually totaling 20 or fewer, meaning it will fit in a 19" rackmount enclosure (17" wide chassis).

Some single-board computers have connectors which allow a stack of circuit boards, each containing expansion hardware, to be assembled without a traditional backplane. Examples of stacking SBC form factors include PC/104, PC/104-Plus, PCI-104, EPIC, and EBX; these systems are commonly available for use in embedded control systems.

Stack-type SBCs often have memory provided on plug-cards such as SIMMs and DIMMs. Hard drive circuit boards are also not counted for determining if a computer is an SBC or not for two reasons, firstly because the HDD is regarded as a single block storage unit, and secondly because the SBC may not require a hard drive at all as most can be booted from their network connections.

1.5 THE FUTURE OF SBCs



Many of today's SBCs have become so powerful that they are beginning to have the capability of modern day PCs and tablets. This trend will continue as more powerful processors make their way into the embedded computing market as ever-increasing performance/price ratios rise, as well as additional manufacturers enter the frontier of supporting open source hardware and software for both DIYers and professionals alike.

An additional trend we will continue to see is the availability of more accessories or add-on boards to extend current SBC platforms, allowing users more options to control and access the outside world. This will benefit both DIYers and professionals alike. Professional engineers can take these accessories and quickly add additional needed functionality to their SBCs to develop working prototypes for projects currently on-hand. DIYers on the other hand that come from a programming background will have increased access to the analog electronics that are required to interface with the outside world.

Lastly, another trend that will most likely continue is the adoption of these SBCs into lower volume end products. Many of today's SBCs are as close to fully vetted designs as those developed specifically for end product usage. This is due to the fact that open source designs are equivalent to having a continual global design review with multitudes of designers and programmers updating and giving feedback on the boards and their software.

Additionally, as the design and testing of these boards is done through high quality design and manufacturing firms like AVID or Embest Technologies, the boards go through the same quality control as any other end product and often come with CE or FCC certifications. And, as the cost of these boards is often well below what an individual or company could produce the board for, entrepreneurs and small companies alike, see these boards as an ideal way to quickly bring designs to market without the overhead needed to develop new hardware, but instead focus on the software innovation that is often a key differentiator in today's end product designs.



2. KNOW YOUR RASPBERRY PI

2.1 WHAT IS A RASPBERRY PI?

The Raspberry Pi is a credit-card-sized computer that costs between \$5 (344.38 INR) and \$35(2,410.98 INR). It's available anywhere in the world, and can function as a proper desktop computer or be used to build smart devices.

The Pi was originally intended to be a microcomputer to teach children coding. Its scope has been expanded after hobbyists and engineers saw its potential, and it is now one of the most popular technology items in the world.

Raspberry Pi is essentially a bare-bones personal computer with a Linux operating system installed. They are incredibly cheap, with the models costing either USD \$25 (171.88 INR) or USD \$35 (2,410.98 INR). This system has essentially been created by the Raspberry Pi foundation in an effort to encourage young people to learn how to code and do computer programming, something that surprisingly few young people are getting into in this era of walled gardens and complete, closed off consoles.

You can expand the Raspberry Pi computer with modules, like adding a camera module or a touchscreen module, to increase the scope of the device.

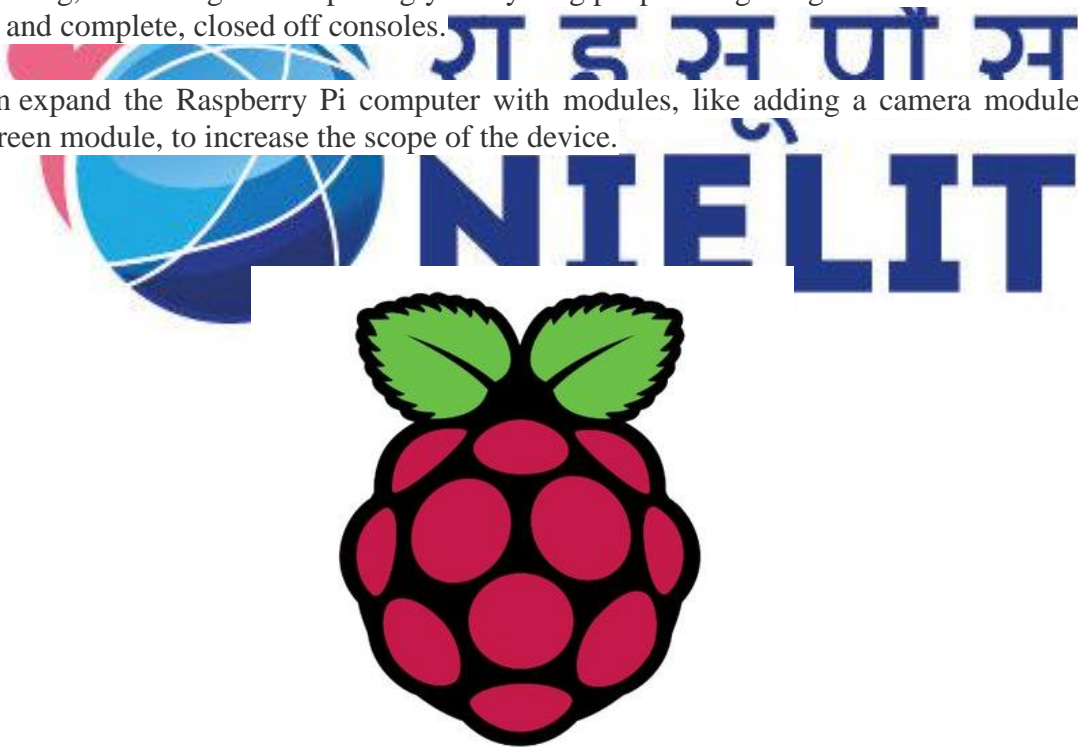


Figure 2.1

Logo of Raspberry Pi. | Website : <https://www.raspberrypi.org> | Twitter: <https://twitter.com/raspberrypi>

2.2 WHO INVENTED RASPBERRY PI?

The Raspberry Pi Foundation was formed in 2008 after a group of academics and technicians—Eben Upton, Rob Mulins, Jack Lang, Alan Mycroft, Pete Lomas, and David Braben—were concerned about students’ declining interest in computer sciences. Their solution was to come up with a low-cost computer to inspire children and make it more accessible.

The idea was that these tiny computers would allow for easy basic programming. Its low power usage and cost were expected to make Pis more easily available in classrooms.



Figure 2.2

Eben Upton is also a founder and former trustee of the Raspberry Pi Foundation, and now CEO of Raspberry Pi (Trading).

Today, a few of the original members still act as the Foundation’s trustees, while Upton has taken charge as CEO and project lead.

2.3 WHY IS IT CALLED RASPBERRY PI?

The “Raspberry” derives is an homage to early computer companies being named after fruit, like Apple, Tangerine Computer Systems, Apricot Computers, and Acorn (which inspired the microcomputer’s design). The “Pi” derives from the original idea to make a small computer to **run only the Python programming language.**

2.4 WHEN WAS RASPBERRY PI LAUNCHED?

The first commercially available Raspberry Pi unit was launched on February 19, 2012, and sales started ten days later. This version could run Linux-based desktop operating systems, and featured 256MB of RAM, one USB port, and no Ethernet port. This was named the Model A.

2.5 WHY TO CHOOSE RASPBERRY PI?

As mentioned, the Raspberry Pi is created to be something to encourage learning, innovation, and experimentation. The computer is powerful for its size, but it costs about the same as a textbook, making it accessible to even those who are low income or living in a poorer country or community. Since our society is becoming more and more reliant on computers, we will need more and more people who know how to code and program, but most young people aren't learning computer science or being exposed to it at all.

2.6 HOW IS IT SO CHEAP?

At \$25 (171.88 INR) for Model A and \$35 (2,410.98 INR) for Model B, the Raspberry Pi is much less expensive than most scientific calculators or cellular phones. These computers are supposed to be accessible for all, so they needed to be as inexpensive as possible, and to accomplish this, these case-free single board systems are made as pared down as possible. This means that users will have to provide their own keyboard or monitor (or television set) in order to use the Raspberry Pi.

2.7 WHAT IS DIFFERENCE BETWEEN RASPBERRY PI'S MODELS?

Raspberry Pi models can be a bit confusing. There are two levels to the naming system. Pi 1, Pi 2, and Pi 3 indicate the "generation" of the model, where roughly Pi 1 is 2012-14 models, Pi 2 is 2015 models, and Pi 3 is 2016 models. So 3 is better than 2, which is better than 1.

Model A, A+, B, and B+ indicate the power and features. It's not like grades though, A is lower than B. There's also the Raspberry Pi Zero, a \$5(344.38 INR) microcomputer for simple projects. It's severely limited in comparison to the Model A or Model B series. Currently, only the Pi 3 Model B, the Pi 2 Model B, the Pi 1 Model B+, and Pi 1 Model A+ are available for purchase.

Here's a quick comparison chart:

Pi Zero	Pi 1 Model A+	Pi 1 Model B+	Pi 2 Model B	Pi 3 Model B
1 GHz 32-bit Single Core processor	700 MHz 32-bit Single Core processor	700 MHz 32-bit Single Core processor	900 MHz 32-bit Quad Core processor	1.2 GHz 64-bit Quad Core Processor
512 MB RAM	512 MB RAM	512 MB RAM	1 GB RAM	1 GB RAM
Broadcom VideoCore IV GPU	Broadcom VideoCore IV GPU	Broadcom VideoCore IV GPU	Broadcom VideoCore IV GPU	Broadcom VideoCore IV GPU
2 micro USB ports	1 micro USB port	1 micro USB port	1 micro USB port	1 micro USB port
No USB ports	1 USB port	4 USB ports	4 USB ports	4 USB ports
1 mini HDMI port, no HDMI	1 HDMI port, no mini HDMI	1 HDMI port, no mini HDMI	1 HDMI port, no mini HDMI	1 HDMI port, no mini HDM
1 microSD slot	1 SD/MMC slot	1 microSD slot	1 microSD slot	1 microSD slot
No dedicated audio port	3.5mm audio out port	3.5mm audio out port	3.5mm audio out port	3.5mm audio out port
No Wi-Fi, No Ethernet	No Wi-Fi, No Ethernet	No Wi-Fi, Ethernet via USB Adapter	No Wi-Fi, Ethernet via USB Adapter	Onboard Wi-Fi, Ethernet port
No Bluetooth	No Bluetooth	No Bluetooth	No Bluetooth	Bluetooth 4.1
65x30 mm (Half of Standard Pi Size)	85.60×56.5 mm (Standard Pi Size)	85.60×56.5 mm (Standard Pi Size)	85.60×56.5 mm (Standard Pi Size)	85.60×56.5 mm (Standard Pi Size)
\$5	\$20	\$25	\$35	\$35

Figure 2.3

2.8 WHERE ARE RASPBERRY PI'S USED?

The Raspberry Pi has won hearts all across the globe, from astronauts to hobbyists. In fact, right now, there are two Raspberry Pi's orbiting the earth, conducting experiments aboard the International Space Station. British astronaut Tim Peake is heading the Astro Pi project, challenging UK school students to write code for experiments that he can perform in space.

Back on earth, a team of computer engineers at the University of Southampton put together 64 Raspberry Pis to build their own supercomputer! Each Pi has a 16GB memory card, making it a 1TB supercomputer. It's like putting together a LEGO set, the makers say, and is an ideal Pi project for schools.

Then there's a group of geeks who are making a autonomous marine unmanned surface vessel (that is, a self-driving boat) with a Raspberry Pi acting as the brain. This drone will swim across the Atlantic Ocean, fitted with sensors to take scientific measurements along the way. They call it FishPi, and it's fascinating.

There are several other places that Raspberry Pis are being used in. And forget about such professional or hobbyist cases, there are plenty of real-world practical applications for average Joe too.

2.9 WHAT IS A RASPBERRY PI GOOD FOR?

Regular people can use the Raspberry Pi in a wide variety of tasks. It's perfect for projects where you need a computer but don't require much processing power, want to save on space, and keep the costs low. Here's a brief list of some ideal uses of the Pi.

- Teach kids (or yourself) how to code
- Use it as a desktop PC
- Easily make a media center with Rasplex or an always-on downloading machine
- Build a motion capture security camera or a DIY pan and tilt camera with Raspberry Pi
- Make your own retro gaming console
- You can make a world clock or an FM radio with the Pi Zero
- Put together a low-cost time-lapse photography camera with the camera module

There are plenty of other things that the Raspberry Pi is good for, and a quick Google search should find the right thing for you. If you're new to Pi, there are some projects for beginners to ease you into tinkering with your Pi.

2.10 HOW MANY RASPBERRY PI HAVE BEEN SOLD?

The last figures released by the company come from February 2016, when they announced they had sold over eight million units of the Raspberry Pi. Of these, over three million were of the Raspberry Pi 2, which had been launched only a year before that.

This makes the Raspberry Pi the UK's all-time best-selling computer, according to The Guardian, breaking the decades-old record set by the Amstrad PCW.

2.11 WHERE CAN YOU BUY RASPBERRY PI?

The direct Distributors of Raspberry Pi in India are FactoryForward, ElementzOnline, Kitsnspares, CrazyPi etc.

Some other websites are:

- ProtoCentral
- RoboCraze
- Element 14 India
- Rhydo Labz
- FlipKart
- Amazon etc.



If you want to buy in Chandigarh you can buy from Rajindra Videos, Booth No. 176, Sector 35-D, Near Local Bus Stop, Chandigarh, 160022.

3. GETTING STARTED WITH RASPBERRY PI

Our project is done on RASPBERRY PI 3 MODEL B+, the latest in the edition, so our main focus will be on the model as mentioned.

3.1 HARDWARE SPECIFICATIONS

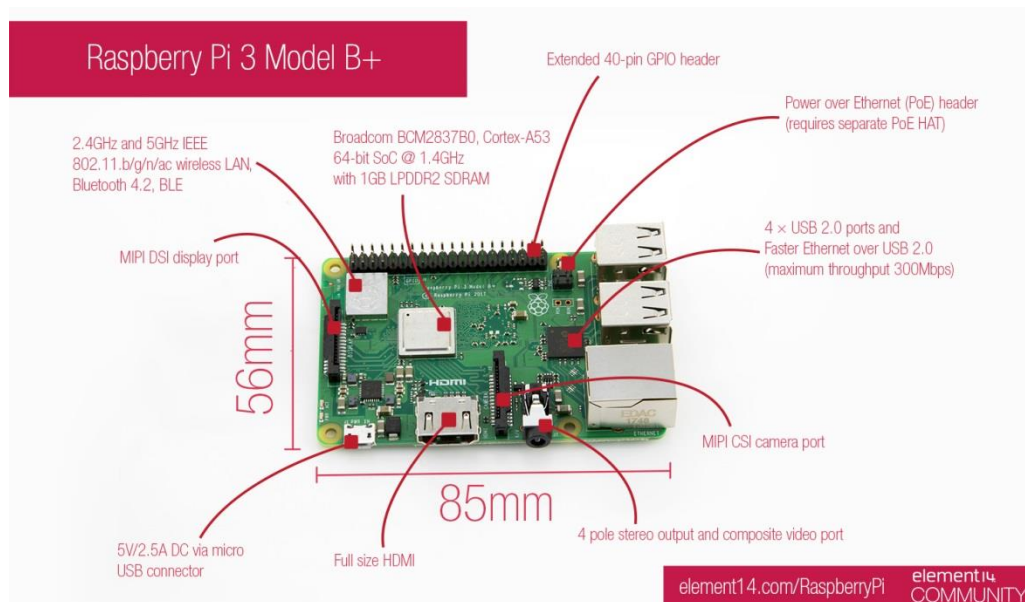


Figure 3.1

A well labeled image of Raspberry Pi 3 Model B+.

Specifications of Raspberry Pi 3 Model B+:

- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- 1GB LPDDR2 SDRAM
- 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE
- Gigabit Ethernet over USB 2.0 (maximum throughput 300 Mbps)
- Extended 40-pin GPIO header
- Full-size HDMI
- 4 USB 2.0 ports
- CSI camera port for connecting a Raspberry Pi camera

- DSI display port for connecting a Raspberry Pi touchscreen display
- 4-pole stereo output and composite video port
- Micro SD port for loading your operating system and storing data
- 5V/2.5A DC power input
- Power-over-Ethernet (PoE) support (requires separate PoE HAT)

Before you plug anything into your Raspberry Pi, make sure that you have all the equipment you need:

- A monitor with the correct cable and adapter
- A micro USB power supply
- A wired keyboard and mouse, or a wireless keyboard and mouse with a Bluetooth adapter
- A micro SD card
- A Raspberry Pi

3.2 INSTALLATION OF AN OPERATING SYSTEM

Beginners should start with the NOOBS (New Out Of Box Software) operating system installation manager, which gives the user a choice of operating system from the standard distributions. SD cards with NOOBS pre-installed should be available from any the global distributors and resellers

Alternatively, NOOBS is available for download on the Raspberry Pi website: raspberrypi.org/downloads.

Once you've downloaded the NOOBS zip file, you'll need to copy the contents to a formatted SD card on your computer.

To set up a blank SD card with NOOBS:

- Format an SD card which is 8GB or larger as FAT.
- Download and extract the files from the NOOBS zip file.
- Copy the extracted files onto the SD card that you just formatted, so that this file is at the root directory of the SD card. Please note that in some cases it may extract the files into a folder; if this is the case, then please copy across the files from inside the folder rather than the folder itself.
- On first boot, the "RECOVERY" FAT partition will be automatically resized to a minimum, and a list of OSes that are available to install will be displayed.

Note: If you're formatting an SD (or micro SD) card that has a capacity over 32GB (i.e. 64GB and above), then see the separate SDXC formatting instructions.

WINDOWS If you are a Windows user, we recommend formatting your SD card using the SD Association's Formatting Tool, which can be downloaded from sdcard.org. Instructions for using the tool are available on the same site.

MAC OS The SD Association's Formatting Tool is also available for Mac users, although the default OS X Disk Utility is also capable of formatting the entire disk. To do this, select the SD card volume and choose Erase with MS-DOS format.

LINUX For Linux users we recommend gparted (or the command line version parted). Norman Dunbar has written up instructions for Linux users.

The following operating systems are currently included in NOOBS:

- Raspbian
- LibreELEC
- OSMC
- Recalbox
- Lakka
- RISC OS
- Screenly OSE
- Windows 10 IoT Core
- TLXOS

As of NOOBS v1.3.10 (September 2014), only Raspbian is installed by default in NOOBS. The others can be installed with a network connection.

Raspbian is the recommended operating system for normal use on a Raspberry Pi.

Reference: For official documentation and other details regarding Raspberry Pi visit www.raspberrypi.org

4. SETTING UP A NAS ON A RASPBERRY PI USING OMV

4.1 WHAT IS OPEN MEDIA VAULT (OMV)?

“openmediavault” is the next generation network attached storage (NAS) solution based on Debian Linux. It contains services like SSH, (S)FTP, SMB/CIFS, DAAP media server, RSync, BitTorrent client and many more. Thanks to the modular design of the framework it can be enhanced via plugins.

“openmediavault” is primarily designed to be used in small offices or home offices, but is not limited to those scenarios. It is a simple and easy to use out-of-the-box solution that will allow everyone to install and administrate a Network Attached Storage without deeper knowledge.

For more information, visit: www.openmediavault.org

4.2 INSTALLATION



रा.इ.स.प्रौ.सं

NOTE: during the installation process the ssh access for the user pi will be disabled. If you log out of your ssh session during the installation you don't have a way of going back in over ssh. Access will be restored from the OMV GUI.

Now we can start installing OMV. Put the below scripts in 2 different .sh files in the home folder of the pi user (/home/pi). Note that you may need to add execution permission to the scripts – using `chmod +x one.sh`

1. `cat <<EOF >> /etc/apt/sources.list.d/openmediavault.list`
2. `deb http://packages.openmediavault.org/public erasmus main`
3. `# deb http://downloads.sourceforge.net/project/openmediavault/packages erasmus main`
4. `## Uncomment the following line to add software from the proposed repository.`
5. `# deb http://packages.openmediavault.org/public erasmus-proposed main`
6. `# deb http://downloads.sourceforge.net/project/openmediavault/packages erasmus-proposed main`
7. `## This software is not part of OpenMediaVault, but is offered by third-party`
8. `## developers as a service to OpenMediaVault users.`
9. `# deb http://packages.openmediavault.org/public erasmus partner`


```

10. # deb http://downloads.sourceforge.net/project/openmediavault/packages erasmus
    partner
11. EOF
12. export LANG=C
13. export DEBIAN_FRONTEND=noninteractive
14. export APT_LISTCHANGES_FRONTEND=none
15. apt-get update
16. apt-get upgrade
17. apt-get --allow-unauthenticated install openmediavault-keyring
18. apt-get update
19. apt-get --yes --force-yes --auto-remove --show-upgraded \
20.     --no-install-recommends \
21.     --option Dpkg::Options::="--force-confdef" \
22.     --option Dpkg::Options::="--force-confold" \
23.     install postfix openmediavault
24. # Initialize the system and database.
25. dpkg-reconfigure openmediavault
26. omv-initsystem

```

This will add the repository to your sources, install some dependencies and initialize OMV.

Execute it with:

1. `sudo ./one.sh`
2. `sudo ./two.sh`

NOTE: Script one should NEVER be executed again. This will cause problems in your sources file.

Now you can navigate to the IP address of your pi (ifconfig command) using a browser (no port is required, OMV runs by default on port 80).

4.3 USAGE

Default login credentials are admin:openmediavault. You can now connect to the gui and experience OMV in all its glory. Format your external drives to a file format for Linux. OMV

can handle this the best by default and it will increase your speed drastically. (note that only Linux based computers can read this by default.)

After OMV has picked up your drive, you have to mount the file system. If you are having issues, have OMV format your filesystem by manually unmounting the drive, creating a new filesystem via the GUI and mount it. Once that is done you are free to create folders, configure them for samba, manage your users and so much more.

4.4 LOG-IN DETAILS

Web interface

- User: admin
- Password: openmediavault

Client (SSH, console)

- User: root
- Password: <The password that you have set during installation>

The server has been configured by DHCP. Login as root user on the shell and run the command:

ifconfig

to get the current IP address.

Open a web browser and enter http:// followed by your IP address: http://192.168.XXXX.XXXX/ to go to the web login:

Enter the default login details user: admin, password: openmediavault: And the OpenMediaVault Admin interface appears.

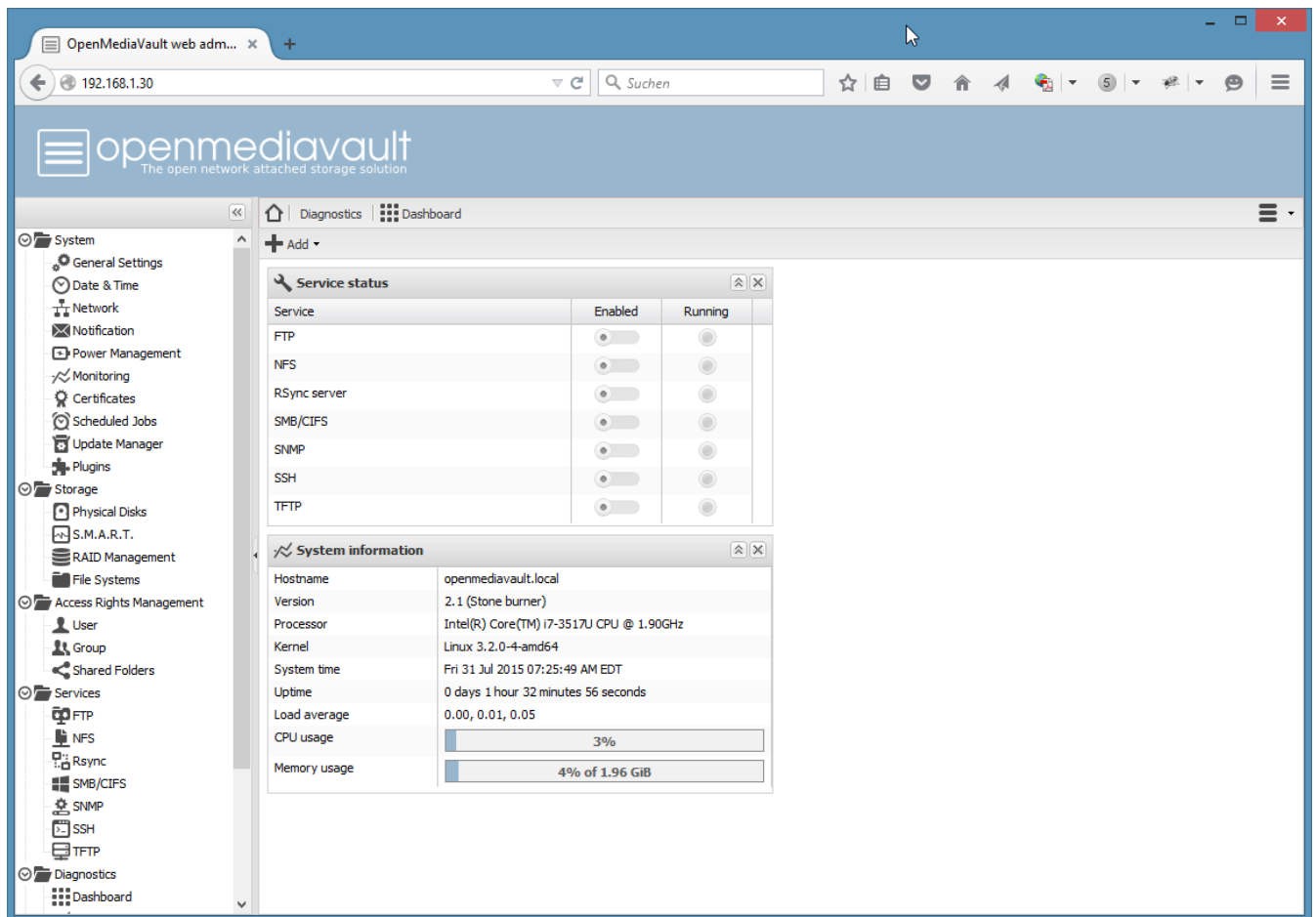


Figure 4.1

OMV INTERFACE WINDOW AFTER LOG-IN

4.5 FIRST STEPS IN OPEN MEDIA VAULT

4.5.1 Changing the web admin password

- To change the web administrator password, go to "General Settings" and change to the tab "Web Administrator Password":
- Enter the new password and press on the "Save" button in the upper left corner of the input form.
- Services like FTP, SMB and SSH are disabled by default.
- In the next step, we will enable FTP and SMB (Microsoft Windows Share).

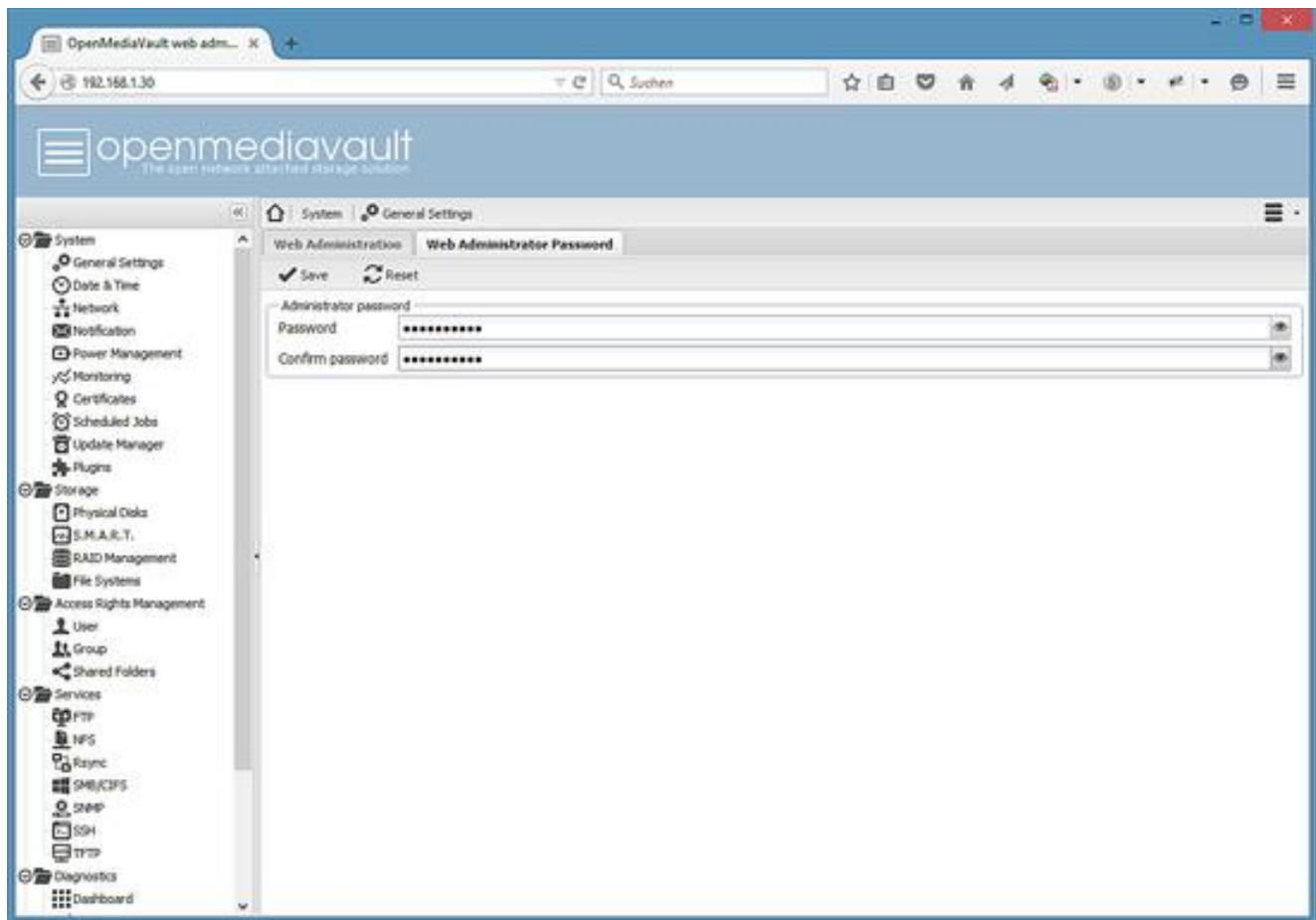


Figure 4.2

4.5.2 Enable FTP

- Go to Services > FTP and enable the "Enable" Checkbox.

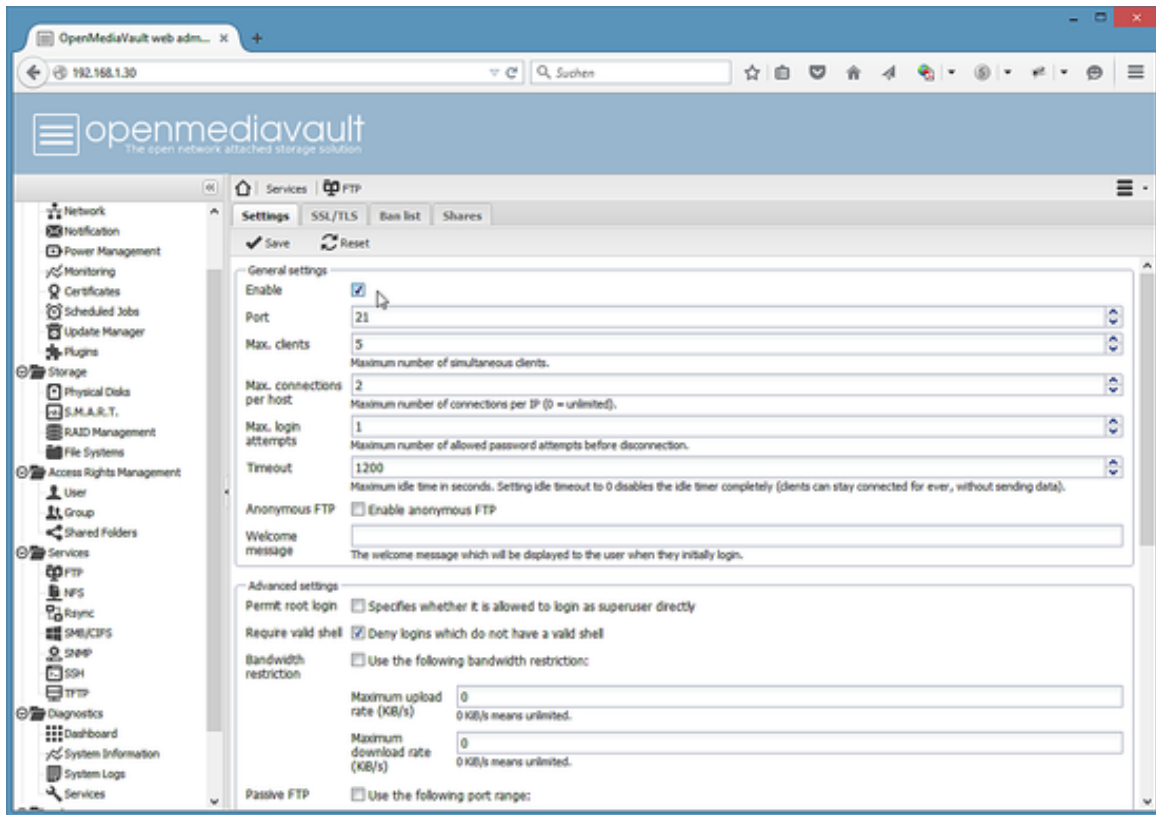


Figure 4.3

- Press the "Save" button in the upper left corner.
- Finally, confirm that the changes shall really be applied.
- Now go to Services > SMB/CIFS and enable this service like you did with FTP.
- Same procedure for the SSH service. Go to Services > SSH and enable the service

4.5.3 Creating a file system as data storage volume

OpenMediaVault needs a separate hard disk or partition to store data (storage volume). In my case, I'll use a second hard disk `/dev/sdb`. Click on **Storage > File Systems > Create** to add the second hard disk as a storage volume. The name of my storage volume is "data".

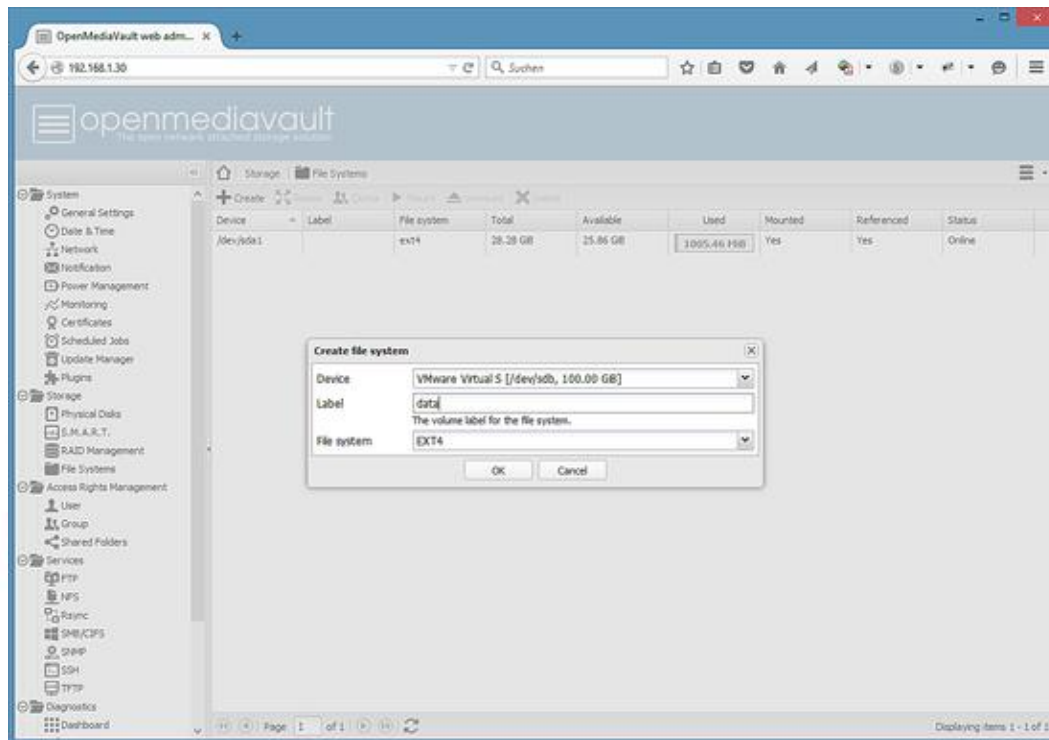


Figure 4.4

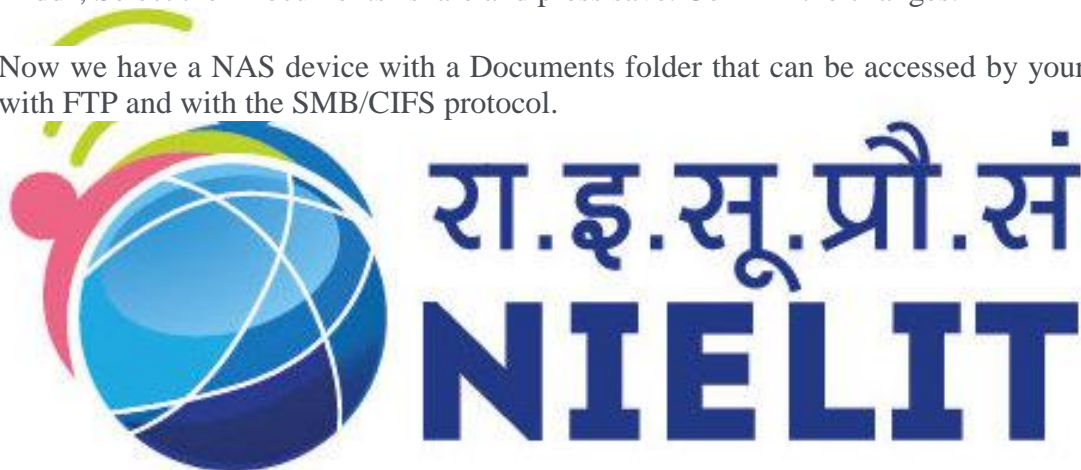
The final list of storage devices shall look like this. Select the "data" volume in the list and click on the "Mount" button to mount the volume. Only mounted volumes will appear as option in the shared folders volume list.

4.5.4 Adding a user

- Now we can add a user to access your file shares. Click on "Access Rights Management" > "User" > Add:
- Enter the user details: username, email address, and password. Then confirm the changes.

4.5.5 Adding a file share

- To store files on the NAS drive, we need a file share that can be accessed by our user. Click on "Access Rights Management" > "Shared Folders" > "Add".
- I will add a folder named "Documents" with the path "Documents/" on the data volume.
- The next step is to grant read/write permissions to the user "till". Click on the "Documents" share in the list and then on the Icon "Privileges" in the menu above the list. This will open the privileges Window, enable "Read/Write" for the user and press save.
- Finally add the share to the services that shall be able to use them. To enable the share for FTP, go to Services > FTP > Shares, Click on "Add", Select the "Documents" share and press save. Confirm the changes.
- Use the same procedure for SMB/CIFS: go to Services > SMB/CIFS > Shares, Click on "Add", Select the "Documents" share and press save. Confirm the changes.
- Now we have a NAS device with a Documents folder that can be accessed by your user with FTP and with the SMB/CIFS protocol.



5. UPLOADING IMAGES TO NAS FROM ANDROID PHONES

5.1 CHOOSING AN APP

There are many applications available for both Android phones some of them are:

- ES File Explorer
- AndSMB
- Network Browser
- Upload 2 NAS Lite
- Astro File Manager

In our project we have chosen Astro File Manager app. for uploading images to our Raspberry Pi NAS. Any other application can be used and all of these applications are available on Google Play Store. The procedure for iPhone users is also identical but for our convenience, we are working on an Android phone.

Our Raspberry Pi NAS can also be accessed by a Linux or Windows System, given they are on the same network, whose procedure can easily be found on internet but in relevance to our project, we have restricted ourselves to Android phones only.

5.2 PROCEDURE

- Make sure your phone and the raspberry are on the same network. An easy way to do it is to convert your Pi into a wi-fi hotspot and connect your phone to it. An intermediate wi-fi router can also be used for this purposes.
- Open Astro File Manager in your phone.
- Under Storage Sections, click on Local Network.
- Fill, in the necessary details and we are good to go.

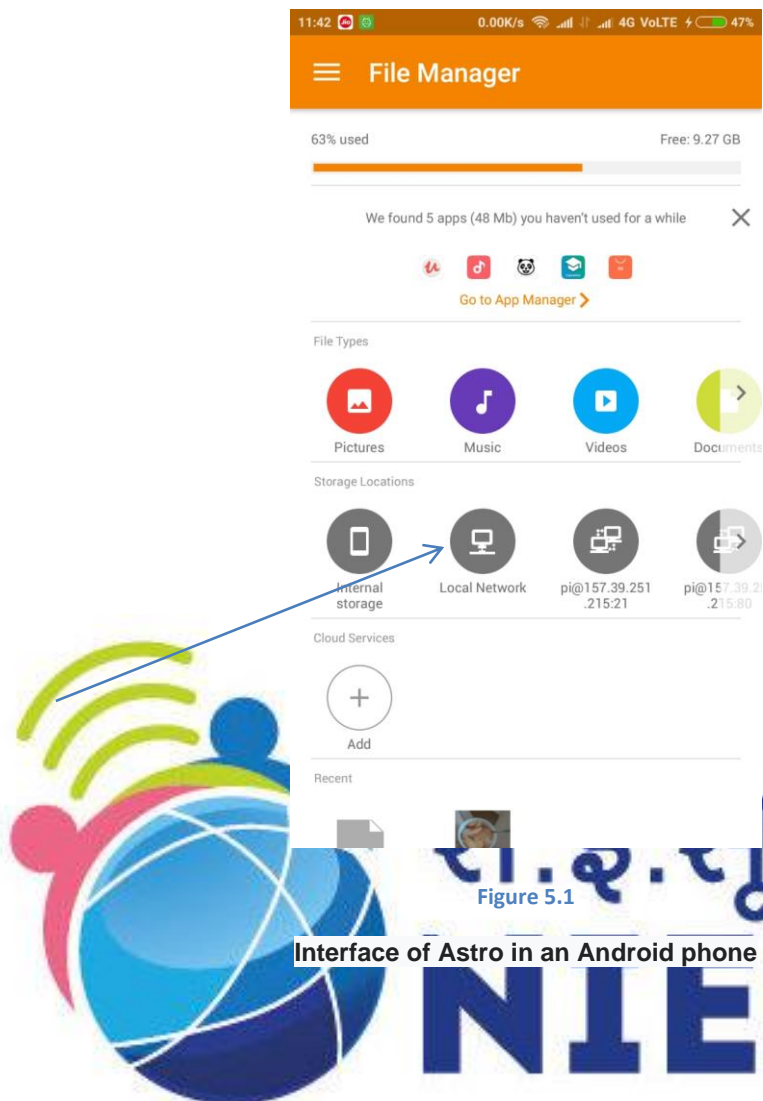


Figure 5.1

Interface of Astro in an Android phone

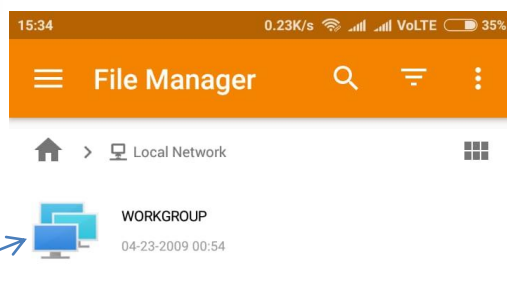


Figure 5.2

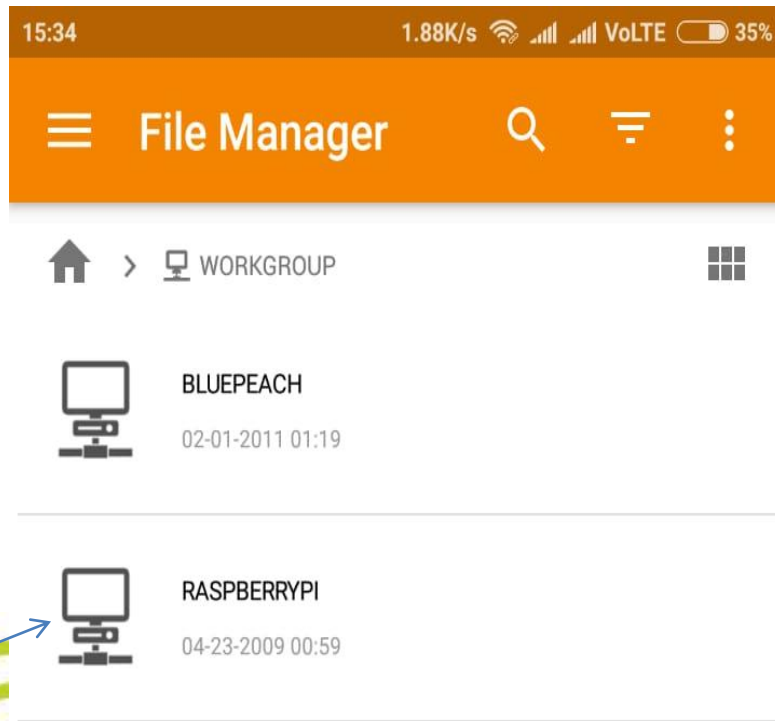


Figure 5.3

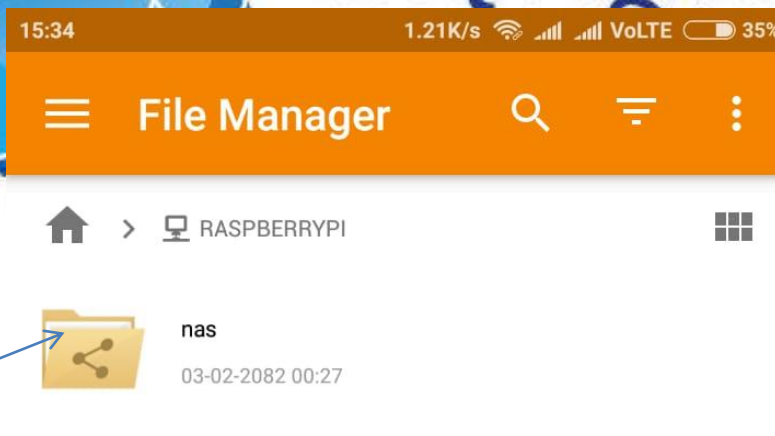


Figure 5.4

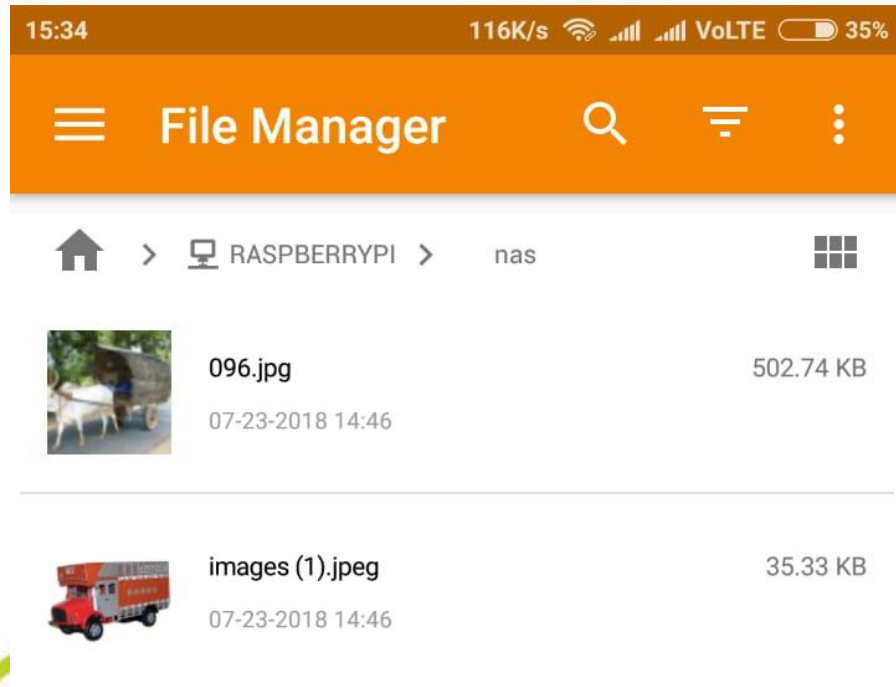


Figure 5.5

6. THEORY FOR IMAGE CLASSIFICATION

6.1 INTRODUCTION TO PACKAGES USED

6.1.1 NumPy

It is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It contains various features including these important ones. A powerful N-dimensional array object, Sophisticated (broadcasting) functions, Tools for integrating C/C++ and Fortran code, Useful linear algebra, Fourier transform, and random number capabilities. Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

a. Arrays in NumPy:

NumPy's main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In NumPy dimensions are called axes. The number of axes is rank. NumPy's array class is called ndarray. It is also known by the alias array.

b. Array creation:

Often, the elements of an array are originally unknown, but its size is known. Hence, NumPy offers several functions to create arrays with initial placeholder content. These minimize the necessity of growing arrays, an expensive operation. To create sequences of numbers, NumPy provides a function analogous to range that returns arrays instead of lists. Arrange, line space, Reshaping array and Flatten array.

c. Array Indexing:

Knowing the basics of array indexing is important for analyzing and manipulating the array object. NumPy offers many ways to do array indexing. Slicing: Just like lists in python, NumPy arrays can be sliced. As arrays can be multidimensional, you need to specify a slice for each dimension of the array. Integer array indexing: In this method, lists are passed for indexing for each dimension. One to one mapping of corresponding elements is done to construct a new arbitrary array. Boolean array indexing: This method is used when we want to pick elements from array which satisfy some condition.


d. Basic operations:

Plethora of built-in arithmetic functions are provided in NumPy. Operations on single array: We can use overloaded arithmetic operators to do element-wise operation on array to create a new array. In case of $+=$, $-=$, $*=$ operators, the existing array is modified. Unary operators: Many unary operations are provided as a method of ndarray class. This includes sum, min, max, etc. These functions can also be applied row-wise or column-wise by setting an axis parameter. Binary operators: These operations apply on array element wise and a new array is created. You can use all basic arithmetic operators like $+$, $-$, $/$, $*$, etc. In case of $+=$, $-=$, $=$ operators, the existing array is modified. Universal functions (ufunc): NumPy provides familiar mathematical functions such as sin, cos, exp, etc. These functions also operate element wise on an array, producing an array as output

e. Sorting array:

There is a simple np.sort method for sorting NumPy arrays. Let's explore it a bit.

6.1.2 Open CV



OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. In simple language it is library used for Image Processing. It is mainly used to do all the operation related to Images. It is used to do work in many ways such as Read and Write Image, Detection of faces and its features, Detection of shapes like Circle, rectangle etc. in a image. e.g.: Detection of coin in images, Text recognition in images. E.g. Reading Number Plates, Modifying image quality and colors e.g Instagram, CamScanner and Developing Augmented reality apps. It can support languages such as. C++, Android SDK, Java And Python is a software library (Pre-defined algorithms and APIs) that can be used to analyses the data from the Camera of an Embedded system or your computer or anything that captures images(e.g. From a LIDAR). It provides hundreds of functions for the capture, analysis, and manipulation of visual data and can eliminate some of the hassle programmers' face when developing applications that rely on computer vision. Portions of the library also provide user interface and pattern recognition functions. OpenCV has been employed in both practical and creative applications including self-piloting vehicles and new forms of digital art. OpenCV has a wide range of applications in traditional computer vision applications such as optical character recognition or medical imaging. For example, OpenCV can detect Bone fractures¹. OpenCV can also help classify skin lesions and help in the early detection of skin melanomas². However, OpenCV coupled with the right processor and camera can become a powerful new class of computer vision enabled IoT sensor. This type of design can scale from simple sensors to multi-camera video analytics arrays. See Designing Scalable IoT Architectures for more information. IoT developers can use OpenCV to build embedded computer vision sensors for detecting IoT application events such as motion detection or people detection. Designers can also use OpenCV

to build even more advanced sensor systems such as face recognition, gesture recognition or even sentiment analysis as part of the IoT application flow. IoT applications can also deploy OpenCV on Fog nodes at the Edge as an analytics platform for a larger number of camera based sensors. For example, IoT applications use camera sensors with OpenCV for road traffic analysis, Advanced Driver Assistance Systems (ADAS)³, video surveillance⁴, and advanced digital signage with analytics in visual retail applications⁵. When developers integrated OpenCV with a neural-network backend, it unleashed the true power of computer vision. Using this approach, OpenCV works with Convolutional Neural Networks (CNN) and Deep Neural Networks (DNN) to allow developers to build innovative and powerful new vision applications. To target multiple hardware platforms, these integrations need to be cross platform by design. Hardware optimization of deep learning algorithms breaks this design goal. The OpenVX architecture standard proposes resource and execution abstractions.

6.1.3 ArgParse

The `argparse` module includes tools for building command line argument and option processors. It was added to Python 2.7 as a replacement for `optparse`. The implementation of `argparse` supports features that would not have been easy to add to `optparse`, and that would have required backwards-incompatible API changes, so a new module was brought into the library instead. `optparse` is now deprecated. `argparse` is a complete argument processing library. Arguments can trigger different actions, specified by the action argument to `add_argument()`. Supported actions include storing the argument (singly, or as part of a list), storing a constant value when the argument is encountered (including special handling for true/false values for Boolean switches), counting the number of times an argument is seen, and calling a callback to use custom processing instructions. The default action is to store the argument value. If a type is provided, the value is converted to that type before it is stored. If the dest argument is provided, the value is saved using that name when the command-line arguments are parsed. After all of the arguments are defined, parse the command-line by passing a sequence of argument strings to `parse_args()`. By default, the arguments are taken from `sys.argv[1:]`, but any list of strings can be used. The options are processed using the GNU/POSIX syntax, so option and argument values can be mixed in the sequence.

The return value from `parse_args()` is a Namespace containing the arguments to the command. The object holds the argument values as attributes, so if the argument's dest is set to "myoption", the value is accessible as `args.myoption`. The `argparse` module makes it easy to write user-friendly command-line interfaces. The program defines what arguments it requires, and `argparse` will figure out how to parse those out of `sys.argv`. The `argparse` module also automatically generates help and usage messages and issues errors when users give the program invalid

arguments. The `argparse` module provides an easy, declarative interface for creating command line tools, which knows how to:

1. parse the arguments and flags from `sys.argv`
2. convert arg strings into objects for your program
3. Format and print informative help messages

6.2 IMAGE CLASSIFICATION

In image classification, an image is classified according to its visual content. For example, does it contain an airplane or not. An important application is image retrieval – searching through an image dataset to obtain (or retrieve) those images with particular visual content.

While human visual image interpretation techniques rely on shape, size, pattern, tone, texture, shadows, and association, digital image interpretation relies mainly on color, i.e. on comparisons of digital numbers found in different bands in different parts of an image.

The objective of digital image classification procedures is to categorize the pixels in an image into land cover classes. The output is a thematic image with a limited number of feature classes as opposed to a continuous image with varying shades of gray or varying colors representing a continuous range of spectral reflectance.

Two major categories of image classification techniques include unsupervised (calculated by software) and supervised (human-guided) classification.

This post is focused on Deep Neural Networks (DNN) and Convolutional Neural Network (CNN) and lies between the two main categories. Training images are labeled in a supervised way by an analyst, but the feature learning and classification are automatically done by software in an unsupervised way

Image classification refers to the task of extracting information classes from a multiband raster image. The resulting raster from image classification can be used to create thematic maps. Depending on the interaction between the analyst and the computer during classification, there are two types of classification: supervised and unsupervised.

With the ArcGIS Spatial Analyst extension, there is a full suite of tools in the Multivariate toolset to perform supervised and unsupervised classification (see an overview of the Multivariate toolset). The classification process is a multi-step workflow; therefore, the Image Classification toolbar has been developed to provide an integrated environment to perform classifications with the tools. Not only does the toolbar help with the workflow for performing unsupervised and supervised classification, it also contains additional functionality for analyzing

input data, creating training samples and signature files, and determining the quality of the training samples and signature files. The recommended way to perform classification and multivariate analysis is through the Image Classification toolbar. Image classification is a process of mapping numbers to symbols:

$$f(x): x \in D; x \in \mathbb{R}^n, D = \{c_1, c_2, \dots, c_L\}$$

Number of bands = n ;

Number of classes = L

$f(\cdot)$ is a function assigning a pixel vector x to a single class in the set of classes D

6.3 CONCEPT OF IMAGE CLASSIFICATION

In order to classify a set of data into different classes or categories, the relationship between the data and the classes into which they are classified must be well understood

- To achieve this by computer, the computer must be trained
- Training is key to the success of classification

Classification techniques were originally developed out of research in Pattern Recognition field. Computer classification of remotely sensed images involves the process of the computer program learning the relationship between the data and the information classes important aspects of accurate classification are


- Learning techniques
- Feature sets

CNN

In machine learning, a convolutional neural network (CNN, or ConvNet) is a type of feed-forward artificial neural network in which the connectivity pattern between its neurons is inspired by the organization of the animal visual cortex, whose individual neurons are arranged in such a way that they respond to overlapping regions tiling the visual field. Convolutional networks were inspired by biological processes and are variations of multilayer perceptrons designed to use minimal amounts of preprocessing. They have wide applications in image and video recognition, recommender systems and natural language processing. In a CNN the input image passes through a series of convolutional, nonlinear, pooling (down sampling), and fully connected layers, and gets an output. The output can be a single class or a probability of classes that best describes the image.

In machine learning, a convolutional neural network (CNN or ConvNet) is a class of deep, feed-forward artificial neural networks, most commonly applied to analyzing visual imagery. CNNs use a variation of multilayer perceptrons designed to require minimal preprocessing.[1] They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics.[2][3] Convolutional networks were inspired by biological processes[4] in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field. CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns the filters that in traditional algorithms were hand-engineered. This independence from prior knowledge and human effort in feature design is a major advantage. A CNN consists of an input and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of convolutional layers, pooling layers, fully connected layers and normalization layers[citation needed]. Description of the process as a convolution in neural networks is by convention. Mathematically it is a cross-correlation rather than a convolution. This only has significance for the indices in the matrix, and thus which weights are placed at which index.

Convolutional



Convolutional layers apply a convolution operation to the input, passing the result to the next layer. The convolution emulates the response of an individual neuron to visual stimuli. Each convolutional neuron processes data only for its receptive field. Although fully connected feed forward neural networks can be used to learn features as well as classify data, it is not practical to apply this architecture to images. A very high number of neurons would be necessary, even in shallow (opposite of deep) architecture, due to the very large input sizes associated with images, where each pixel is a relevant variable. For instance, a fully connected layer for a (small) image of size 100 x 100 has 10000 weights for each neuron in the second layer. The convolution operation brings a solution to this problem as it reduces the number of free parameters, allowing the network to be deeper with fewer parameters.[8] For instance, regardless of image size, tiling regions of size 5 x 5, each with the same shared weights, requires only 25 learnable parameters. In this way, it resolves the vanishing or exploding gradients problem in training traditional multi-layer neural networks with many layers by using back propagation [citation needed].

Pooling

Convolutional networks may include local or global pooling layers[clarification needed], which combine the outputs of neuron clusters at one layer into a single neuron in the next layer.[9][10] For example, max pooling uses the maximum value from each of a cluster of neurons at the prior layer. Another example is average pooling, which uses the average value from each of a cluster of neurons at the prior layer[citation needed].

Fully connected

Fully connected layers connect every neuron in one layer to every neuron in another layer. It is in principle the same as the traditional multi-layer perceptron neural network (MLP).

Weights

CNNs share weights in convolutional layers, which mean that the same filter (weights bank [clarification needed]) is used for each receptive field [clarification needed] in the layer; this reduces memory footprint and improves performance.

7. UNDERSTANDING THE CODE

7.1 UNDERSTANDING THE PYTHON CODE

7.1.1 Introduction

We would like to stress on certain points:

- One should not be training neural networks on the Raspberry Pi (unless you're using the Pi to do the "Hello, World" equivalent of neural networks — but again, we would still argue that your laptop/desktop is a better fit).
- With the Raspberry Pi there just isn't enough RAM.
- The processor is too slow.
- And in general it's not the right hardware for heavy computational processes.
- Instead, we should first train your network on your laptop, desktop, or deep learning environment.
- Once the network is trained, you can then deploy the neural network to your Raspberry Pi.
- We will now demonstrate how we can use the Raspberry Pi and pre-trained deep learning neural networks to classify input images.

7.1.2 Deep learning on the Raspberry Pi with OpenCV

When using the Raspberry Pi for deep learning we have two major pitfalls working against us:

- Restricted memory (only 1GB on the Raspberry Pi 3).
- Limited processor speed.

This makes it near impossible to use larger, deeper neural networks.

Instead, we need to use more computationally efficient networks with a smaller memory/processing footprint such as MobileNet and SqueezeNet. These networks are more appropriate for the Raspberry Pi; however, you need to set your expectations accordingly — you should not expect blazing fast speed.

In this tutorial we'll specifically be using SqueezeNet.

7.1.2 What is SqueezeNET?

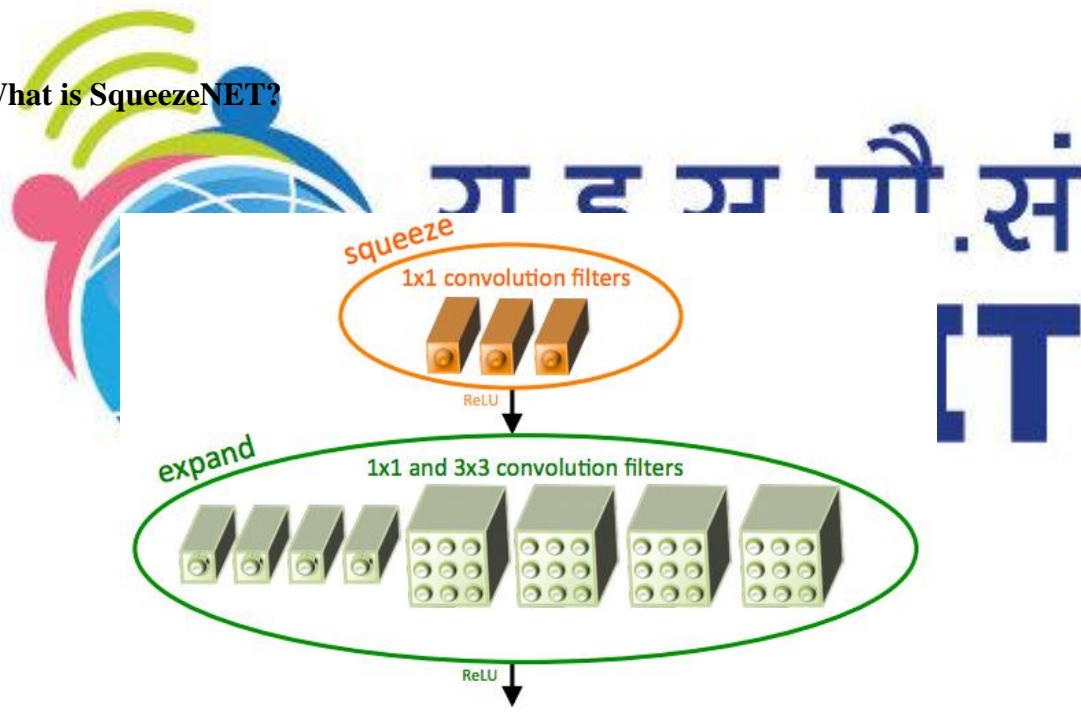


Figure 7.1

The “fire” module in SqueezeNet, consisting of a “squeeze” and an “expand” (Iandola et al., 2016).

SqueezeNet was first introduced by Iandola et al. in their 2016 paper, SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size.

The title alone of this paper should pique your interest. State-of-the-art architectures such as ResNet have model sizes that are >100MB. VGGNet is over 550MB. AlexNet sits in the middle of this size range with a model size of ~250MB.

In fact, one of the smaller Convolutional Neural Networks used for image classification is GoogLeNet at ~25-50MB (depending on which version of the architecture is implemented).

The real question is: Can we go smaller?

As the work of Iandola et al. demonstrates, the answer is: Yes, we can decrease model size by applying a novel usage of 1×1 and 3×3 convolutions, along with no fully-connected layers. The end result is a model weighing in at 4.9MB, which can be further reduced to < 0.5MB by model processing (also called “weight pruning” and “sparsifying a model”).

SqueezeNet can classify images in approximately half the time of GoogLeNet, making it a reasonable choice when applying deep learning on your Raspberry Pi.

7.1.3 Running a deep neural network on the Raspberry Pi

To get started, create a new file named `pi_deep_learning.py`, and insert the following source code.

```
1. # import the necessary packages
2. import numpy as np
3. import argparse
4. import time
5. import cv2
```

Lines 2-5 simply import our required packages.

From there, we need to parse our command line arguments:

```
6. # construct the argument parse and parse the arguments
7. ap = argparse.ArgumentParser()
8. ap.add_argument("-i", "--image", required=True,
9.     help="path to input image")
10. ap.add_argument("-p", "--prototxt", required=True,
```



```

11. help="path to Caffe 'deploy' prototxt file")
12. ap.add_argument("-m", "--model", required=True,
13. help="path to Caffe pre-trained model")
14. ap.add_argument("-l", "--labels", required=True,
15. help="path to ImageNet labels (i.e., syn-sets)")
16. args = vars(ap.parse_args())

```

As is shown on **Lines 8-16** we have four *required* command line arguments:

- **--image:** The path to the input image.
- **--prototxt:** The path to a Caffe prototxt file which is essentially a plaintext configuration file following a JSON-like structure. I cover the anatomy of Caffe projects in my PyImageSearch Gurus course.
- **--model:** The path to a pre-trained Caffe model. As stated above, you'll want to train your model on hardware which packs much more punch than the Raspberry Pi — we can, however, leverage a small, pre-existing model on the Pi.
- **--labels:** The path to class labels, in this case ImageNet “syn-sets” labels.

Next, we'll load the class labels and input image from disk:

```

17. # load the class labels from disk
18. rows = open(args["labels"]).read().strip().split("\n")
19. classes = [r[r.find(" ") + 1:].split(",")[0] for r in rows]

20. # load the input image from disk
21. image = cv2.imread(args["image"])

```

NOTE: The **synset_words.txt** file can be downloaded from:

https://github.com/HoldenCaulfieldRye/caffe/blob/master/data/ilsrvrc12/synset_words.txt .

You'll see on each line/row there is an ID and class labels associated with it (separated by commas). But we have modified the file into five categories viz: Vehicle, Nature, Food, Animal and Miscellaneous.

Lines 18 and 19 simply read in the labels file line-by-line (**rows**) and extract the first relevant class label. The result is a **classes** list containing our class labels.

Then, we utilize OpenCV to load the image on **Line 21**.

Now we'll make use of OpenCV 3.3's Deep Neural Network (DNN) module to convert the **image** to a **blob** as well as to load the model from disk:

```
22. # our CNN requires fixed spatial dimensions for our input image(s)
23. # so we need to ensure it is resized to 227x227 pixels while
24. # performing mean subtraction (104, 117, 123) to normalize the input;
25. # after executing this command our "blob" now has the shape:
26. # (1, 3, 227, 227)
27. blob = cv2.dnn.blobFromImage(image, 1, (227, 227), (104, 117, 123))

28. # load our serialized model from disk
29. print ("[INFO] loading model...")
30. net = cv2.dnn.readNetFromCaffe(args["prototxt"], args["model"])
```

Be sure to make note of the comment preceding our call to **cv2.dnn.blobFromImage** on **Line 27** above.

Common choices for width and height image dimensions inputted to Convolutional Neural Networks include 32×32 , 64×64 , 224×224 , 227×227 , 256×256 , and 299×299 . In our case we are pre-processing (normalizing) the image to dimensions of 227×227 (which are the image dimensions SqueezeNet was trained on) and performing a scaling technique known as mean subtraction. Its importance can be found over internet.

In case you missed it above, it is worth noting here that we are loading a *pre-trained* model. The training step has already been performed on a more powerful machine and is outside the scope of this report.

Now we're ready to pass the image through the network and look at the predictions:

```
31. # set the blob as input to the network and perform a forward-pass to
32. # obtain our output classification
33. net.setInput(blob)
34. start = time.time()
35. preds = net.forward()
36. end = time.time()
37. print("[INFO] classification took {:.5} seconds".format(end - start))

38. # sort the indexes of the probabilities in descending order (higher
39. # probability first) and grab the top-5 predictions
40. preds = preds.reshape((1, len(classes)))
41. idxs = np.argsort(preds[0])[::-1][:5]
```

To classify the query **blob**, we pass it forward through the network (**Lines 33-37**) and print out the amount of time it took to classify the input image (**Line 43**).

We can then sort the probabilities from highest to lowest (**Line 40**) while grabbing the top five **predictions** (**Line 41**).

The remaining lines (1) draw the highest predicted class label and corresponding probability on the image, (2) print the top five results and probabilities to the terminal, and (3) display the image to the screen:

But, we only want to get the top prediction and simple return it. For this purpose we have used the following lines:

```
42. text = "Label: {}, {:.2f}%".format(classes[idxs[0]],
43.   preds[0][idxs[0]] * 100)
44. cv2.putText(image, text, (5, 25), cv2.FONT_HERSHEY_SIMPLEX,
45.   0.7, (0, 0, 255), 2)
46. print (classes[idxs[0]],(preds[0][idxs[0]])*100)
```

Source: The above code is a modification of original lines of code written by *Adrian Rosebrock* which can be found on www.pyimagesearch.com

7.2 UNDERSTANDING THE SHELL SCRIPT

The Shell Script as follows:

- `#!/bin/bash`
- `indir=$(echo $* | grep -oE -- --indir=[^\]*)`
- `indir=${indir##--indir=}`
- `outdir=$(echo $* | grep -oE -- --outdir=[^\]*)`
- `outdir=${outdir##--outdir=}`
- `pushd /home/pi/Desktop/newimagerecognitiondata`
- `#SCRIPTDIR<-change this to model directory`
- `if [-z "$indir"] && [-z "$outdir"]`

- then
 1. echo "Usage: \$0 --indir=/home/pi/e --outdir=/home/pi/b"
 2. exit -1
- fi

- for f in "\$indir"/*
- do
 1. class=\$(python2.7 ./pi_deep_learning.py --prototxt models/bvlc_googlenet.prototxt --model models/bvlc_googlenet.caffemodel --labels synset_words.txt --image "\$f" | grep -v INFO)
 2. class=\$(echo "\$class" | grep -oE '\..*\ ' | tr -d "'")
 3. [! -d "\$outdir"/"\$class"] && mkdir -p "\$outdir"/"\$class"
 4. cp -v "\$f" "\$outdir"/"\$class"
- done

In above shell script, the image uploaded is copied to the respective folder on the basis of the top prediction (prediction with higher probability as compared to rest four categories) that in which among the five classifications in which one it fits the most. If the folder of the name of that particular category is not present then it will be created and the image will be moved into it.

Now, the above shell-script can be run periodically at a certain time interval. There are certain ways to do it. One of it is that, we can simply use the *while* command with a delay of desired time interval, the delay can be given by *sleep* command. Another way, which we have used here is using *crontab* and the code is as follows:

```
*/2 * * * * /home/pi/movefile --indir=/sharedfolders/nas --outdir=/sharedfolders/sumeer >>
/var/log/nas/cron.log
```

The above line code allows us to run our shell-script on a interval of every 2 minutes, that is, image sorting will occur every two minutes.

To find more about the commands used in this shell-script visit:

- <https://www.raspberrypi.org/documentation/linux/usage/commands.md>
- www.unix.com
- <http://www.linuxforums.org>
- <https://community.linuxmint.com/tutorial/view/244>
- <https://www.sdstate.edu/information-technology/linux-command-line-documentation>

8. CONCLUSION

At NIELIT, we took up training of 6 weeks of Raspberry Pi with Python where we were made familiar with the hardware and software aspects of Raspberry Pi with focus on its real life and industry based applications. Along with this, concept of shell programming, Python and C programming on Pi, Node.js, MqTT, Apache web-server, NAS and interfacing various electronic components to Pi was also introduced to us .

Tools like Raspberry Pi and necessary peripherals and Anaconda-Navigator were made easily available to us and we were guided to work efficiently on these tools. Our main work was done using a Raspberry Pi 3 along with JuPyter-Notebook and simple text file on Raspbian for our shell script. This project can be extended further with no limits. Currently with it, we are able to upload images from our Android, Windows and Linux devices on the Raspberry Pi NAS and within two minutes the images will sorted in the folder of their respective category. We should keep in mind that the path of files is mentioned correctly in the code and we should be ready to remove any kind of error, if it comes while working.

Pi has limited processing capabilities and should be prevented from overheating. A heat sink can be put on the Pi to cool down its temperature and prevent any hardware failure. The references to the source are mentioned wherever necessary and be viewed later anytime.

